Permutation Codes for the Gaussian Broadcast Channel with Two Receivers

CHRIS HEEGARD, MEMBER, IEEE, HUGO E. dePEDRO, MEMBER, IEEE, AND JACK K. WOLF, FELLOW, IEEE

Abstract—**A deterministic coding scheme for reliable transmission over the Gaussian broadcast channel with two receivers is considered. The coding scheme is based upon Slepian's permutation mudulation codes. It is shown that it is relatively easy for both receivers to accomplish maximum likelihood detection even though one receiver must instrument a composite hypothesis test. Bounds on the performance of various codes are given. The parameters of the codes are chosen in order to achieve the best performance. The performance of these best codes are compared with results predicted by random coding and with time sharing of ordinary permutation codes.**

I. Introduction

Permutation codes were introduced by Slepian [1] as a coding scheme for an additive Gaussian noise single-user communication channel. An attractive characteristic of these codes is that one can achieve maximum likelihood decoding with a decoder whose complexity grows algebraically with block length.

In this paper we consider the use of permutation codes for the Gaussian broadcast channel with two receivers [2]. We will show that if the code is used in a particular manner, the decoding algorithm proposed by Slepian for the single-receiver case can be used for one receiver while the other receiver uses this algorithm with a slight modification. Both receivers achieve maximum likelihood decoding. Good permutation codes for the Gaussian broadcast channel are found by a computer search. The performance of these codes are compared with results previously obtained by random coding arguments.

In the next two sections, the work of Slepian on permutation codes and the work of Cover [2] and others [3]–[5] on the broadcast channel are briefly summarized. In subsequent sections, we examine in detail the notion of utilizing permutation codes for the Gaussian broadcast channel.

II. Permutation Codes for the Single-User Channel

Slepian considered two classes of permutation codes for the single-user channel: Variant I and II codes. A description of these codes follows. The codewords of the code are n-vectors with real number components. We denote these codewords as $\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_M$ where M is the number of codewords.

Variant I Codes: The codeword $\boldsymbol{u}_1$ is an n-vector of the form

$$\boldsymbol{u}_1=(\underbrace{\alpha_1,\cdots,\alpha_1}_{\longleftarrow n_1 \longrightarrow},\ \underbrace{\alpha_2,\cdots,\alpha_2}_{\longleftarrow n_2 \longrightarrow},\cdots,\ \underbrace{\alpha_k,\cdots,\alpha_k}_{\longleftarrow n_k \longrightarrow})$$

where the α_i are k real numbers such that $\alpha_1<\alpha_2<\cdots<\alpha_k$. The n_i are positive integers satisfying

$$n_1+n_2+\cdots+n_k=n.$$

The other codewords $\boldsymbol{u}_2, \boldsymbol{u}_3, \cdots, \boldsymbol{u}_M$ are chosen as *all distinct* vectors that can be obtained by rearranging (permuting) the components of $\boldsymbol{u}_1$ in all possible ways. There are a total of

$$M=n!/\prod_{i=1}^{k} n_i!$$

codewords.

Variant II Codes: The first codeword $\boldsymbol{u}_1$ is as above with the added restriction that all α_i are nonnegative. The $M-1$ other codewords are formed by assigning algebraic signs to the nonzero components of $\boldsymbol{u}_1$ in all possible ways and then rearranging these signed components in all possible ways. The number of codewords is then

$$M=\begin{cases} 2^n\left(n!/\prod_{i=1}^{k} n_i!\right), & \text{if } \alpha_1>0 \\ 2^{n-n_1}\left(n!/\prod_{i=1}^{k} n_i!\right), & \text{if } \alpha_1=0. \end{cases}$$

We next consider decoding algorithms for Variant I and II codes. We assume the following.

a) Each codeword from the code has *a priori* probability $1/M$ of being transmitted.

b) The received vector consists of the transmitted codeword plus a noise vector. The components of the noise vector are identically distributed statistically independent Gaussian variates with mean zero and variance σ^2.

c) The receiver is to choose that codeword that minimizes the probability of decoding to the wrong codeword. Such a decoding rule is called maximum likelihood decoding.

A block diagram of this system is shown in Fig. 1, where $\hat{\boldsymbol{u}}$ is that codeword selected by the decoder. Slepian has shown that the decoding algorithms that lead to the

Manuscript received November 3, 1976; revised March 23, 1978. This work was supported in part by the National Science Foundation under Grant GK-38148.

C. Heegard is with Linkabit Corporation, LaJolla, CA 92121.
H. dePedro is with GTE Sylvania, Needham, MA 02194.
J. K. Wolf is with the University of Massachusetts, Amherst, MA 01003.

0018-9448/78/0900-0569$00.75 ©1978 IEEE

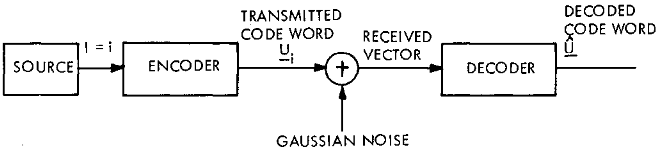

Fig. 1. Block diagram and single receiver systems.

smallest error probabilities for the two types of codes are as follows.

Variant I Decoding Algorithm:

1) Replace the n_1 smallest components of the received vector by α_1.
2) Replace the n_2 next smallest components of the received vector by α_2.

.
.
.

k) Replace the n_k largest components of the received vector by α_k.

Variant II Decoding Algorithm:

1) Replace the n_1 components of the received vector that are of the smallest absolute value by $+\alpha_1$ or $-\alpha_1$, each sign chosen to agree with that of the component it replaces. If $\alpha_1 = 0$, the sign is immaterial.
2) Replace the n_2 components of the received vector that are of the next smallest absolute value by $+\alpha_2$ or $-\alpha_2$, each sign chosen to agree with the component it replaces.

.
.
.

k) Replace the n_k components of the received vector that are of the largest absolute value by $+\alpha_k$ or $-\alpha_k$, each sign chosen to agree with the component it replaces.

We will refer to these algorithms as Slepian decoding algorithms. Slepian has derived upper and lower bounds on the probability of error for these decoding algorithms and has optimized the choices of the α_i and n_i to minimize the upper bounds.

III. Gaussian Broadcast Channel and Information Theory

The two-receiver broadcast channel as considered by Cover [2] and others [3]–[5] is a model of a communications system where a single codeword is transmitted over two distinct channels and is received by two receivers. We assume that the channels have different signal-to-noise ratios. The receiver with the better signal-to-noise ratio must decode all of the information carried by the codeword, while the other receiver must decode only some of this information. A model for the system where the channel noise is assumed additive is shown in Fig. 2.

