Pseudo-Gray Coding

KENNETH ZEGER AND ALLEN GERSHO, FELLOW, IEEE

Abstract—**A pseudo-gray code is an assignment of n-bit binary indexes to 2^n points in a Euclidean space so that the Hamming distance between two points corresponds closely to the Euclidean distance. Pseudo-Gray coding provides a redundancy-free error protection scheme for vector quantization (VQ) of analog signals when the binary indexes are used as channel symbols on a discrete memoryless channel and the points are signal codevectors. Binary indexes are assigned to codevectors in a way that reduces the average quantization distortion introduced in the reproduced source vectors when a transmitted index is corrupted by channel noise. A globally optimal solution to this problem is generally intractable due to an inherently large computational complexity. A locally optimal solution, the Binary Switching Algorithm, is introduced, based upon the objective of minimizing a useful upper bound on the average system distortion. The algorithm yields a significant reduction in average distortion, and converges in reasonable running times. The use of pseudo-Gray coding is motivated by the increasing need for low bit-rate VQ-based encoding systems that operate on noisy channels, such as in mobile radio speech communications.**

I. Introduction

A long-standing problem in quantization theory is how to effectively combat the performance degradation caused by noisy channels. One approach is to use redundancy bits for error control coding. A fairly straightforward argument shows, however, that superior performance can always be achieved by instead using these extra bits to design a higher resolution quantizer with no explicit error control. In effect, the error control coding is built into the higher resolution quantizer.

On the other hand, for a quantizer with a fixed codebook and no error correction, performance gain can be achieved by judiciously assigning channel words (i.e., binary indexes) to codevectors. Intuitively, vectors that are "close" to each other should be assigned indexes which differ in as few bit positions as possible. In this way, channel errors cause an index to be decoded as a vector which approximates the codevector that was supposed to be correctly decoded (i.e., without channel noise). The problem of how to make these assignments in an effective manner has essentially been an unanswered question until recently. This paper proposes a solution to this question based on a certain upper bound approximation that turns out to be exact in the case of the mean-square distortion criterion. An index assignment algorithm is presented that proves very effective in controlling performance degradation caused by channel noise.

A. Background

Vector Quantization (VQ) has been established as an important and successful source coding technique in many digital communication applications (see, for example, [1]). Real-time implementations of VQ coding systems are beginning to emerge and are likely to play an increasing role in voice, image, and video communications systems in the future. Often, the binary indexes produced by a VQ coder account for most or all of the information transmitted to a receiver across a noisy channel. This commonly occurs in mobile radio and satellite communications, where low bit-rate digital speech applications can suffer severe quality degradation from the effects of channel noise on transmitted codevector indexes.

Several studies of classical Gray coding have been conducted for scalar quantization of sources. The numerical and perceptual effects of channel errors on PCM signals have been investigated in [2]-[6]. Rydbeck and Sundberg [4] demonstrated the importance of proper index assignment in scalar quantizer design for noisy channels. The design of optimal codes for discrete-alphabet uniformly distributed sources using a mean-square fidelity criterion was considered by Wolf and Redinbo [7] and Redinbo [8], [9]. Many papers have studied combined source and channel coding, but generally ignore the channel codeword assignment question (see [10]-[16]). The problem of finding an optimal index mapping function is, however, considered by Farvardin and Vaishampayan [17] in the context of combined source-channel coding of scalar PCM signals. Cox *et al.* [18] have designed an ad hoc channel error protection scheme for subband coding of speech signals, by identifying important channel bits and using rate compatible punctured convolutional coding. Specific techniques for reducing the effects of channel errors without adding redundancy bits have been given for certain image coding applications using Walsh Transforms [19] and DPCM [20].

Few research efforts in the past have focused on the index assignment problem for vector quantizers. The growing popularity of vector quantization as a realizable source coding technique has drawn attention to the advantages of codevector index assigning schemes, and several limited heuristic algorithms have recently been described [21]-[23]. In [24], a technique is described that recursively constructs good index assignments by examining the related minimum weight hypercube problem. By using a well-known solution to the weighted matching problem in graph theory, their algorithm is both efficient and yields good results for certain quantized images corrupted by channel noise. Finally, simulated annealing techniques have been utilized for improving the index assignment functions of vector quantizers in [25]-[27].

This paper offers a systematic examination of the index assignment problem in vector quantization and presents an algorithm that effectively reduces the average distortion of a VQ system by rearranging the positions of codevectors in a given codebook. The algorithm is guaranteed to converge to a locally optimal state after a finite number of iterations, and operates over a wide class of common distortion functions. Pseudo-Gray coding can provide additional protection from channel errors in applications where the primary channel coding is achieved as part of the digital modulation scheme, as in trellis-coded modulation. This work was motivated by such a situation where the application was speech coding for mobile satellite communications [21].

In Section II, a formal model of vector quantization on a noisy channel is given, and the pseudo-Gray coding problem is introduced. In Section III, equations describing an upper bound on the expected overall system distortion due to quantization and channel errors are given. The class of distortion measures considered includes metrics and positive powers of norms. For each distortion measure, it is shown that one can minimize a quantity (common to each distortion function) independent of the source statistics, with a

Paper approved by the Editor for Quantization, Speech/Image Coding of the IEEE Communications Society. Manuscript received July 13, 1988; revised October 24, 1989. This work was supported in part by the Jet Propulsion Laboratory, California Institute of Technology, sponsored by NASA, and by the General Electric Company, Bell Communications Research, Inc., and the State of California MICRO program. This paper was presented in part at the International Conference on Communications, Philadelphia, PA, June 1988.

K. Zeger is with the Department of Electrical Engineering, University of Hawaii, Holmes Hall 483, Honolulu, HI 96822.

A. Gersho is with the Department of Electrical and Computer Engineering, Communications Research Laboratory, University of California, Santa Barbara, CA 93106.

IEEE Log Number 9039096.

0090-6778/90/1100-2147$01.00

certain upper bound assumption. The general case of multiple bit errors occurring in a single transmitted binary index is initially examined. The results are then specialized to the case of at most a single bit error per transmitted index. These results lead to the goal of seeking a rearrangement of a given codebook that minimizes an expected distortion term. In the single bit error case, it is noted that this minimization is independent of the value of the channel's crossover probability. Section IV describes the Binary Switching Algorithm in detail, and discusses the computational complexity requirements of the algorithm. Numerical results of experiments using the Binary Switching Algorithm are given in Section V, which exhibit the achievable gain in SNR using pseudo-Gray coding. It is shown that on binary symmetric channels, over a wide range of bit error probabilities, one can obtain SNR gains in the range of 1–5 dB with vector quantized waveforms. Such an improvement is often quite significant, and particularly so considering that an increase in the bit rate is not required.

II. Vector Quantization Model

A. VQ on a Noiseless Channel

Vector Quantization encodes each vector from a sequence of source vectors with a channel symbol—a binary word chosen from a finite set. A typical VQ system contains a finite predetermined collection of codevectors (a codebook), and a vector distortion measure which, when given two vectors, yields a distance (or distortion) between them. A sequence of input vectors (e.g., a block of sequential samples of a waveform) is coded by the VQ system by associating with each input vector the binary word (or index) of a codevector whose distance from the input vector is minimized. This index is subsequently transmitted to a receiver which decodes the codevector associated with the index (by a table lookup operation) and uses the codevector as an approximation to the original input vector to the system.

The design of codebooks for a given digital encoding system is performed before the actual use of the VQ coder. Only VQ systems that use static codebooks, which retain the same codevectors throughout their use, will be considered. At the time of VQ operation, the entire codebook is completely determined and available to both a transmitter (encoder) and a receiver (decoder). In systems with dynamically changing codebooks, the algorithms described in this paper can be generalized by reexecuting the procedures whenever a codebook undergoes change, although this may prove very costly in terms of computational complexity.

A *vector quantizer* Q is a mapping of p-dimensional Euclidean space $\boldsymbol{R}^p$ into a finite subset Y of $\boldsymbol{R}^p$ given by

$$Q: \boldsymbol{R}^p \rightarrow Y \tag{1}$$

where $Y = \{y_0, y_1, \cdots, y_{N-1}\}$ and $y_i \in \boldsymbol{R}^p$ for $0 \le i \le N-1$. The ordered set Y is known as a *codebook* and the N elements of Y are called *codevectors*. For convenience, it is assumed that the size of the codebook Y is $N = 2^b$, where b is a positive integer. The subscripts of the codevectors are the codevector *indexes*, each index representing a b-bit binary channel word, written as decimal integers for notational brevity. The set of N vector indexes can be written as $\{0,1\}^b$, the set of all binary words of length b bits.

A vector quantizer Q can be decomposed into two separate mappings: a *coder* and a *decoder*. A coder $C: \boldsymbol{R}^p \rightarrow \{0,1\}^b$ maps the vectors of $\boldsymbol{R}^p$ to b-bit indexes, and a decoder $D: \{0,1\}^b \rightarrow Y$ maps indexes to vectors in $\boldsymbol{R}^p$. Thus,

$$Q = D \circ C \tag{2}$$

where $D(k) = y_k$ for $k \in \{0,1\}^b$.

Corresponding to an N-point vector quantizer Q is a partition $\{R_i\}$ where, for each $i \in \{0,1\}^b$,

$$R_i = C^{-1}(i) = \{x \in \boldsymbol{R}^p : C(x) = i\}. \tag{3}$$

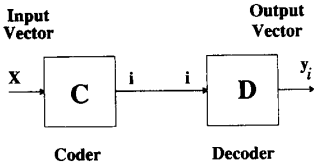


Fig. 1. Block diagram of VQ on noiseless channel.