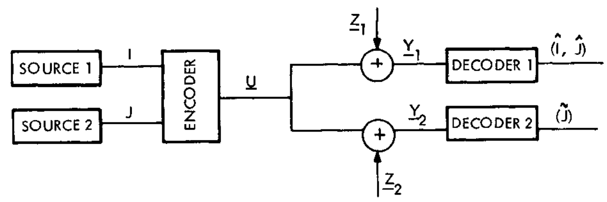

Fig. 2. Two-receiver broadcast channel (additive noise).

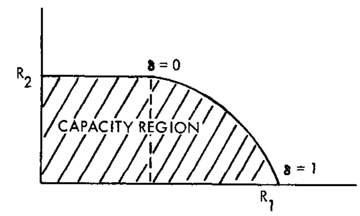

Fig. 3. Capacity region of Gaussian broadcast channel.

We now give a more detailed description of the broadcast channel. Every n time units, source 1 produces a message I, and source 2 produces a message J. Messages I and J are statistically independent random variables that are uniformly distributed over the sets $1,2,\cdots,M_1 = 2^{nr_1}$ and $1,2,\cdots,M_2 = 2^{nr_2}$, respectively. It will be convenient to define $R_1 = r_1 + r_2$ and $R_2 = r_2$. Each pair of messages is statistically independent of all other message pairs.

The encoder has available a set of $M = M_1 \cdot M_2$ codewords $\boldsymbol{u}_i$, $i = 1,2,\cdots,M$, each one of which is a vector with n real components. The encoder is a one-one mapping from the M message pairs (I,J) to the M codewords. The codewords are chosen such that the average value of the sum of the squares of their components (averaged over all codewords) is constrained to be less than or equal to nS.

Decoder 1 receives the transmitted vector corrupted by the noise vector z_1, a vector that adds component by component to the transmitted codeword. The components of z_1 are statistically independent Gaussian random variables of zero mean and variance σ_1^2. The codeword received by decoder 2 suffers a similar fate, but the components of the additive Gaussian vector z_2 have variance σ_2^2.[1]

Decoder 1 produces an estimate of both I and J. Call its estimates $(\hat{I},\hat{J})$. Decoder 2 produces an estimate only of J. Its estimate will be called $\tilde{J}$. The measures of goodness for the system are the probabilities of error for the two decoders, P_1 and P_2, defined as

$$\text{decoder 1:} \quad P_1 = P\{(\hat{I} \neq I) \cup (\hat{J} \neq J)\}$$

$$\text{decoder 2:} \quad P_2 = P\{\tilde{J} \neq J\}.$$

Of interest are those rate pairs (r_1, r_2), or equivalently (R_1, R_2), such that both P_1 and P_2 can be made as small as desired by choosing n large enough. This region called the capacity region is found by a random coding argument. (Actually the random coding argument only shows that

[1]An equivalent model would have y_2, the received vector for decoder 2, formed by adding a Gaussian vector to y_1, the received vector for decoder 1. In this form, the channel is referred to as a *degraded* broadcast channel.

good codes exist in that region. A converse theorem is used to prove that good codes do not exist outside the region.) The capacity region has been shown to be given by the following parametric equations:

$$R_1 \leqslant \frac{1}{2}\log_2\left(1+\frac{\bar{\delta} S}{\delta S+\sigma_2^2}\right)+\frac{1}{2}\log_2\left(1+\frac{\delta S}{\sigma_1^2}\right) \text{ bits/time unit}$$

$$R_2 \leqslant \frac{1}{2}\log_2\left(1+\frac{\bar{\delta} S}{\delta S+\sigma_2^2}\right) \text{ bits/time unit.}$$

Here δ is a parameter that is allowed to vary over the range (0, 1), and $\bar{\delta}$ is defined as $(1-\delta)$. The general shape of the region is shown in Fig. 3.

The discrete-time model described is often taken to represent two band-limited Gaussian channels both of bandwidth W where all signals are sampled every $1/2W$ s. In that case, the basic unit of time is $1/2W$ s. In addition, if all logarithms are taken base 10, the rates measured in units of dits/Hz and denoted (R_1', R_2') are bounded as

$$R_1' \leqslant \log_{10}\left(1+\frac{\bar{\delta} S}{\delta S+\sigma_2^2}\right)+\log_{10}\left(1+\frac{\delta S}{\sigma_1^2}\right) \text{ dits/Hz}$$

$$R_2' \leqslant \log_{10}\left(1+\frac{\bar{\delta} S}{\delta S+\sigma_2^2}\right) \text{ dits/Hz.}$$

Treating these inequalities as equalities and only considering the interesting part of the boundary where $R_2' < R_1'$, one can solve for S/σ_1^2 in terms of R_1', R_2', and $\sigma_2^2/\sigma_1^2 \triangleq K$ $(K \geqslant 1)$ as

$$\frac{S}{\sigma_1^2} = 10^{R_1'} + 10^{R_2'}(K-1) - K \triangleq S_{\text{ideal}}.$$

S_{ideal} is the minimum normalized signal power required to achieve the rates R_1' and R_2' for a ratio of channel noise powers K. Later the actual normalized signal power of deterministic codes that achieve low probabilities of error $(P_1 \leqslant 10^{-5}, P_2 \leqslant 10^{-5})$ will be compared with S_{ideal}.

IV. Permutation Codes for the Gaussian Broadcast Channel

We next consider how permutation codes can be used to transmit information over the Gaussian broadcast channel with two receivers. Several different approaches were considered, and only the best of these is discussed here. The basic problem was to find a technique that led to two decoders of reasonable complexity while yielding small probabilities of error for reasonable signal-to-noise ratios.

The basic coding technique is as follows. The set of M codewords (from either a Variant I or II permutation code) is partitioned into M_2 sets, each set containing $M_1 = M/M_2$ codewords. The encoder mapping from the message pairs to the codewords is such that when the sources produce the pair $(I,J)=(i,j)$, the encoder produces the ith codeword from the jth set.

The first decoder that operates on the received signal with the higher signal-to-noise ratio produces the minimum error probability estimate of which codeword was transmitted. This results in the message estimate $(\hat{I},\hat{J})$. Such a decoder can utilize the appropriate Slepian decoding algorithm to achieve this minimum error probability.

The second decoder that operates on the received signal with the lower signal-to-noise ratio must make a minimum probability of error estimate of the *set* to which the transmitted codeword belonged. This second decoder must instrument a composite hypothesis test that usually requires a very complex algorithm. However, we will show that if the sets are chosen in a particular manner, this second decoder can be instrumented in a manner almost identical to the first decoder. In particular, for a given method of choosing the sets, the second decoder uses the appropriate Slepian decoding algorithm to find the minimum probability of error estimate of both I and J and then ignores the estimate for I. The resultant estimate for J, namely $\tilde{J}$, is then the minimum probability of error estimate of the set to which the signal belonged.

We now explain how to choose the M_2 sets. We begin by considering Variant I codes. For simplicity, we initially consider the special case where all n components of the vector $\boldsymbol{u}_1$ are distinct: i.e., $n_1 = n_2 = \cdots = n_k = 1$. Later we shall remove this restriction and also consider Variant II codes.

We consider that the codeword $\boldsymbol{u}_1$ is broken up into L *segments* as

$$\boldsymbol{u}_1 = (\overbrace{\alpha_1,\alpha_2,\cdots,\alpha_{m_1}}^{\longleftarrow m_1 \longrightarrow} \,\vdots\, \overbrace{\alpha_{m_1+1},\cdots,\alpha_{m_1+m2}}^{\longleftarrow m_2 \longrightarrow} \,\vdots\, \cdots \,\vdots\, \overbrace{\cdots \alpha_n}^{\leftarrow m_L \rightarrow})$$

where the borders of the segments are shown by dotted vertical lines. The number of components of $\boldsymbol{u}_1$ in the jth segment is m_j.

We now consider all those permutations that rearrange components *within* a segment but do not interchange components that are in distinct segments. There will be $M_1 = \Pi_{j=1}^{L}(m_j)!$ such permutations. Using a double subscript notation for the codewords (the reason for which will become apparent soon) we call these codewords $\boldsymbol{u}_{11}$, $\boldsymbol{u}_{12}, \cdots \boldsymbol{u}_{1M_1}$. For example if $\boldsymbol{u}_1 = (-2, 0, 1 \,\vdots\, 4, 5)$, then

$$\begin{aligned}
\boldsymbol{u}_{11} &= (-2, \quad 0, \quad 1 \mid 4, \quad 5)\\
\boldsymbol{u}_{12} &= (-2, \quad 1, \quad 0 \mid 4, \quad 5)\\
\boldsymbol{u}_{13} &= (\ \ 0, \quad -2, \quad 1 \mid 4, \quad 5)\\
\boldsymbol{u}_{14} &= (\ \ 0, \quad 1, \quad -2 \mid 4, \quad 5)\\
\boldsymbol{u}_{15} &= (\ \ 1, \quad -2, \quad 0 \mid 4, \quad 5)\\
\boldsymbol{u}_{16} &= (\ \ 1, \quad 0, \quad -2 \mid 4, \quad 5)\\
\boldsymbol{u}_{17} &= (-2, \quad 0, \quad 1 \mid 5, \quad 4)\\
\boldsymbol{u}_{18} &= (-2, \quad 1, \quad 0 \mid 5, \quad 4)\\
\boldsymbol{u}_{19} &= (\ \ 0, \quad -2, \quad 1 \mid 5, \quad 4)\\
\boldsymbol{u}_{1,10} &= (\ \ 0, \quad 1, \quad -2 \mid 5, \quad 4)\\
\boldsymbol{u}_{1,11} &= (\ \ 1, \quad -2, \quad 0 \mid 5, \quad 4)\\
\boldsymbol{u}_{1,12} &= (\ \ 1, \quad 0, \quad -2 \mid 5, \quad 4).
\end{aligned}$$

These $M_1 = 12$ vectors make up the first set in the partition.

It is well known that all $n!$ permutations of the vector $\boldsymbol{u}_1$ form a *group* with respect to an operation that permutes components of the vectors. Furthermore, the M_1 permutations of the vector $\boldsymbol{u}_1$, which are formed by considering all rearrangements of components within segments but not interchanging components from different segments, form a *subgroup* of this group.

Given any group and any subgroup of the group, one can obtain a coset decomposition of the group. In our case there will be $M_2 = M/M_1 = n!/\pi_{j=1}^{L}(m_j)!$ such cosets, each coset containing $\pi_{j=1}^{L}(m_j)!$ vectors. It is these cosets that form the sets in our partition. Since permutations do not commute, we must specify the order of the operations used in forming the cosets. Rather than set up here the complicated notation needed to treat the problem in general, we illustrate the procedure by a simple example. Let $\boldsymbol{u}_1 = (-1,\ \ 0 \mid 3,\ \ 4)$, and

$$\begin{array}{ll}
\boldsymbol{u}_{11}=(-1,\ \ 0 \mid 3,\ \ 4) & \boldsymbol{u}_{21}=(-1,\ \ 3 \mid 0,\ \ 4) \\
\boldsymbol{u}_{12}=(\ \ 0,\ -1 \mid 3,\ \ 4) & \boldsymbol{u}_{22}=(\ \ 0,\ \ 3 \mid -1,\ \ 4) \\
\boldsymbol{u}_{13}=(-1,\ \ 0 \mid 4,\ \ 3) & \boldsymbol{u}_{23}=(-1,\ \ 4 \mid 0,\ \ 3) \\
\boldsymbol{u}_{14}=(\ \ 0,\ -1 \mid 4,\ \ 3) & \boldsymbol{u}_{24}=(\ \ 0,\ \ 4 \mid -1,\ \ 3) \\
\\
\boldsymbol{u}_{31}=(-1,\ \ 4 \mid 3,\ \ 0) & \boldsymbol{u}_{41}=(3,\ \ 0 \mid -1,\ \ 4) \\
\boldsymbol{u}_{32}=(\ \ 0,\ \ 4 \mid 3,\ -1) & \boldsymbol{u}_{42}=(3,\ -1 \mid 0,\ \ 4) \\
\boldsymbol{u}_{33}=(-1,\ \ 3 \mid 4,\ \ 0) & \boldsymbol{u}_{43}=(4,\ \ 0 \mid -1,\ \ 3) \\
\boldsymbol{u}_{34}=(\ \ 0,\ \ 3 \mid 4,\ -1) & \boldsymbol{u}_{44}=(4,\ -1 \mid 0,\ \ 3) \\
\\
\boldsymbol{u}_{51}=(4,\ \ 0 \mid 3,\ -1) & \boldsymbol{u}_{61}=(3,\ \ 4 \mid -1,\ \ 0) \\
\boldsymbol{u}_{52}=(4,\ -1 \mid 3,\ \ 0) & \boldsymbol{u}_{62}=(3,\ \ 4 \mid 0,\ -1) \\
\boldsymbol{u}_{53}=(3,\ \ 0 \mid 4,\ -1) & \boldsymbol{u}_{63}=(4,\ \ 3 \mid -1,\ \ 0) \\
\boldsymbol{u}_{54}=(3,\ -1 \mid 4,\ \ 0) & \boldsymbol{u}_{64}=(4,\ \ 3 \mid 0,\ -1).
\end{array}$$

For this example there are six sets, each set consisting of four vectors with two segments each, indicated by the dotted lines. The details of the procedure are explained in Appendix A.