From this definition, it follows that

$$\bigcup_{i=0}^{N-1} R_i = \boldsymbol{R}^p \text{ and } R_i \cap R_j = \phi \qquad \text{whenever } i \neq j. \tag{4}$$

A vector quantizer is completely specified by its partition $\{R_i\}$ and output set Y. Fig. 1 is a block diagram of a vector quantizer on a noiseless channel.

B. VQ on a Noisy Channel

The effects of channel errors on indexes can result in significant distortion in decoded vectors. The magnitude of this degradation is measured in terms of the distortion function defined for a VQ system. The indexing of vectors in a codebook influences the average distortion caused by channel errors. By arranging the codevectors such that index errors cause incorrectly received vectors to be close on average to the original vectors, the expected distortion due to channel noise can be reduced. The problem of finding the best codebook rearrangement involves searching every possible index assignment for the one that yields the best possible performance. This task requires enormous computational complexity due to the combinatoric nature of the problem, since there are $N!$ assignments of indexes to N codevectors. A suboptimal solution to this problem is thus sought.

Define the *bitwise addition* operation $\oplus: \{0,1\}^b \times \{0,1\}^b \rightarrow \{0,1\}^b$ as follows. If $i = i_1 i_2 \cdots i_b \in \{0,1\}^b$ and $j = j_1 j_2 \cdots j_b \in \{0,1\}^b$, where $i_k, j_k \in \{0,1\}$ for $1 \le k \le b$, then $i \oplus j$ is the element of $\{0,1\}^b$ with binary representation $c_1 c_2 \cdots c_b$, where $c_k = i_k + j_k \pmod 2$ for $1 \le k \le b$. That is, each binary digit c_k is the logical exclusive-OR of i_k and j_k.

If the indexes in the set $\{0,1\}^b$ are transmitted across a noisy channel, their values in general will be received as different indexes in the same set $\{0,1\}^b$. Hence, a *memoryless noisy channel* can be represented by a mapping $\tau: \{0,1\}^b \rightarrow \{0,1\}^b$ given by

$$\tau(i) = i \oplus \eta \qquad (i \in \{0,1\}^b) \tag{5}$$

where η is a random variable taking on values from the set $\{0,1\}^b$. τ is a function of the random variable η that describes the effect of channel errors upon a transmitted binary index.

Let S_N denote the set of all one-to-one functions $\pi: \{0,1\}^b \rightarrow \{0,1\}^b$. Each of the $N!$ bijections (permutations) $\pi \in S_N$ is called an *index assignment function* for the quantizer Q. For each index $i \in \{0,1\}^b$, π uniquely maps i to another index of $\{0,1\}^b$, namely, $\pi(i)$. A permutation can be thought of as a rearrangement of the order in which vectors appear in a VQ codebook.

Definition: For any vector quantizer $Q = D \circ C$ and for any permutation π of $\{0,1\}^b$, a *noisy channel vector quantizer*, Q_π, is a mapping from $\boldsymbol{R}^p$ to the set of codevectors Y given by

$$Q_\pi = D \circ \pi^{-1} \circ \tau \circ \pi \circ C \tag{6}$$

where τ is a noisy channel mapping, and π^{-1} is the inverse function of the permutation π.

For any input vector, the output quantized vector produced by Q_π is a function of the random variable η associated with the noisy channel and the input random variable from $\boldsymbol{R}^p$. For a given channel, Q_π can be represented by the triple (C, D, π). A block diagram depicting a noisy channel vector quantizer is shown in Fig. 2. In the case of a noiseless channel, the channel noise random variable η is identically zero, τ is the identity mapping on $\{0,1\}^b$,

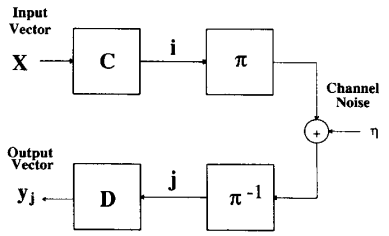


Fig. 2. Block diagram of VQ on discrete memoryless channel. C = quantizer encoder, D = quantizer decoder, π = index permutation.

and $Q_\pi = Q$. The choice of index assignment function π has no effect on Q_π's performance on a noiseless channel.

Define a real-valued *distortion function* $d: \boldsymbol{R}^p \times \boldsymbol{R}^p \to \boldsymbol{R}$ to be a function that assigns to every two vectors in $\boldsymbol{R}^p$ a nonnegative number describing their distance from each other. We assume a finite-variance source random variable $\boldsymbol{X}$ from $\boldsymbol{R}^p$ with some given cumulative distribution function. For a fixed vector quantizer Q and a given channel, one seeks to find a noisy channel vector quantizer Q_π that minimizes the average distortion given by

$$e_\pi \doteq E\big[d(X, Q_\pi(X))\big]. \tag{7}$$

For a given vector quantizer Q, a noisy channel vector quantizer is completely specified by choosing a permutation function π. Hence, for a given Q, one seeks to minimize the average distortion e over all permutation functions $\pi \in S_N$,

$$e_{\min} \doteq \min\{e_\pi : \pi \in S_N\}. \tag{8}$$

Let $i = C(X)$ be the unique integer in $\{0, 1\}^b$ such that $X \in R_i$ and assume that the source vectors are independent of the channel noise η. Let p denote the probability mass function of the codevectors induced by the source statistics, given by

$$p(y_k) = \Pr[X \in R_k] \tag{9}$$

where Pr denotes the probability of a random event. The number $p(y_k)$ gives the probability that a particular codevector is selected by the encoder to represent an analog input vector. This probability distribution is determined by the input signal vector statistics. The vector probabilities $p(y_k)$ can be computed *a priori* by tabulating statistics on the relative frequency with which each codevector is chosen as the best match.

The binary index transmitted across the channel to represent X is

$$\pi(C(X)) = \pi(i), \tag{10}$$

the received index is $\tau(\pi(i))$, and the index of the codevector used to approximate X in the receiver is $\pi^{-1}(\tau(\pi(i)))$. If we define

$$j \doteq \pi^{-1}(\tau(\pi(i))) = \pi^{-1}(\pi(i) \oplus \eta), \tag{11}$$

then codevector y_i is chosen by the encoder as an approximation of X, and y_j is the decoded codevector selected as an approximation of X. Index i is a random variable depending on the source statistics of X, and index j is a random variable depending on both X and on the channel noise statistics determined by the discrete random variable η, as well as on the deterministic mapping π. The total distortion due to the combined effect of the quantization and channel index errors is $d(X, y_j)$. In order to optimize the performance of the coding system for a given codebook, the expected value of this distortion must be minimized over all possible permutations in S_N. Thus, we take the expectation in (7) by averaging over both the distribution of the input vector X and the distribution of channel bit errors. The quantity to be minimized can be written as

$$e_\pi = E\big[d(X, y_j)\big]. \tag{12}$$

With the mean-square distortion function, an optimal vector quantizer on a noiseless channel satisfies the well-known nearest-neighbor and centroid conditions. In the presence of channel noise, generalized versions of these conditions are satisfied [15]. Totty and Clark [3] showed (later generalized by Messerschmitt [28]) that when channel noise is added to an optimal quantizer, the mean-square error can be separated into the sum of the mean-square error for the quantizer without channel noise and a term due only to channel noise, as given by the following Lemma.

Lemma 1: The average distortion of a noisy channel quantizer with the mean-square distortion function that satisfies the centroid condition can be written as $E\|X - y_j\|^2 = E\|X - y_i\|^2 + E\|y_i - y_j\|^2$.

III. Minimum Distortion Index Assignment

A. Optimal Pseudo-Gray Coding

To minimize (12) requires determining which permutation π minimizes the quantity e_π. In general, this may be quite a difficult task since both the source and channel statistics must be taken into account. There is, however, an asymptotic procedure for determining the best permutation π, the permutation that achieves $e_{\min}$. A *training set* T is a finite collection of vectors,

$$T = \{w_1, w_2, \cdots, w_M\} \tag{13}$$

where $w_k \in \boldsymbol{R}^p$ for $1 \le k \le M$. In practice, a training set is much larger than a codebook, i.e., $|T| \gg |Y|$. The expected system distortion e_π for a given permutation π can be approximated by

$$e_\pi \approx \frac{1}{M}\sum_{k=1}^{M} d(w_k, y_j) \tag{14}$$

where the index j is determined for each w_k by a channel simulation and the given mapping π. For sufficiently large numbers of training vectors, M, the summation in (14) will approach the actual value of e_π. Hence, by approximating the source statistic by those of a large training set of vectors, and by simulating the channel characteristics, there is an effective procedure for determining the value of e_π with any desired accuracy for any given permutation π. By iterating this procedure for each $\pi \in S_N$, one can determine which permutation π is optimal (yields the minimal expected distortion e_π). If d is the mean-square distortion function and the codevectors satisfy the centroid condition, e_π can be computed without a channel simulation, by using Lemma 1.

To determine the optimal codebook permutation in this way, however, is a problem of enormous complexity. Very large training sets must be used to accurately simulate the effects of channel errors, and every permutation in S_N must be checked to determine which is optimal. Since S_N has $N!$ permutations, even a relatively small codebook size (e.g., $N = 16$) presents a formidable challenge. For this reason, a suboptimal solution to the pseudo-Gray coding problem is sought. We concentrate on finding a permutation π which reduces the average distortion e_π by way of a locally optimal algorithm. Attention is restricted to certain classes of commonly used distortion functions.