In the general situation, the ith element of the jth set would be denoted $\boldsymbol{u}_{ji}$. This would be the vector selected by the encoder given that the sources produced the pair $(I,J)=(i,j)$. Decoder 1 would use the Slepian decoding algorithm to decode. Decoder 2 would also use the Slepian decoding algorithm to first select a codeword. This word may indeed be different from that word selected by decoder 1 since there is different noise on the channels, and thus the decoders operate on different data. Assume the result of decoder 2 using the Slepian algorithm results in a codeword in the $\tilde{j}$th set. (It is not important which codeword in that set was selected.) The decoder then outputs the estimate $\tilde{J}=\tilde{j}$. A sketch of a proof is given in Appendix A showing that such a decoder achieves the minimum probability of error in choosing among the sets.

The procedure for forming the sets of codewords when not all components of the vector $\boldsymbol{u}_1$ are distinct is as follows. Again, the first set is formed from all permutations of $\boldsymbol{u}_1$ that do not interchange components in different segments. The only restriction is that if one component of $\boldsymbol{u}_1$ is in a particular segment, then all components identical to that component are in that segment. For example, the following segmentation of $\boldsymbol{u}_1$ would *not* be allowed: $\boldsymbol{u}_1 = (112 \mid 23)$, since the element two in two different segments. The other sets are chosen as the cosets.

The partitioning of the codewords in Variant II codes into sets is now described by considering the following example. Let $\boldsymbol{u}_1 = (0,\ \ 1 \mid 2)$, and

$$\begin{array}{ll}
\boldsymbol{u}_{11}=(0,\ \ 1 \mid 2) & \boldsymbol{u}_{21}=(2,\ \ 1 \mid 0) \\
\boldsymbol{u}_{12}=(1,\ \ 0 \mid 2) & \boldsymbol{u}_{22}=(2,\ \ 0 \mid 1) \\
\\
\boldsymbol{u}_{31}=(0,\ \ 2 \mid 1) & \boldsymbol{u}_{41}=(\ \ 0,\ -1 \mid 2) \\
\boldsymbol{u}_{32}=(1,\ \ 2 \mid 0) & \boldsymbol{u}_{42}=(-1,\ \ 0 \mid 2) \\
\\
\boldsymbol{u}_{51}=(2,\ -1 \mid 0) & \boldsymbol{u}_{61}=(\ \ 0,\ \ 2 \mid -1) \\
\boldsymbol{u}_{52}=(2,\ \ 0 \mid -1) & \boldsymbol{u}_{62}=(-1,\ \ 2 \mid 0) \\
\\
\boldsymbol{u}_{71}=(0,\ \ 1 \mid -2) & \boldsymbol{u}_{81}=(-2,\ \ 1 \mid 0) \\
\boldsymbol{u}_{72}=(1,\ \ 0 \mid -2) & \boldsymbol{u}_{82}=(-2,\ \ 0 \mid 1) \\
\\
\boldsymbol{u}_{91}=(0,\ -2 \mid 1) & \boldsymbol{u}_{10,1}=(\ \ 0,\ -1 \mid -2) \\
\boldsymbol{u}_{92}=(1,\ -2 \mid 0) & \boldsymbol{u}_{10,2}=(-1,\ \ 0 \mid -2) \\
\\
\boldsymbol{u}_{11,1}=(-2,\ -1 \mid 0) & \boldsymbol{u}_{12,1}=(\ \ 0,\ -2 \mid -1) \\
\boldsymbol{u}_{11,2}=(-2,\ \ 0 \mid -1) & \boldsymbol{u}_{12,2}=(-1,\ -2 \mid 0).
\end{array}$$

The general procedure for forming the table is to form the sets as for Variant I codes with the signs of all components positive and then to form new sets by repeating these sets using all distinct signs for the nonzero components.

It should be noted that the extra sign information (n or $(n-n_1)$ bits/channel use) is decoded by both receivers.

V. Bounds on the Probability of Error

Slepian derived both upper and lower bounds for the probability of decoding error for his decoding algorithm. These bounds apply directly to decoder 1. We will use only the upper bounds. In Appendix B, we derive upper bounds to the probability of error for the second decoder for both Variant I and II codes.

For the numerical results given in the next section, one minor modification was made for the case of Variant II codes where $\alpha_1 = 0$. For this case, the second decoder does not assign signs to *any* of the components in the first segment. This has the effect of decreasing the rate to the decoder while at the same time decreasing the probability of error for the decoder. The first decoder does assign signs to all nonzero components.

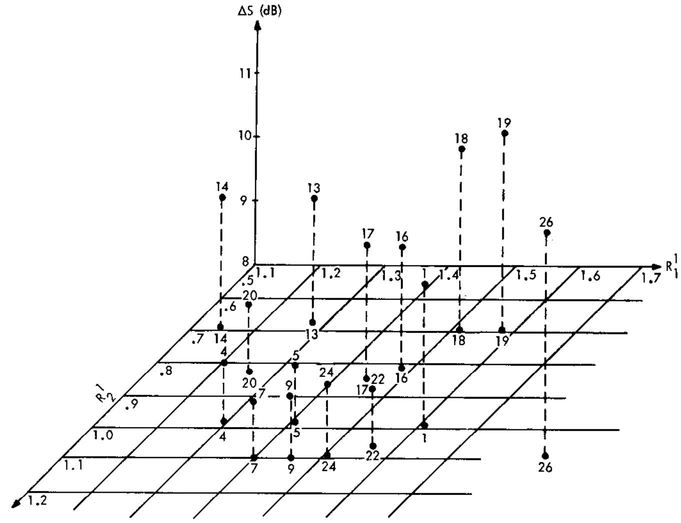

Fig. 4. Performance of codes ($K=10$, $P_1=P_2\leq 10^{-5}$).

VI. Performance of Codes

In this section, we describe results obtained by a computer program that performed a gradient search to find the parameters of good Variant II codes. The following conditions were assumed in this search.

a) The value of $K=\sigma_2^2/\sigma_1^2$ was set equal to 10. ($\sigma_1^2=1$ for normalization.)

b) The upper bounds on P_1 and P_2, the probabilities of error, were set equal to 10^{-5}.

c) For each code, the following values were fixed:

1) block length n,
2) size of the segments $m_1, m_2, \cdots, m_k$,
3) multiplicity of the components $n_1, n_2, \cdots, n_L$.

d) The amplitudes of the components were varied in the search until the smallest signal power S was achieved. It was always assumed, however, that the vector $\boldsymbol{u}_1$ began with n_1 zeros.

The best amplitudes for certain codes meeting these requirements are given in Table I. To explain the entries in the table, the first code (code 1) has

a) block length 10,
b) three segments, the first with four components, the second with five components, and the last with one component,
c) $\boldsymbol{u}_1$ starts with two zeros, then has one component with the next amplitude, one component with the next amplitude, etc.,
d) the vector $\boldsymbol{u}_1$ is given as
$\boldsymbol{u}_1=(0,\ 0,\ 6.6841,\ 13.1562\ 33.1201,\ 33.1201,\ 40.0098,\ 40.0098,\ 46.8531,\ 67.4472)$.

For each code the value of S was computed such that the upper bounds for the probabilities of error for both decoders were equal to 10^{-5}. Then after computing R_1' and R_2', S_{ideal} was determined from the formula given in Section III. Finally $\Delta S=S-S_{\text{ideal}}$ was computed for each code and plotted in dB versus R_1' and R_2' in Fig. 4. As can be seen, the codes require about 9 dB more signal power than that power predicted by information theory for very long random codes. Numerical values of ΔS for the various codes are given in Table II.

TABLE I
PARAMETERS OF CODES

Code #	n	n_i's and m_i's	Amplitudes			
1	10	211/221/1	0.	6.7841 40.0098	13.1562 46.8531	33.1201 67.4472
4	10	31/41/1	0.	6.6775 54.6912	27.2913	34.0939
5	10	13/41/1	0.	6.6685 55.6230	28.3092	35.1217
7	10	21/41/2	0.	6.5820 54.9714	27.3243	34.0814
9	10	12/41/2	0.	6.5802 55.5812	27.9569	34.7199
13	10	16/11/1	0.	6.9053 54.5443	28.0459	34.3743
14	10	52/11/1	0.	7.0693 53.6634	27.1799	33.4686
16	10	15/111/1	0.	6.9552 40.6995	27.8499 61.0288	34.2299
17	10	24/11/11	0.	7.0619 54.2385	27.9372 60.8449	34.2672
18	10	1141/2/1	0.	6.5360 40.2017	13.2893 61.1071	20.1253
19	10	1141/11/1	0.	6.5850 39.6873	13.3853 46.2806	20.2633 66.5394
20	7	13/2/1	0.	6.5702	27.7159	48.5341
22	20	55/42/4	0.	7.3051 58.4083	29.5432	36.449
24	10	12/32/2	0.	6.5986 56.0613	27.6881	34.5324
26	10	111/11111/11	0.	6.7255 39.2061 59.1436	13.2489 45.8006 79.4198	32.6464 52.4410 86.3957

TABLE II
PERFORMANCE OF CODES

Code #	R_1'	R_2'	ΔS(dB)
1	1.613	.981	10.2317
4	1.301	.981	8.7303
5	1.422	.981	8.7357
7	1.397	1.101	9.0116
9	1.457	1.101	9.0393
13	1.282	.692	9.90685
14	1.137	.692	10.0368
16	1.438	.826	9.8889
17	1.457	.861	10.2013
18	1.518	.692	10.8219
19	1.578	.692	11.1125
20	1.265	.8355	9.1399
22	1.568	1.06	8.8633
24	1.517	1.101	9.0869
26	1.854	1.102	11.5347

It is natural to compare the performance of the aforementioned coding technique with those results obtained by time-sharing two of Slepian's permutation modulation codes. Let C_1 and C_2 be two permutation modulation codes with rates R_{c1} and R_{c2}, respectively (measured in

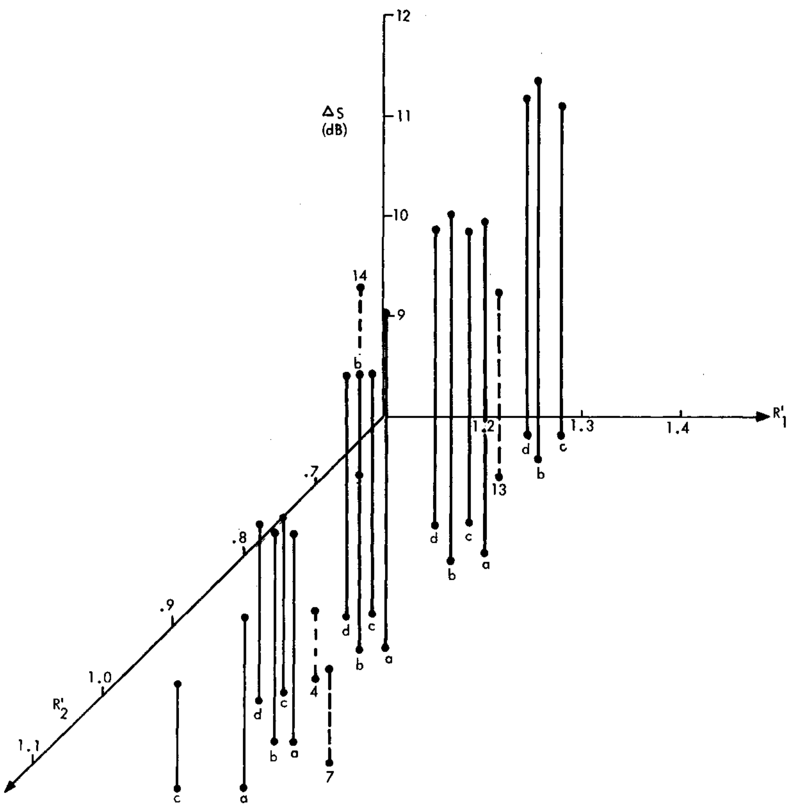

Fig. 5. Comparison of performance of codes with time-shared codes.

dits/Hz). C_1 and C_2 may be two distinct codes or the same code. We transmit a codeword from the code C_1 for a fraction of the time equal to $(1-\lambda)$ where $0 \leqslant \lambda \leqslant 1$, at a power P_1. We transmit a codeword from the code C_2 for a fraction of the time equal to λ at a power $P_2 > P_1$. The low noise receiver (receiver 1) attempts to decode both codewords, while the high noise receiver (receiver 2) only attempts to decode the codeword from code C_2. The rates of transmission to the two receivers then are

$$R_1' = (1-\lambda)R_{c1} + \lambda R_{c2} \quad \text{(dits/Hz)}$$
$$R_2' = \lambda R_{c2} \quad \text{(dits/Hz)},$$

while the average power is given as

$$P_{av} = (1-\lambda)P_1 + \lambda P_2.$$

Two codes, each of block length 10, were chosen to compare with broadcast codes of the same block length. These codes have the following parameters.

	n	m	Amplitudes
Code A	10	3, 3, 3, 1	0, 7.0814, 14.0512, 21.0215
(Code 13, Table I, Slepian)			
Code B	10	2, 4, 2, 2	0, 7.0672, 14.0365, 20.9950
(Code 14, Table I, Slepian)			

The codes were chosen so that the rates fell into a range comparable to those rate points found in Table I. Four classes of codes were considered.