B. Local Optimum

We consider *metric* distortion functions and *r*th-*power of a norm* distortion functions, the latter being of the form

$$d(X, Y) = \|X - Y\|^r \tag{15}$$

where $\|\cdot\|$ is any norm on $\boldsymbol{R}^p$ and $r \ge 1$. When $r = 2$, we get the important *squared-error distortion function*, using the usual Euclidean norm. It is shown that the average distortion with each distance measure can be upper bounded by quantities that are amenable to minimization via a common descent algorithm.

Case 1 – d is a Metric: By rewriting $d(X, y_j)$ and taking expectations, e_π can be decomposed into the sum of an average distortion of a vector quantizer on a noiseless channel and a term due to channel errors,

$$E\big[d(X, y_j)\big] = E\big[d(X, y_i)\big] + E\big[d(X, y_j) - d(X, y_i)\big]. \tag{16}$$

Note that if the encoder C is a nearest-neighbor encoder, then

$$E[d(X, y_j) - d(X, y_i)] \geq 0 \tag{17}$$

which implies that channel errors on transmitted indexes always increase the average system distortion, as would be expected. The triangle inequality implies

$$e_\pi = E[d(X, y_j)] \leq U_\pi \tag{18}$$

where

$$U_\pi = E[d(X, y_i)] + E[d(y_j, y_i)]. \tag{19}$$

The quantity U_π provides an upper bound for the minimum total expected distortion of the VQ system. This motivates us to seek to *minimize this upper bound* as a method of reducing the expected codebook distortion. This approach does not necessarily guarantee minimization of the total distortion, but is a heuristic solution that can contribute to its reduction. Our experimental results help to justify its use.

The quantity $E[d(X, y_i)]$, the expected distortion due to the approximation of an input vector by a codevector, is independent of the assignment of indexes to codevectors, and depends only upon the original design of the codebook. It is thus a constant with respect to the minimization problem (over all permutations). To minimize the upper bound U_π, it suffices to find

$$\min_{\pi \in S_N} E[d(y_j, y_i)]. \tag{20}$$

Case 2 – d is an rth Power of Norm: Where $r = 1$, d is a metric so that Case 1 applies as well. The Minkowski inequality for expectations can be written as [29]

$$(E\|X + Y\|^r)^{\frac{1}{r}} \leq (E\|X\|^r)^{\frac{1}{r}} + (E\|Y\|^r)^{\frac{1}{r}} \tag{21}$$

where $X, Y \in R^p$. Hence, we have

$$e_\pi = E\|X - y_j\|^r = E\|(X - y_i) + (y_i - y_j)\|^r \leq U'_\pi \tag{22}$$

where

$$U'_\pi = \left\{ (E\|X - y_i\|^r)^{\frac{1}{r}} + (E\|y_i - y_j\|^r)^{\frac{1}{r}} \right\}^r. \tag{23}$$

From (23) it can be seen that e_π is upper bounded by a quantity containing two components. The first is independent of the permutation π, and the second component, containing $d(y_i, y_j)$, varies depending on the choice of π. To minimize the upper bounds in both Case 1 and Case 2 (for a fixed collection of codevectors), it suffices to minimize the quantity

$$D_\pi \doteq E[d(y_i, y_j)] \tag{24}$$

over all permutations in S_N. The subscript π indicates the fact that the value of the expectation depends on π.

The inequality $e_\pi \leq U'_\pi$ is valid over a broad class of quantizers, namely, any quantizer with a distortion measure that is an rth power of a norm and which transmits symbols across a discrete memoryless channel. In the special case when the distortion function is the mean-square criterion and the quantizer is optimal for a noiseless channel, Lemma 1 shows that $e_\pi = U_\pi$, so minimizing D_π over all π actually minimizes e_π as well (instead of an upper bound on e_π).

We examine the expected value D_π in terms of the probability functions of X and η under the assumption of a memoryless binary symmetric channel. The expectation in D_π can be reduced to a very simple formula involving a summation of "costs" of codevectors, where each cost is a readily computable function. This provides the basis for an efficient algorithm that performs the pseudo-Gray minimization in a locally optimal manner. Since the channel noise is assumed independent of the source and $\{R_k\}$ partitions the space R^p, we have

$$D_\pi = \sum_{t \in \{0,1\}^b} \sum_{k=0}^{N-1} E[d(y_i, y_j) \mid \eta = t,\ X \in R_k] \cdot \Pr[\eta = t] \Pr[X \in R_k]. \tag{25}$$

Using the fact that $i = k$ whenever $X \in R_k$,

$$D_\pi = \sum_{k=0}^{N-1} p(y_k) \sum_{t=0}^{N-1} d(y_k, y_{\pi^{-1}(\pi(k) \oplus t)}) \Pr[\eta = t]. \tag{26}$$

Note that by minimizing the upper bound in (19) and (23) rather than the exact average distortion, *we have eliminated any need to consider the explicit distribution of the input vector X in order to find an optimal permutation*. It is sufficient to know the codevector probabilities $p(y_k)$ and the channel statistics.

For each binary index $q \in \{0, 1\}^b$ and each integer m with $0 \leq m \leq b$, define the *m*th-*neighbor set* of q as

$$N^m(q) \doteq \{r \in \{0, 1\}^b : H(q, r) = m\} \tag{27}$$

where $H(\cdot, \cdot)$ is the Hamming distance function. $N^m(q)$ is the set of all integers whose binary representations differ from that of q in exactly m positions (have Hamming distance equal to m). Equivalently, $N^m(q)$ is the set of indexes that index q can be transformed into as the result of exactly m bit errors. If q is a b-bit binary word, then counting the number of different ways that q can be changed in exactly m bit positions gives

$$|N^m(q)| = \binom{b}{m}. \tag{28}$$

For example, if $b = 4$, then the neighbor sets of the index "2" given in decimal notation are $N^0(2) = \{2\}$, $N^1(2) = \{3, 0, 6, 10\}$, $N^2(2) = \{1, 4, 8, 7, 11, 14\}$, $N^3(2) = \{12, 15, 9, 5\}$, and $N^4(2) = \{13\}$.

In a binary symmetric channel (BSC) with error transition probability ϵ,

$$\Pr[\eta = t] = \epsilon^{H(t,0)}(1 - \epsilon)^{b - H(t,0)}. \tag{29}$$

If a channel is binary symmetric, then for any integers $s, t \in \{0, 1\}^b$,

$$\Pr[\eta = s] = \Pr[\eta = t] \quad \text{iff } H(s, 0) = H(t, 0). \tag{30}$$

The probability that a transmitted binary word from $\{0, 1\}^b$ is received as a particular binary word depends only on their Hamming distance apart from each other and for a fixed m, $\Pr[\eta = t]$ is a constant over all $t \in N^m(C(X))$. For each m, with $0 \leq m \leq b$ and $H(t, 0) = m$, define

$$q_m \doteq \Pr[\eta = t] = \epsilon^m(1 - \epsilon)^{b-m}. \tag{31}$$

The second summation in (26) is taken over the set of all 2^b codevectors in Y. This set of vectors can be partitioned into b neighbor sets, such that each set in the partition of Y consists of all vectors whose indexes are a fixed Hamming distance from the index i. Using this observation, the summation over k can be written as two summations:

$$\begin{aligned} D_\pi &= \sum_{k=0}^{N-1} p(y_k) \sum_{m=0}^{b} q_m \sum_{z \in N^m(0)} d(y_k, y_{\pi^{-1}(\pi(k) \oplus z)}) \\ &= \sum_{k=0}^{N-1} \sum_{m=0}^{b} p(y_k) q_m \sum_{w \in N^m(\pi(k))} d(y_k, y_{\pi^{-1}(w)}). \end{aligned} \tag{32}$$

Define the *m*th *cost* of y_k, with respect to the permutation function π as

$$C_\pi^{(m)}(y_k) \doteq p(y_k) \sum_{w \in N^m(\pi(k))} d(y_k, y_{\pi^{-1}(w)}). \tag{33}$$

$C_\pi^{(m)}(y_k)$ measures the relative contribution to the overall expected bit error distortion of the codebook, when exactly m bit errors occur, and y_k is the codevector selected by the encoder.

The above formulation yields a means of minimizing the average distortion upper bounds, U_π and U'_π. Note $m = 0$ implies $N^0(\pi(k)) = \{\pi(k)\}$, so that

$$C_\pi^{(0)}(y_k) = p(y_k)\, d(y_k, y_{\pi^{-1}(\pi(k))}) = 0. \tag{34}$$

Finally, define the *total cost*, $C_\pi(y)$, of a given codevector y, with

respect to the permutation π as

$$C_\pi(y) = \sum_{m=1}^{b} q_m C_\pi^{(m)}(y), \tag{35}$$

which measures the total contribution to bit error distortion due to possible channel errors when a particular codevector, y, is selected by the encoder. In terms of the cost functions, the problem of minimizing D_π reduces to finding a permutation π that minimizes the equivalent quantity

$$D_\pi = \sum_{k=0}^{N-1} C_\pi(y_k), \tag{36}$$

where the minimization ranges over all possible permutation functions in S_N.

We wish to minimize D_π in the hope of reducing the expected distortion introduced to the VQ system from the combined effect of coding error and channel noise. In general, as can be seen in the above equation, this minimization involves the knowledge of the channel's error transition probability ϵ as well as the distance function d and the codevector probabilities $p(y_k)$.

C. Asymptotic Properties

When the channel error probability ϵ is sufficiently small, the probability of multiple bit errors in an index is very small relative to the probability of zero or one bit error. Often it is adequate to consider only the effects of single bit errors on channel words. If one assumes only single bit errors, then to minimize D_π, it suffices to minimize a simplified version of (36) that does not depend on the value of ϵ, and is less computationally demanding.

Neglecting the effect of multiple bit errors assumes that $q_m = 0$ for $m \geq 2$ or, equivalently for every codevector $y \in Y$,

$$C_\pi(y) \approx \epsilon(1-\epsilon)^{b-1} C_\pi^{(1)}(y) \tag{37}$$

Substituting for q_1,

$$D_\pi \approx \epsilon(1-\epsilon)^{b-1} \sum_{k=0}^{N-1} C_\pi^{(1)}(y_k). \tag{38}$$

To minimize D_π in this case, it suffices to find the permutation function π which minimizes the quantity

$$d_\pi \doteq \sum_{k=0}^{N-1} C_\pi^{(1)}(y_k). \tag{39}$$

Since minimizing d_π does not depend upon the value of the parameter ϵ, the solution applies to *any* memoryless binary symmetric channel. This result leads to a tractable algorithm for solving the pseudo-Gray problem in a locally optimal manner. All that need be known *a priori* about the communication channel in this situation is that it is memoryless and binary symmetric in nature. The exact frequency in which errors occur is not significant in determining which codebook permutation function minimizes the derived upper bound.

It is important to examine the accuracy of the distortion upper bounds U_π and U'_π given in (19) and (23). In actual VQ systems, one often encounters channels with relatively low noise levels or vector quantizers with very high resolution. We consider the deviation of U_π and U'_π from e_π in these situations in limiting cases. As the channel becomes less noisy or as the resolution of the quantizers increases, the upper bounds approach equality. For a given source, and either a fixed channel or a fixed noiseless quantizer, there always exist noisy channel vector quantizers that make the upper bounds U_π and U'_π arbitrarily accurate. In this uniform sense, the bounds are as tight as possible. The following lemma summarizes this property.

Lemma 2: In the limit of high resolution or low channel noise, each of the bounds U_π and U'_π tends to e_π, independent of the permutation π.

The results of the previous sections can be generalized and applied to arbitrary error correcting codes. In general, if more errors occur than a code is capable of handling, invalid data are received (assuming retransmissions are unallowable). If, however, a given channel code is used to code vector quantization indexes, additional performance can be achieved. Beyond a code's error correcting capability, incorrect codevector indexes are received and decoded as erroneous codevectors. The expectation of the resulting vector distortion can be reduced by *a priori* carefully assigning channel words to codevector indexes. Pseudo-Gray coding can "extend" the error correcting capability of a given channel code in the sense that, in addition to correcting all errors less than a certain amount, it can reduce the average error that a VQ systems experiences when more errors occur than the channel code can handle. Applications of pseudo-Gray coding to error correcting codes will be presented in a future publication.

IV. Binary Switching Algorithm

A. Algorithm Description

In this section, a Binary Switching Algorithm is presented that performs pseudo-Gray coding on a given VQ codebook. A description of a channel (such as the error probability ϵ on a BSC) and a predesigned VQ codebook are given as input. The algorithm eventually halts and gives as output an assignment of binary indexes (i.e., a mapping π) to the vectors of the codebook. The new index assignment provides an improvement in the average distortion due to channel errors, over an arbitrary index assignment, as confirmed by experimental results (Section V).

At this point, a simple fact is worth noting. For any index mapping π, the selection of codevector y_i as an approximation to some input vector X causes the transmission of index $\pi(i)$. This is conceptually equivalent to first shuffling the vectors in a codebook, so that for each index i, vector y_i moves to location $\pi(i)$, and then using the identity mapping for π.

An interesting question naturally arises with an algorithm that produces as its output a permutation π. What data representation should be used to specify π? The easiest way to specify π apparently is to produce the original codebook, but with the codevector locations shuffled in the described manner. This has the added advantage that no memory is needed to specify π other than the storage required for the codebook itself. The algorithm thus receives a codebook as input, and outputs the same codebook but with its vectors in different locations.

The Binary Switching Algorithm rearranges a codebook such that the summation in (36) is locally a minimum. The main idea involves iteratively switching the positions of two codevectors to reduce the term D_π after every switch. A monotonic decrease in D_π results as the algorithm progresses. Each such switch constitutes a change in the permutation π. The choice of which pair of vectors to switch in the codebook at each iteration is determined by a heuristic ordering process, which works well in practice. Each codevector y is assigned a cost, $C_\pi(y)$, as given in (35). The codevectors are sorted in decreasing order of their cost values. The vector with the largest cost, say, y_0, is selected as a candidate to be switched first. A trial is conducted, where y_0 is temporarily switched with each of the other codevectors to determine the potential decrease in D_π following each switch. The codevector which yields the greatest decrease in D_π when switched with y_0 is then switched with y_0. If no such vector exists (an unsuccessful switch attempt), then the second highest cost vector is used to check for the most cost-efficient switch, and so on. If every codevector is such that when switched with every other codevector, no decrease in D_π results, then the algorithm halts in a locally optimal state.

By ordering the codevectors based upon a measure of the quality of their codebook positions (costs), the algorithm is given a sense of direction in terms of which vectors are important to switch first. The vectors with the highest costs contribute largely to the expected VQ distortion, which is equal to the sum of these costs. If the total number of program iterations is limited because of running time constraints, then this directedness becomes increasingly significant.

The completion of each switch or unsuccessful switch attempt constitutes the end of an iteration. Following such a switch, the process repeats; the vectors are resorted based on cost, and new switches are attempted.

It is important to point out that from any initial codebook index assignment, any other index assignment can be "reached" by performing vector switches entirely of the type described above. This follows trivially from the well-known result in group theory that any element of the symmetric group can be decomposed into a product of 2-cycles. This fact guarantees that the restriction to vector switches does not exclude the globally optimal index assignment as a potential final permutation of the BSA.

B. Initializing a Codebook Permutation

The input to the Binary Switching Algorithm is a codebook with a particular initial index assignment. The initial index assignment (i.e., initial vector locations) can affect the performance of the pseudo-Gray algorithm, since only a local optimum is determined. There are many different ways to specify an initial index assignment.

The simplest initial assignment is an arbitrary one. That is, take the codebook and input it directly to the pseudo-Gray algorithm following codebook design, with the vectors in whatever order they appear. In this case, the entire process of minimizing D_π consists only of the vector switching operations described.

With the use of preprocessing, an initial index assignment can be constructed in an attempt to reduce the quantity D_π when the algorithm reaches a locally optimal state. Such techniques are generally heuristic in nature. One such preprocessing technique involves examining the codevectors that have the highest probabilities of being used [i.e., the largest $p(y_k)$]. Since these vectors are used more frequently than others in the codebook, they can initially be assigned to have Hamming neighbors that are very close in terms of the distortion function d. This can be achieved by distributing several of the highest probability codevectors throughout the codebook and placing close vectors as their nearest neighbors. In this way, the process of minimizing D_π is given a "head start," based on an intuitive notion of what an optimal codebook permutation should look like.

Another option for codebook initialization is one that makes use of random codebook permutations. Initially, a predetermined number of random index assignments is performed. The assignment with the lowest distortion, D_π, is then chosen as the starting codebook for the vector switching algorithm.

C. Halting Conditions

There are several options available to determine when the pseudo-Gray algorithm will halt. As noted previously, the default is for the algorithm to keep running until a local optimum is reached. In this situation, if any two codevectors are switched in position, then no decrease in D_π can result. This stopping condition can yield extremely long running times for large codebooks. Of course, since running the algorithm on a codebook is a one-time operation, this may be tolerable.

There is no known reasonable bound on the number of iterations required before the algorithm halts. A weak upper bound on running time can be obtained by considering the maximum number of vector switches possible. Each vector switch yields a strictly positive decrease in D_π, and hence a codebook with an index assignment mapping π that is different from any codebook permutation that the algorithm had previously generated (otherwise, two different distortions would result from the same permutation π). Since there are at most $N!$ distinct codebook permutations, there can be at most $N!$ vector switches before the algorithm halts. In practice, this upper bound is normally not closely approached, as can be seen in the tabulation of execution times (Table II) for various trials of the algorithm.

An alternative to stopping when no more vector switches are possible is to perform some random perturbations from the locally optimal state, attempting to find a decrease in D_π. If such a new codebook permutation is found, then the entire algorithm can be used again to locally optimize this new codebook. With this method, one continually reaches locally optimal states followed by randomly jumping into new states with less distortion.

If computation time is critical, an upper bound can be set on the number of iterations to perform in the program. A program variable $I_{\max}$ is provided to specify the maximum number of iterations allowed before the program should halt. In this situation, the program keeps switching codevectors until either a local optimum is reached or the designated number of iterations is performed.

Binary Switching Algorithm Parameters

Input Parameters:

Y	Input codebook $\{y_i: 0 \le i \le N-1\}$
$p(y_i)$	Vector probabilities $(0 \le i \le N-1)$.
d	Distortion function.
ϵ	Channel error probability (for a BSC).
$I_{\max}$	Number of iterations to perform before halting. Default is ∞ (runs until a local optimum is reached).