	C_1	C_2
Class a	code A	code A
Class b	code A	code B
Class c	code B	code A
Class d	code B	code B

The performance of these time-shared codes are given in Fig. 5 along with a few of the broadcast codes. Although no broadcast code had exactly the same rate pair as a time-shared code, broadcast codes are seen to outperform time-shared codes having similar rates.

Acknowledgment

The authors wish to acknowledge the detailed comments of the referees that were of great help in revising this paper.

Appendix A

This appendix serves several purposes. It describes how the codewords are partitioned into sets by means of a coset decomposition. It proves the optimality of the Slepian decoding algorithm for decoder 1, using a graphical technique. This technique then is used to give a sketch of a proof of the optimality of a modified Slepian decoding algorithm for decoder 2.

A permutation p is an operator that transforms one real n-vector into another real n-vector by permuting (rearranging) the components of the vector. If $p=(i,j,k,\cdots,l,m)$, then $v=pu$ is a vector where the jth smallest element of u is replaced by the

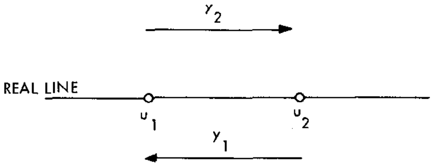

Fig. 6. Dip for permutation $p=(1,2)$.

ith smallest element of $\boldsymbol{u}$, the kth smallest element is replaced by the jth smallest element, $\cdots$, the mth smallest element is replaced by the lth smallest element, and the ith smallest element is replaced by the mth smallest element.

We first consider the Slepian decoding algorithm for decoder 1 and show that it leads to the smallest possible probability of error. It is well known that if the codewords are of equal energy and equal *a priori* probabilities, the minimum probability of error decoding algorithm chooses the codeword that maximizes the likelihood function. For the additive Gaussian channel, the optimum decoder finds the codeword $\boldsymbol{u}_i$ that maximizes the inner product $(\boldsymbol{y}_1, \boldsymbol{u}_i)$.

For any received vector $\boldsymbol{y}_1$, take the codeword $\boldsymbol{u}$ that has its components ordered the same as $\boldsymbol{y}_1$. Consider another codeword that is related to $\boldsymbol{u}$ by a permutation p, i.e., $\boldsymbol{v}=p\boldsymbol{u}$. We consider a quantity to be called the *difference inner product* (abbreivated dip) of p defined as $(\boldsymbol{y}_1, \boldsymbol{\Delta})_p \equiv (\boldsymbol{y}_1, \boldsymbol{u}-\boldsymbol{v})$. We will show that the dip is always greater than or equal to zero for all permutations with equality resulting from the identity permutation $\boldsymbol{v}=\boldsymbol{u}$. This will prove the optimality of the Slepian decoding algorithm since it shows that $(\boldsymbol{y}_1, \boldsymbol{u}) \geqslant (\boldsymbol{y}_1, \boldsymbol{v})$ for all $\boldsymbol{v}=p\boldsymbol{u}$ if $\boldsymbol{u}$ is the codeword whose components are in the same order as the components of $\boldsymbol{y}_1$.

We begin by considering a simple example. Assume that $p=(1,2)$ so that $\boldsymbol{u}$ and $\boldsymbol{v}$ agree in all positions except for the two smallest components. Let u_1 and u_2 be the smallest and next smallest components of $\boldsymbol{u}$, respectively. Similarly, let y_1 and y_2 be the smallest and next smallest components of $\boldsymbol{y}_1$. Then the dip is $(\boldsymbol{y}_1, \boldsymbol{\Delta})_p = y_1(u_1-u_2)+y_2(u_2-u_1)=(y_2-y_1)(u_2-u_1)$, which is certainly greater than zero since $y_2>y_1$ and $u_2>u_1$. A diagram of this result is Fig. 6. Points u_1 and u_2 are plotted on the real line with u_2 to the right of u_1 since $u_2>u_1$. An arrow above the real line pointing from u_1 to u_2 contains a "positive" weighting y_2. An arrow below the line pointing from u_2 to u_1 represents a "negative" weighting y_1.[2] The net weight of the length of the line segment from u_1 to u_2 (i.e., (u_2-u_1)) is the sum of all the positive weightings minus the sum of all the negative weightings.

More complicated dips can be represented in this manner. We consider only permutations with a single cycle since multiple-cycle permutations would simply involve multiple graphs. For example, the permutation $p=(i,j,k,\cdots,l,m)$ is drawn according to the following rules.

1) Label the points $u_i, u_j, u_k, \cdots, u_l, u_m$ along the real line.
2) Draw an arrow from u_i to u_j, u_j to $u_k, \cdots, u_l$ to u_m, and u_m to u_i. If the arrow points to the right, draw it above the real line; if it points to the left, draw it below the line.
3) Weight each arrow according to its termination, e.g., for u_i to u_j the weight is y_j.

[2]The terms "positive" and "negative" here do not refer to the signs of y_1 and y_2, but rather to the fact that the length of the line segment from u_1 to u_2 is weighted by (y_2-y_1).

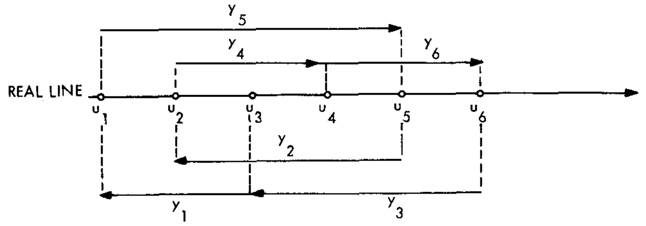

Fig. 7. Dip for permutation $p=(1,5,2,4,6,3)$.