Output Parameters:

Y'	Pseudo-Gray coded codebook (permuted version of Y).
$D_{\pi i}$	Initial value of distortion in Y. π_i is the initial permutation.
$D_{\pi f}$	Final value of distortion in Y. π_f is the final permutation

Procedures Used in the Algorithm:

INITIALIZE_CODEBOOK(): Preprocess codebook Y before running the vector switching algorithm on it.

SORT_INDEXES(): Construct an array $A(n)$ of the indexes of codevectors sorted in order of decreasing *Cost*.

SWITCH(i, j): Switch the index assignments of the codevectors y_i and y_j in Y.

UPDATE_COST(j): Calculate and return the quantity $C_\pi(y_i)$ using the vector probabilities $p(y)$.

DISTORTION(): Calculate and return the quantity D_π for codebook Y.

Binary Switching Algorithm

```
BEGIN: N := 2^b
    INITIALIZE_CODEBOOK()
    I := 1                    ;;;I = iteration number
    I_max := ∞
    i := 0
LOOP: FOR k := 0 TO N - 1
        UPDATE_COST(k)
    NEXT k
    SORT_INDEXES()            ;;; A is an array of indexes
    δ* := 0
    FOR j := 1 TO N - 1
        IF j = A(i) THEN NEXT j
        D_π := DISTORTION()       ;;;Current permutation is π
        SWITCH (A(i), j)          ;;;Create a new permutation π'
        D_π' := DISTORTION ()
        δ := D_π' - D_π
        IF δ ≥ 0 THEN GOTO UNSWITCH
        IF |δ| > δ* THEN
            δ* := |δ|
            j* := j               ;;;j* is the Bestindex.
UNSWITCH: SWITCH (A(i), j)
    NEXT j
    IF δ* = 0 THEN
        IF i = N - 1 THEN STOP
        ELSE i := i + 1: GOTO LOOP
    SWITCH (A(i), j)
    i := 0
    I := I + 1
    IF I > I_max THEN STOP
    GOTO LOOP
```

TABLE I
COMPUTATIONAL TIME COMPLEXITIES PER ITERATION. N = CODEBOOK SIZE

	General Case		Single Bit Errors	
	Initialization	Updates	Initialization	Updates
SORT_INDEXES	$O(N \log N)$	$O(N \log N)$	$O(N \log N)$	$O(N)$
D_π, δ	$O(N^2)$	$O(N)$	$O(N \log N)$	$O(\log N)$
Total	$O(N^2)$	$O(N^3)$	$O(N \log N)$	$O(N^2 \log N)$

D. Computational Complexity

Since the optimization problem at hand involves summations that range over the set of all possible codebook permutations, it is important to consider the computational complexity of such a task. An optimal solution can be found by way of exhaustively searching the $N!$ possible configurations of the codebook, which for even small size codebooks can be prohibitive. A codebook of size 32 has approximately 10^{35} different configurations, making an exhaustive search essentially impossible.

The Binary Switching Algorithm provides a reduced complexity method for obtaining good codebook permutations. The computationally demanding portions of the algorithms are the execution of the procedure SORT_INDEXES and the calculation of the variable δ. After the first iteration, many of the computational requirements can be reduced by updating various quantities, rather than recomputing them in their entirety. The algorithm's complexity requirements can be divided between those of the first iteration (which can be interpreted as a startup overhead cost) and those of subsequent iterations. By making the single bit error assumption, the algorithm's complexity can be reduced. The complexities of the more efficient single bit error case and the general case are analyzed in this section and are given as functions of N, the codebook size.

The procedure SORT_INDEXES sorts the N indexes in $\{0,1\}^b$ such that the corresponding codevectors in Y are in decreasing order of costs, C_π. Assuming the costs are precomputed, the sort can be done in average time $O(N \log N)$ using a standard sorting algorithm [30]. Following the initial sort, a more efficient sort update can be implemented when only single bit errors are assumed. Switching two vectors affects the values of $C_\pi(y_k)$ for at most $2b$ [i.e., $O(\log N)$] codevectors, since the number of vectors with Hamming distance one from either of the two switched vectors is at most $2b$. Sorting the indexes in this case reduces to the problem of resorting $O(N)$ indexes, given that $O(\log N)$ of them have been shuffled. To do this, one can first resort the $O(\log N)$ shuffled indexes in average time $O(\log N \cdot \log \log N)$, and then merge them with the remaining indexes in time $O(N)$.

The complexity of initially computing various quantities needed by the algorithm turns out to be more demanding than the complexity of subsequent iterations. By examining the number of terms in the summation in (33), it can be seen that the complexity of computing the term $C_\pi^{(m)}(y_k)$ for a fixed integer m and a fixed codevector y_k is

$$O(|N^m(\pi(k))|) = O\left(\binom{b}{m}\right). \tag{40}$$

Using (35), the complexity of computing $C_\pi(y_k)$ for a fixed codevector y_k is thus

$$O\left(\sum_{m=1}^{b}\binom{b}{m}\right) = O(2^b) = O(N), \tag{41}$$

and the complexity in the single bit error case is

$$O\left(\binom{b}{1}\right) = O(\log N). \tag{42}$$

From (36), the complexity of initially computing D_π is $O(N^2)$ since $C_\pi(y_k)$ must be computed N times. With single bit errors, it suffices to compute d_π instead of D_π. The complexity of initially $\{0,1\}^b$ such that $\pi(r) = \pi'(s)$, $\pi(s) = \pi'(r)$, and $\pi(l) = \pi'(l)$ for every $l \in \{0,1\}^b - \{r,s\}$. To calculate the value $d_{\pi'}$, the computing d_π is

$$O(N \cdot |N^{(1)}(\pi(k))|) = O\left(N \cdot \binom{b}{1}\right) = O(N \log N). \tag{43}$$

For any iteration other than the initial one, let π' be the permutation that results when two vectors are switched in location from the previous permutation π. The value of $D_{\pi'}$ (respectively, $d_{\pi'}$) can be obtained by modifying the value of D_π (respectively, d_π) according to the changes induced by a single vector switch. The relationship between π and π' is that there exist 2 indexes $r, s \in$ relation $d_{\pi'} = (d_{\pi'} - d_\pi) + d_\pi$ and the fact that d_π is known give

$$\delta = d_{\pi'} - d_\pi = \sum_{k=0}^{N-1}\left[C_{\pi'}^{(1)}(y_k) - C_\pi^{(1)}(y_k)\right]. \tag{44}$$

The only terms in the above summation that can be nonzero are those for which $\pi(k)$ is not equal to $\pi(r)$, $\pi(s)$, or any element in the neighbor sets $N^1(\pi(r))$ and $N^1(\pi(s))$. Hence, if $C_\pi^{(1)}(y_k) \neq C_{\pi'}^{(1)}(y_k)$, then $\pi(k) \in A$, where $A = \{\pi(r), \pi(s)\} \cup N^1(\pi(r)) \cup N^1(\pi(s))$. To compute δ, it suffices to sum over all k in the set $\pi^{-1}(A)$, giving

$$\begin{aligned} d_{\pi'} - d_\pi &= \sum_{k \in \pi^{-1}(A)}\left[C_{\pi'}^{(1)}(y_k) - C_\pi^{(1)}(y_k)\right] \\ &= \left[C_{\pi'}^{(1)}(y_r) - C_\pi^{(1)}(y_r)\right] + \left[C_{\pi'}^{(1)}(y_s) - C_\pi^{(1)}(y_s)\right] \\ &\quad + \sum_{k \in \pi^{-1}(A) - \{s,r\}}\left[C_{\pi'}^{(1)}(y_k) - C_\pi^{(1)}(y_k)\right]. \end{aligned}$$

Each of the first two terms above can be computed in time $O(\log N)$. Each difference of costs in the summation of the third term above can be computed in time constant in N, since only one term in (35) (taking $m = 1$) needs to be updated for each term. Thus, since $|A| \leq 1 + 1 + b + b = O(\log N)$, the quantity $d_{\pi'} - d\pi$ can be computed in time $O(\log N + \log N + |A|) = O(\log N)$.

Omitting the single bit error assumption, each vector switch has an effect on the cost of every codevector. In general, for two codevectors switched, y_r and y_s, the terms $C_\pi(y_r)$ and $C_\pi(y_s)$ must be entirely recalculated in time $O(N)$, while for each remaining codevector y, the quantity $C_\pi(y)$ can be computed by updating at most two of its mth costs, $C_\pi^{(m)}(y)$. Furthermore, for each of these two mth costs, at most two components of the sum in definition of $C_\pi^{(m)}(y)$ need to be adjusted, so recomputing $C_\pi^{(m)}(y)$ can be done in constant time. Computing $D_{\pi'} - D_\pi$ can therefore be done in time $O(2N + (N-2) \cdot 2) = O(N)$.

During each iteration of the algorithm, δ must be computed for each trial pair of vectors switched, before determining which vector pair provides the most gain. The number of such computations of δ is bounded by the total number of possible pairs of codevectors, which is $O(N^2)$. The total computational complexity of an iteration is then of order N^2 times the complexity of computing δ. These are summarized in Table I.