The example of Fig. 7 should clarify this procedure. From the graph we find

$$\begin{aligned}(y_1,\boldsymbol{\Delta})_p &= (y_6-y_3)(u_6-u_5)+(y_5+y_6-y_2-y_3)(u_5-u_4)\\ &\quad+(y_5+y_4-y_2-y_1)(u_4-u_3)+(y_5+y_4-y_2-y_1)(u_3-u_2)\\ &\quad+(y_5-y_1)(u_2-u_1).\end{aligned}$$

The dip $(y_1,\boldsymbol{\Delta})_p$, as previously described, is positive for any permutation p and equal to zero for the identity permutation. This follows from two facts. The first is that every permutation forms a closed cycle. This means that the total length of positive-weighted line segments (in the dip diagram) is equal to the total length of negative-weighted line segments. The second fact is a result of rule 3), which assigns the weights to the dip diagram. For each weighted line segment u_i to u_{i+1}, the positive weights are $\geqslant y_{i+1}$, while the negative weights are $\leqslant y_i$. Since $y_{i+1}>y_i$, the net weight must be >0, i.e., positive. Since decoder 1 must find the maximum inner product, the optimum decoding algorithm is to decode to the codeword that has the same order as the received vector $\boldsymbol{y}_1$. This is precisely the Slepian decoding algorithm.

We now turn to the decoding algorithm for the second decoder. Again, if all codewords have equal energy and equal *a priori* probability, the decoding algorithm that minimizes the probability of choosing the wrong subset of codewords is the one that maximizes the likelihood function over all the subsets. The likelihood function over the Gaussian channel for the ith codeword subset is

$$\sum_j \exp\frac{-\|\boldsymbol{y}_2-\boldsymbol{u}_{ij}\|^2}{2} = k\sum_j \exp^{(\boldsymbol{y}_2,\boldsymbol{u}_{ij})}$$

where $\|\cdot\|$ denotes the Euclidean norm.

This likelihood function is maximized for the kth subset if for every $i\neq k$

$$(\boldsymbol{y}_2,\boldsymbol{u}_{kj}) \geqslant (\boldsymbol{y}_2,\boldsymbol{u}_{ij})$$

for all values of j $(j=1,2,\cdots,M_1)$.

We now consider how to form the M_2 subsets of codewords. We will use the fact that the set of all unique permutations on a vector $\boldsymbol{u}$ forms a non-Abelian group. We begin by partitioning the elements of $\boldsymbol{u}$ into L ordered classes C_i, $1\leqslant i\leqslant L$. The maximum component in class C_i is less than the minimum component in class C_j if $i<j$. We now form a subgroup of permutations S. The set S is described in terms of the characteristics of the dip $(\boldsymbol{y}_2,\boldsymbol{\Delta})_p$ diagram. A permutation is a member of S if the weighted arrows of the dip diagram do not cross class boundaries. Certainly the identity permutation is contained in S.

As an example we consider the simplest nontrivial case where $n=3$ and $L=2$. Specifically, we let $C_1=\{u_1^1,u_2^1\}$, $C_2=\{u_3^2\}$ (superscript indicates class membership); then $S=\{I,(1,2)\}$. The permutation $(1,2)\in S$, but $(2,3)\notin S$ since the former's arrows do not cross the class boundary while the latters do (Fig. 8).

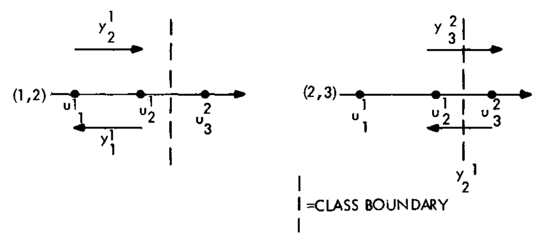

Fig. 8. Dip for permutation $\in S$ and $\notin S$.

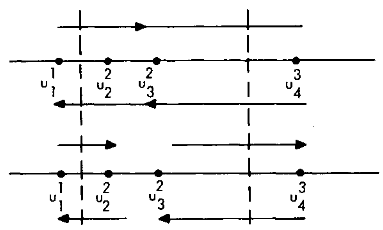

Fig. 9. "Broken" cycles yield permutations that satisfy condition 2 on T.

The set S forms a subgroup. The cardinality of the subgroup is

$$|S| = \prod_{i=1}^{L} |C_i|! \equiv M_1.$$

Since we have now defined a subgroup to the group of all permutations on $\boldsymbol{u}$, we can decompose the group into cosets. There are $n!/|S| \equiv M_2$ cosets, each represented by a permutation called the coset leader. The set of coset leaders is represented by T. The ith coset consists of the composition permutations $s_j t_i$, $t_i \in T$, $s_j \in S$ for all values of j (the permutation s_j would be performed before t_i). For purposes of proving optimality, the set T is chosen judiciously. The permutation p is acceptable as a coset leader if the dip diagram satisfies the following conditions.

1) The weighted arrows always cross at least one class boundary.
2) No two arrows of opposite directions terminate in the same class.

By definition, the identity permutation is included in T, and the only permutation in common between S and T is the identity. We give a plausibility argument, rather than a proof, that we can always find a permutation for the coset leader of each coset that satisfies the two conditions. The first condition is intuitively reasonable. Since the permutations of S do not cross class boundaries, the coset leaders must do so if we wish to complete the coset decomposition. It is slightly harder to see that the second condition can *always* be met. Let us consider an example. Suppose $n=4$, $L=3$, $C_1=\{u_1^1\}$, $C_2=\{u_2^2, u_3^2\}$, and $C_3=\{u_4^3\}$. We might try the permutation $(1,2,4,3)$ as a coset leader, but the second condition could be violated. To remedy this, the permutation cycle could be "broken" into the two cycles $(1,2)$ and $(3,4)$ that would both satisfy the second condition and would generate the desired coset (Fig. 9). This breaking of the permutation cycles can always be done and will yield the desired coset leader [6].

Consider the "modified" difference inner product (mdip) $(\boldsymbol{y}_2, \boldsymbol{\Delta})_{s,t}$, $\boldsymbol{\Delta} = \boldsymbol{v} - \boldsymbol{w}$. The vectors $\boldsymbol{u}$, $\boldsymbol{v}$, and $\boldsymbol{w}$ are related by the permutations $\boldsymbol{v} = s\boldsymbol{u}$, $\boldsymbol{w} = t\boldsymbol{v}$ where $s \in S$ and $t \in T$. The ordering of $\boldsymbol{u}$ and $\boldsymbol{y}_2$ are assumed the same. We examine the influence of the permutation t on the mdip. For a given t there exists a set of

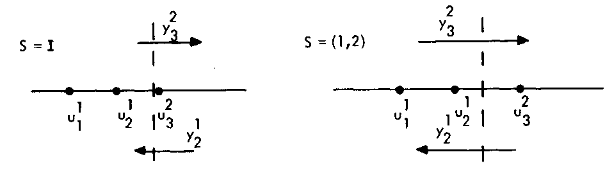

Fig. 10. Two mdip diagrams for $t=(2,3)$.