Although the order of magnitude of run times for the BSA can be estimated by complexity analysis, often the actual run times must be known. The BSA typically required about 25 h of CPU time on a 2 MIPS machine (SUN-3/180) for a codebook of size 256, and about 13 h for size 128, each with the single bit error assumption. There may be interest in using the BSA for even larger codebooks and

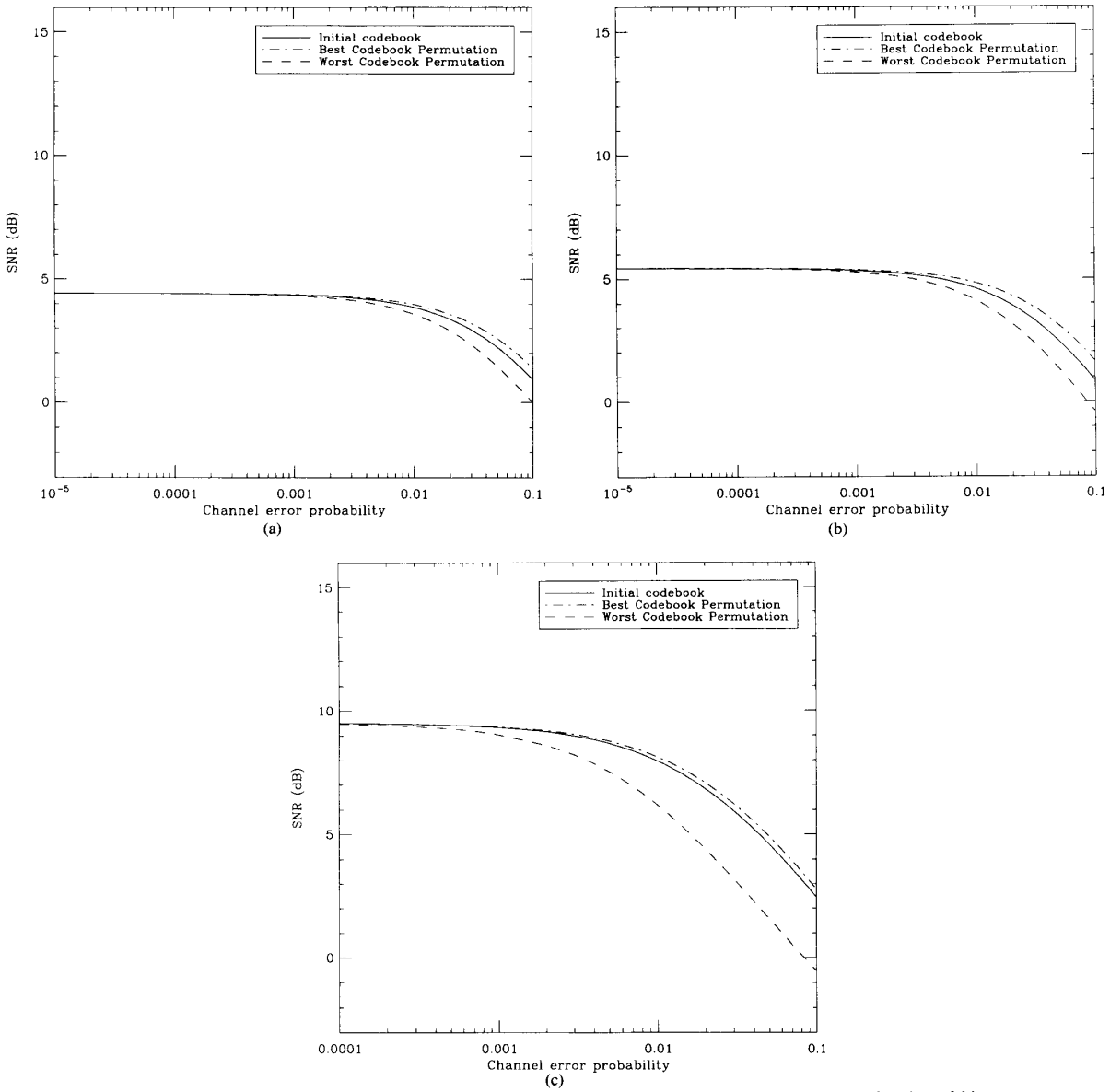


Fig. 3. Performance of vector quantizer with index permutation on a Binary Symmetric Channel as a function of bit error probability. (a) Codebook size = 16, vector dimension = 4, source = Gaussian iid. (b) Codebook size = 16, vector dimension = 4, source = 1st-order Gauss-Markov ($\alpha = 0.5$). (c) Codebook size = 16, vector dimension = 4, source = 1st-order Gauss-Markov ($\alpha = 0.9$).

omitting the single bit error assumption, in which case greater run times can be expected, although more efficient programming and faster computers may ease the burden somewhat.

V. Experimental Results

The Binary Switching Algorithm was implemented in software and tested on various inputs. It was allowed to run until the distortion reached a local optimum. A Euclidean mean-square distortion measure was used as the fidelity criterion (avoiding a channel simulation), and codebooks were designed using a standard Generalized Lloyd Algorithm [31] with training ratios of at least 50. The resulting codebooks with arbitrary initial index assignments were used as input to the BSA, which yielded as output the same codebook but in a different ordering.

The input signals tested include speech waveforms, 1st-order Gauss-Markov processes, and Gaussian i.i.d. signals. The speech waveforms were sampled at 8 kHz, digitized, and partitioned into input vectors of dimension 4 with codebooks of size 64, 128, or 256. The Gauss-Markov processes were of the form

$$x_n = \alpha x_{n-1} + w_n \tag{45}$$

where w_n is a zero-mean, unit variance, Gaussian white noise process, with $\alpha = 0.9$, $\alpha = 0.5$, or $\alpha = 0$ (Gaussian i.i.d.). Each of these was tested with quantizers having the following codebook size and dimension values (N, p): $(16, 4)$, $(64, 6)$, $(256, 4)$.

In order to reduce the running time of the algorithm, the single bit error assumption was used. Designing the codebook permutations for single bit errors proved satisfactory even when multiple bit

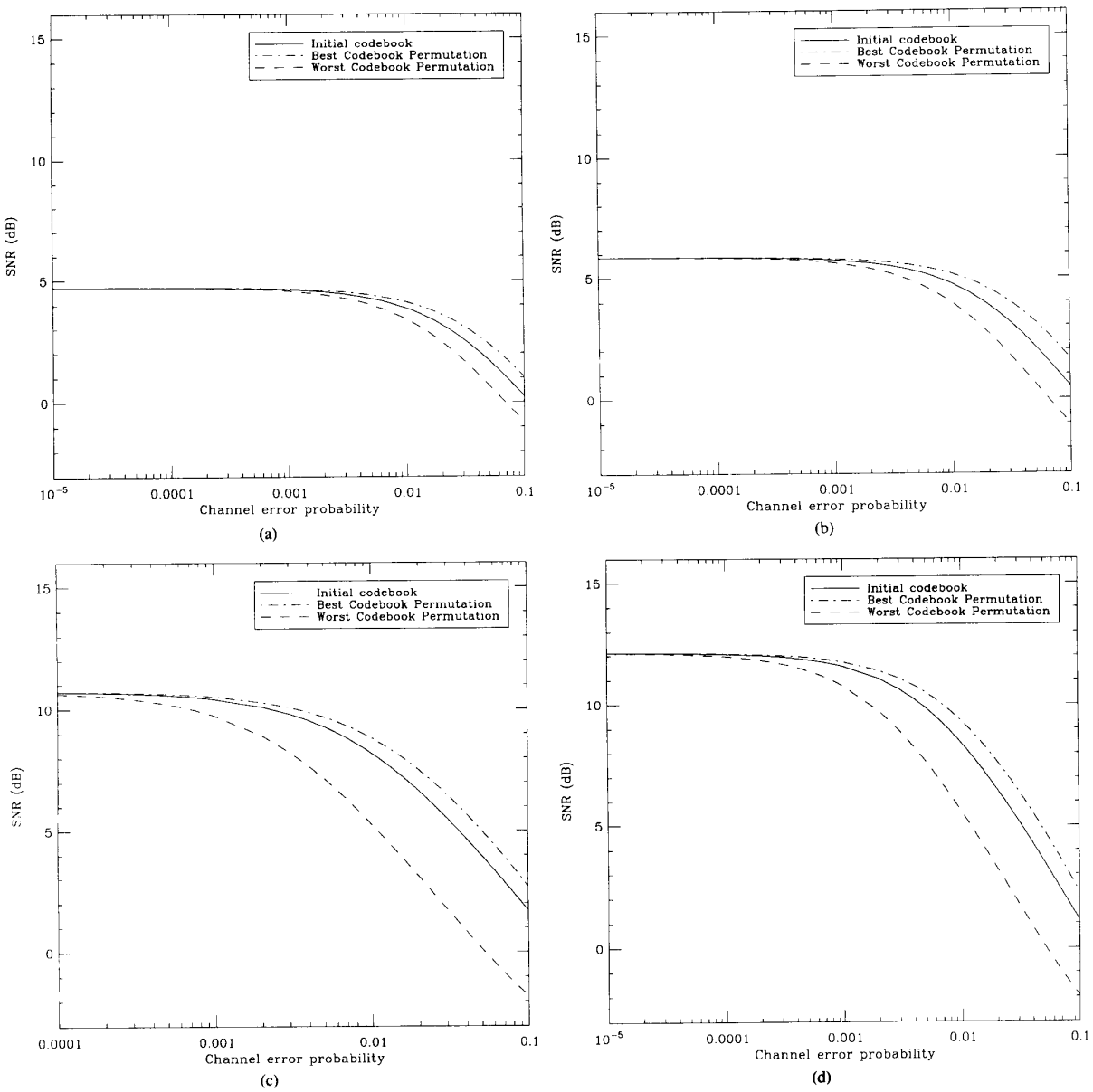


Fig. 4. Performance of vector quantizer with index permutation on a Binary Symmetric Channel as a function of bit error probability. (a) Codebook size = 64, vector dimension = 6, source = Gaussian iid. (b) Codebook size = 64, vector dimension = 6, source = 1st-order Gauss-Markov (α = 0.5). (c) Codebook size = 64, vector dimension = 6, source = 1st-order Gauss-Markov (α = 0.9). (d) Codebook size = 64, vector dimension = 6, source = speech.

errors were allowed, since the SNR improvements provided by the BSA for single bit errors closely follow those for multiple errors, as can be seen by comparing Fig. 5(c) and (d).

For each codebook tested, a computation was made at various channel noise levels of the system's signal-to-noise ratio (SNR), defined as $10\log_{10}(\sigma_x^2/\sigma_n^2)$, where σ_x^2 and σ_n^2 are the signal and noise variances, respectively. The SNR's were computed for the codebooks both before and after applying the BSA for BSC bit error probabilities ranging from 10^{-4} to 10^{-1}.

In addition, by making a minor alteration to the BSA, it is possible to produce rearrnaged codebooks with locally *maximum* average distortion D_π. The previous experiments were repeated for these codebooks with locally "bad" index assignments. The importance of bad codebook permutations is that they provide upper bounds on the expected distortion that would result from using random codebook permutations in a noisy channel VQ system. The signal-to-noise ratios of the "worst case" and "best case" codebooks produced by pseudo-Gray coding provide a range of possible performancces for a typically chosen codebook that has not been pseudo-Gray coded. Of course, neither are truly "worst" nor "best" cases because the algorithm is not guaranteed to achieve a global optimization. The performance of the BSA can be measured either by the dB gain in SNR of a best case codebook over the initial codebook or over a worst case codebook. The various SNR's are plotted in Figs. 3-5.

The increase in SNR from the initial codebook to a "best case"

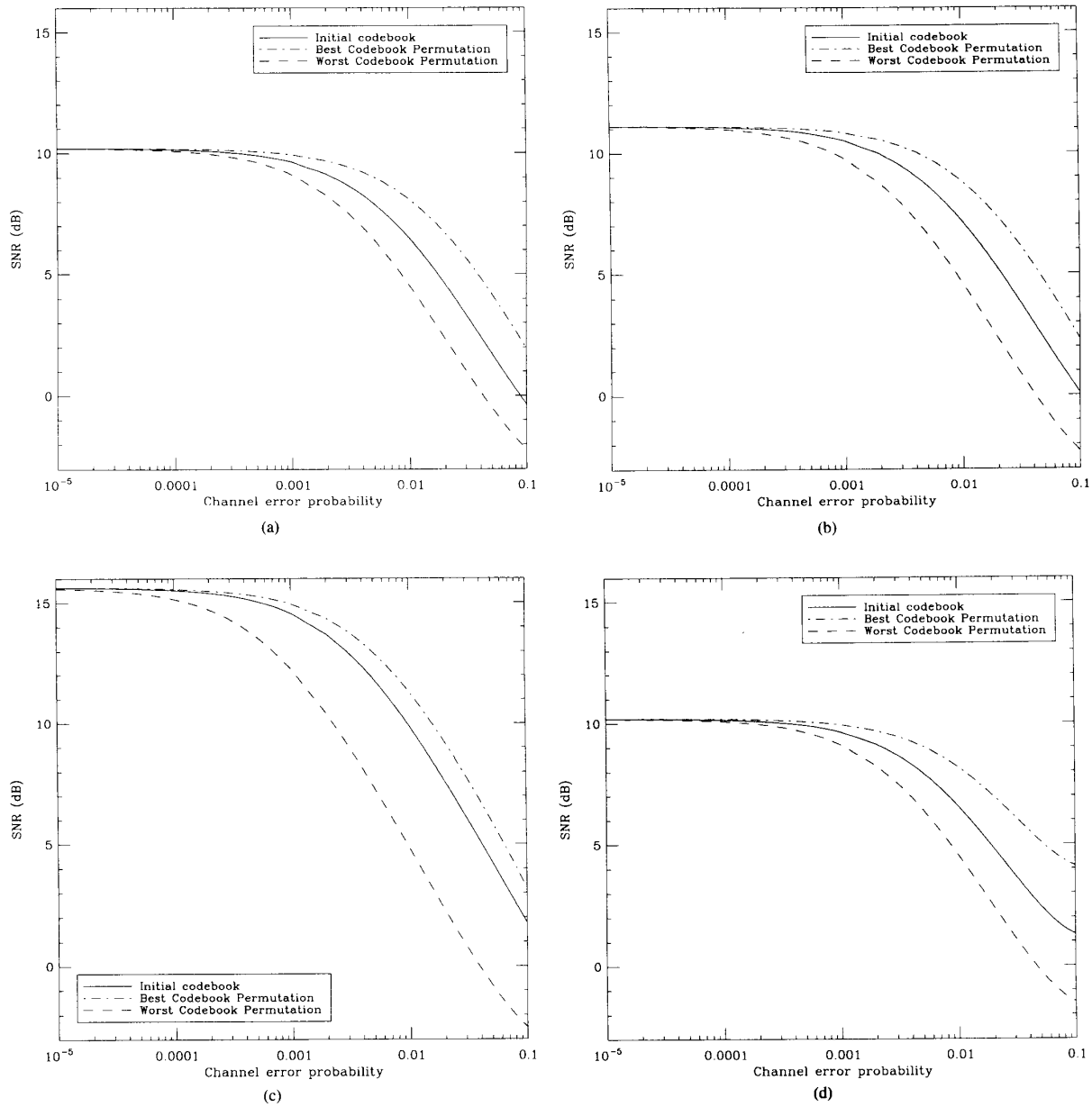
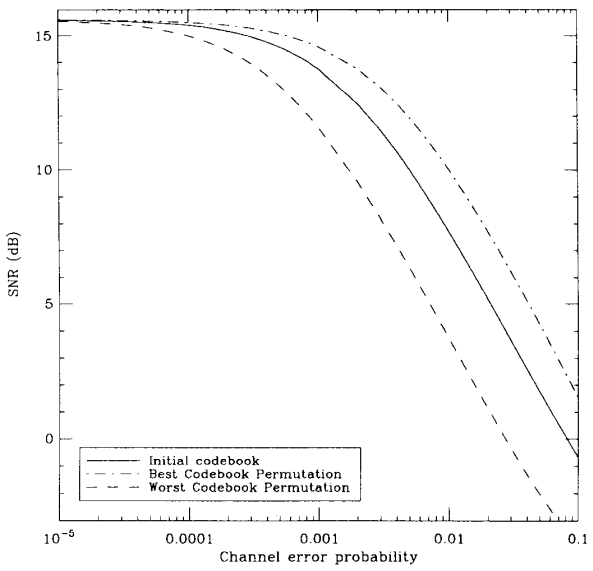


Fig. 5. Performance of vector quantizer with index permutation on a Binary Symmetric Channel as a function of bit error probability. (a) Codebook size = 256, vector dimension = 4, source = Gaussian iid. (b) Codebook size = 256, vector dimension = 4, source = 1st-order Gauss-Markov ($\alpha = 0.5$). (c) Codebook size = 256, vector dimension = 4, source = 1st-order Gauss-Markov ($\alpha = 0.9$). (d) Assuming only single bit errors. Codebook size = 256, vector dimension = 4, source = Gaussian iid. (e) Codebook size = 256, vector dimension = 4, source = speech.

codebook was observed to generally be less than 2.5 dB over the range of channel error probability ϵ between 0 and 0.1, while the increase in SNR from the "worst case" codebook to a "best case" codebook reached as high as 6 dB over this range. The increase in SNR achievable using pseudo-Gray coding is more prominent for higher channel error probabilities.

The SNR's of the codebooks processed by the BSA are often substantial increases over both the starting codebook SNR's and those in the worst cases. The improvements tended to rise with increasing codebook size. This may be due in part to the fact that in larger size codebooks, there is a greater "flexibility" to rearrange vectors, since there are many more locations available, and since the total number of possible vector switches at any one time for a codebook of size N grows as $O(N^2)$.

An interesting fact that should be noted is that at relatively high channel noise levels, the performances of vector quantizers tested with the "best" index assignments can decrease as the vector dimension increases. For example, at a fixed rate of 1 b/source sample, the SNR of a vector quantizer for a Gaussian i.i.d. source decreases from 1.4 dB to 1.1 dB for a bit error probability of 0.1



(e)

Fig. 5. (*Continued*).

TABLE II
SIMULATION RESULTS OF BINARY SWITCHING ALGORITHM (dim = 4)

	Best Codebook		Worst Codebook	
CB Size	Number of Iterations	Number of Switches	Number of Iterations	Number of Switches
16	37	9	59	15
32	91	24	277	66
64	1571	85	6342	610
128	8945	273	46011	1982
256	58312	987	297410	6002

[see Figs. 3(a) and 4(a)]. This performance is consistent with that reported in [16], and is due to the fact that the quantizers are not "channel optimized," since their encoders and decoders were designed under noiseless channel assumptions. In contrast, truly optimal quantizers on noisy channels would be expected to increase in performance as the vector dimension increases.

Some important measurable parameters can yield insight into the success of the algorithm. The number of iterations executed by the algorithm before halting and the number of vector switches carried out during the program's operation were recorded for various codebook sizes and are shown in Table II. The number of iterations is directly proportional (ignoring the minimal initialization overhead) to the actual running time of the program, and generally increased with codebook size. It was observed while running the BSA that vector switches occurred much more frequently at the beginning of the program than toward the end (when approaching a final permutation). This artifact was likely a result of the ordering imposed on the codevectors before allowing vector switches. Similarly, the number of iterations executed remained constant in order of magnitude for codebooks of the same size.

VI. Concluding Remarks

Using the BSA for VQ index assignment can lead to increased performance in the presence of channel noise. Although optimal index assignment is a computationally demanding feat, the BSA provides efficient means of obtaining locally optimal solutions. We note that pseudo-Gray coding can be extended to systems with error correction coding—a topic of a future publication.

Several questions remain concerning VQ design for noisy channels. An interesting one concerns the performance of high-resolution quantizers when channel noise is added. More precisely, suppose a sequence of quantizers Q_n is given where the nth quantizer has n output points and such that $\lim_{n\to\infty} E[d(X, Q_n(X))] = 0$. If a fixed level of channel noise (e.g., BSC) is added, must it still be the case that this limit equals zero? Another problem of great interest is to provide tight mathematical bounds on the potential performance gain that pseudo-Gray coding can provide for a given source and quantizer. Analytic estimates on the running time of the BSA would also be of value.

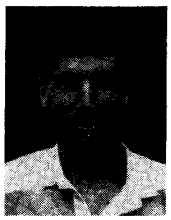
Studies are in progress to utilize pseudo-Gray coding in more complex coding structures, such as in vector-excited and adaptive-predictive coding schemes and with error correction coding. Integrating pseudo-Gray coding into systems specifically designed for noisy channels should become an important task of the future.

References

[1] A. Gersho and V. Cuperman, "Vector quantization: A pattern matching technique for speech coding," *IEEE Commun. Mag.*, vol. 21, pp. 15-21, Dec. 1983.
[2] I. Dostis, "The effect of digital errors on PCM transmission of compandored speech," *Bell Syst. Tech. J.*, vol. 44, pp. 2227-2243, Dec. 1965.
[3] R. E. Totty and G. C. Clark, "Reconstruction error in waveform transmission," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 336-338, Apr. 1967.
[4] N. Rydbeck and C. W. Sundberg, "Analysis of digital errors in nonlinear PCM," *IEEE Trans. Commun.*, vol. COM-24, pp. 59-65, Jan. 1976.
[5] P. Noll, "Effects of channel errors on the signal-to-noise performance of speech encoding systems," *Bell Syst. Tech. J.*, vol. 54, pp. 1615-1636, Nov. 1975.
[6] R. Zelinski, "Effects of transmission errors on the mean-squared error performance of transform coding systems," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 531-537, Oct. 1979.
[7] G. Wolf and G. Redinbo, "The optimum mean-square estimate for decoding binary block codes," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 344-351, May 1974.
[8] G. Redinbo, "Optimum symbol-by-symbol mean-square error channel coding," *IEEE Trans. Inform. Theory*, vol. IT-25, pp. 387-392, July 1979.
[9] ——, "On the design of mean-square error channel coding systems using cyclic codes," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 406-413, May 1982.
[10] J. Modestino and D. Daut, "Combined source-channel coding of images," *IEEE Trans. Commun.*, vol. COM-27, pp. 1644-1659, Nov. 1979.
[11] J. Modestino, D. Daut, and A. Vickers, "Combined source-channel coding of images using the block cosine transform," *IEEE Trans. Commun.*, vol. COM-29, pp. 1261-1274, Sept. 1981.
[12] J. Dunham and R. Gray, "Joint source and channel trellis encoding," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 516-519, July 1981.
[13] A. Kurtenbach and P. Wintz, "Quantization for noisy channels," *IEEE Trans. COmmun.*, vol. COM-17, pp. 291-302, Apr. 1969.
[14] D. Goodman and C. Sundberg, "Combined source and channel coding for variable-bit-rate speech transmission," *Bell Syst. Tech. J.*, vol. 62, pp. 2017-2036, Sept. 1983.
[15] K. Zeger and A. Gersho, "Vector quantization design for memoryless noisy channels," in *Proc. IEEE Int. Conf. Commun.*, Philadelphia, PA, June 1988.
[16] N. Farvardin, "A study of vector quantization for noisy channels," *IEEE Trans. Inform. Theory*, submitted for publication.
[17] N. Farvardin and V. Vaishampayan, "Optimal quantizer design for noisy channels: An approach to combined source-channel coding," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 827-838, Nov. 1987.
[18] R. V. Cox, J. Hagenauer, N. Seshadri, and C.-E. Sundberg, "A sub-band coder designed for combined source and channel coding," in *Proc. Int. Conf. Acoust., Speech, Signal Processing*, New York, 1988, pp. 235-238.
[19] W. Wong and R. Steele, "Partial correction of transmission errors in Walsh transform image without reference to error correction coding," *Electron. Lett.*, vol. 14, pp. 298-300, May 1978.

[20] R. Steele, D. Esdale, and D. Goodman, "Partial correction of transmission errors in DPCM without recourse to error correction coding," *Electron. Lett.*, vol. 13, pp. 351-353, June 1977.
[21] J. H. Chen, G. Davidson, A. Gersho, and K. Zeger, "Speech coding for the mobile satellite experiment," in *Proc. IEEE Int. Conf. Commun.*, Seattle, WA, June 1987.
[22] K. Zeger and A. Gersho, "Zero redundancy channel coding in vector quantisation," *Electron. Lett.*, vol. 23, no. 12, pp. 654-656, June 1987.
[23] J. De Marca and N. Jayant, "An algorithm for assigning binary indices to the codevectors of a multi-dimensional quantizer," in *Proc. IEEE Int. Conf. Commun.*, Seattle, WA, June 1987.
[24] N.-T. Cheng and N. G. Kingsbury, "Robust zero-redundancy vector quantization for noisy channels," in *Proc. IEEE Int. Conf. Commun.*, Boston, MA, June 1989.
[25] J. R. B. De Marca, N. Farvardin, and Y. Shoham, "Robust vector quantization for noisy channels," in *Proc. Mobile Satellite Conf.*, May 1988, pp. 515-520, JPL Publ. 88-9.
[26] D. J. Goodman and T. J. Moulsley, "Using simulated annealing to design transmission codes for analogue sources," *Electron. Lett.*, vol. 24, no. 10, pp. 617-618, May 1988.
[27] N. Farvardin, "Optimal binary code word assignment for vector quantization over a noisy channel—An application of simulated annealing," presented at the *IEEE Int. Symp. Inform. Theory*, Kobe, Japan, June 1988.
[28] D. G. Messerschmitt, "Accumulation of distortion in tandem communication links," *IEEE Trans. Inform. Theory*, vol. IT-25, pp. 692-698, Nov. 1979.
[29] B. R. Bhat, *Modern Probability Theory*. New Delhi, India: Wiley Eastern Limited, 1981.
[30] D. E. Knuth, *The Art of Computer Programming* Vol. 3. Reading, MA: Addison-Wesley, 1973.
[31] 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.

★



Kenneth Zeger was born in Boston, MA, on August 18, 1963. He received both the S.B. and S.M. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology (M.I.T.) in 1984, and both the Ph.D. degree in electrical and computer engineering and the M.A. degree in pure mathematics at the University of California, Santa Barbara (UCSB) in 1990.

During 1980-1981 he worked on adaptive antenna array designs for Z-A Inc. in Glenside, PA. He has worked on real-time speech recognition for Hewlett-Packard Co. at the General Systems Division in Sunnyvale, CA, and on speech compression techniques at HP Laboratories in Palo Alto, CA, during the years 1982-1985. In 1984 he served as a consultant to Automatic Data Processing Co. on digital network design. In July 1990 he joined the Electrical Engineering Faculty at the University of Hawaii at Manoa as an Assistant Professor. His present research interests include combined source/channel coding, speech and image compression, and computational complexity theory.

Dr. Zeger was awarded a four-year Faculty Development Graduate Fellowship by the American Electronics Association in 1985, and was the recipient of a University of California Regents Fellowship in 1989. He is a member of the Communication Theory Technical Committee of the IEEE Communications Society, and is Co-Chairman of the IEEE Workshop on Communication Theory, held June 1990 in Ojai, CA.

★

Allen Gersho (S'58-M'64-SM'78-F'82) is Professor of Electrical and Computer Engineering at the University of California, Santa Barbara (UCSB), and Director of the Center for Information Processing Research at UCSB. He received the B.S. degree from M.I.T. in 1960 and the Ph.D. degree from Cornell University in 1963.

He was at Bell Laboratories from 1963 to 1980. His current research activities are in compression of speech, audio, images, and video signals. He holds patents on speech coding, quantization, adaptive equalization, digital filtering, and modulation and coding for voiceband data modems. He served as a member of the Board of Governors of the IEEE Communications Society from 1982 to 1985, and is a member of the Communication Theory Technical Committee and the Signal Processing and Communications Electronics Technical Committee of the IEEE Communications Society. He has served as Editor of IEEE Communications Magazine and Associate Editor of the IEEE TRANSACTIONS ON COMMUNICATIONS.

Dr. Gersho was awarded the Guillemin-Cauer Prize Paper Award from the Circuits and Systems Society in 1980. In 1983, he received the Donald McClennan Meritorious Service Award from the IEEE Communications Society, and in 1984 he was awarded an IEEE Centennial Medal. In 1987 and 1988 he received NASA Tech Brief Awards for technical innovation.