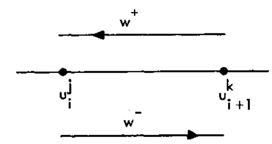

Fig. 11. Manner in which w^- and w^+ are assigned.

M_1 mdip that can be represented by a mdip diagram similar to the diagrams previously used.

Consider the simplest nontrivial case ($n=3$, $L=3$, $C_1=u_1^1, u_2^1$, $C_2=u_3^2$) for the permutation $t=(2,3)$. The two mdip diagrams (one for $s=I$ and one for $s=(1,2)$) are shown in Fig. 10. From the diagram, we see that the mdip are

$$(\boldsymbol{y}_2, \boldsymbol{\Delta})_{t=(2,3), s=I} = (y_3^2 - y_2^1)(u_3^2 - u_2^1)$$
$$(\boldsymbol{y}_2, \boldsymbol{\Delta})_{t=(2,3), s=(1,2)} = (y_3^2 - y_2^1)(u_3^2 - u_1^1).$$

It is important to recognize the common characteristics of any set of mdip diagrams for a given permutation $t \in T$. Each diagram has the same number of arrows with each corresponding arrow having the same weight. Also, the end points of each corresponding arrow (same weight on arrow) lie in the same class (i.e., the superscripts of the end points of each arrow are independent of the permutations $s \in S$ for each $t \in T$).

The mdip is positive for all $s \in S$ for any $t \in T$. When $t = I$, the mdip is zero. This is a result of the fact that the permutations are closed as well as the manner in which the superscripts are assigned on the weights of the mdip diagram. Consider the assignment of the weights on the line segment u_i^j to u_{i+1}^k (see Fig. 11). There are two cases we must consider.

Case 1: If $k=j+1$, then $w^+ \geqslant y^{j+1}$ (y^i = an element from the ith class), and $w^- \leqslant y^j$ (condition 1) on coset leaders). Since $y^i > y^l$ if $i > l$, then $w^+ - w^- > 0$ (i.e., the net weight on the length of the line segment u_i^j to u_{i+1}^k is positive).

Case 2: If $k=j$, then $w^+ \geqslant y^j$, and $w^- \leqslant y^j$. The net weight $w^+ - w^-$ will be definely positive iff either $w^+ > y^j$ or $w^- < y^j$. This will always be the case due to condition 2) on the coset leaders.

Therefore the mdip $(\boldsymbol{y}_2, \boldsymbol{\Delta})_{t,s}$ will be positive for every $t \in T$ and $s \in S$. This leads to the maximum likelihood decoding algorithm for decoder 2. The encoder will transmit a codeword from a fixed coset decomposition table based on the vector $\boldsymbol{u}$. The receiver will attempt to determine which coset the received vector $\boldsymbol{y}_2$ belongs. If the ordering of $\boldsymbol{y}_2$ is the same as the vector $\boldsymbol{u}$, then the maximum likelihood estimate of the coset is that which has the identity coset leader since the mdip is positive for all values of $t \in T$. If the ordering of $\boldsymbol{y}_2$ is related to $\boldsymbol{u}$ by the permutation p, then ideally the receiver could apply the inverse permutation p^{-1} to $\boldsymbol{y}_2$ and to the coset decomposition table (which would leave all dip unchanged) and then decode in the same manner as if $\boldsymbol{y}_2$ and $\boldsymbol{u}$ were of the same ordering. The net result of this discussion is that the optimum decoding algorithm for decoder 2 is to first decode the received vector $\boldsymbol{y}_2$ by the Slepian decoding algorithm giving $\hat{\boldsymbol{u}}$. The estimate of the coset of the transmitted codeword is the coset that contains $\hat{\boldsymbol{u}}$.

Appendix B
Upper Bound on Probability of Error for Second Decoder

Variant I Codes

We relabel the components of $\boldsymbol{u}_1$ as

$$\boldsymbol{u}_1 = \begin{array}{c|c|c} \overbrace{(\alpha_{11}\cdots\alpha_{11}}^{n_{11}}\ \overbrace{\alpha_{12}\cdots\alpha_{12}}^{n_{12}}\ \cdots\ \overbrace{\alpha_{1p_1}\cdots\alpha_{1p_1}}^{n_{1p_1}} & \overbrace{\alpha_{21}\cdots\alpha_{21}}^{n_{21}}\cdots & \cdots\ \overbrace{\alpha_{L1}\cdots\alpha_{L1}}^{n_{L1}}\cdots\ \overbrace{\alpha_{Lp_L}\cdots\alpha_{Lp_L})}^{n_{Lp_L}} \\ \longleftarrow m_1 \longrightarrow & \longleftarrow m_2 \longrightarrow & \longleftarrow m_L \longrightarrow \\ \text{segment } 1 & \text{segment } 2 & \text{segment } L \end{array}$$

We will assume that $\boldsymbol{u}_1$ was transmitted and that $\boldsymbol{R}$ was received by the second decoder. By assumption, the components of $\boldsymbol{R}$ are independent Gaussian random variables with means α_{ij} and variances σ_2^2. For convenience in this derivation, we normalize so that $\sigma_2^2=1$.

Define the new sets of random variables X_i and Y_i, $i=1,2,\cdots,L$, as

X_i = minimum component of $\boldsymbol{R}$ in segment i
Y_i = maximum component of $\boldsymbol{R}$ in segment i.

Using the union bound, the probability of error for decoder 2, given that $\boldsymbol{u}_1$ was transmitted, can be upper bounded as

$$P_{2/\boldsymbol{u}_1} \leqslant \sum_{i=1}^{L-1} P[Y_i > X_{i+1}].$$

To evaluate $P[Y_i > X_{i+1}]$ we note that

$$P[Y_i > X_{i+1}] = \int_{-\infty}^{\infty}\int_{-\infty}^{y_i} f_{Y_iX_{i+1}}(y_i, x_{i+1})\, dx_{i+1}\, dy_i$$

where $f_{YX}(\cdot,\cdot)$ is the joint pdf of Y and X.

Since the components of the noise are statistically independent, X_{i+1} and Y_i are statistically independent random variables. Then

$$P[Y_i > X_{i+1}] = \int_{-\infty}^{\infty} f_{Y_i}(y_i)\int_{-\infty}^{y_i} f_{X_{i+1}}(x_{i+1})\, dx_{i+1}\, dy_i = \int_{-\infty}^{\infty} f_{Y_i}(y_i) F_{X_{i+1}}(y_i)\, dy_i$$

where $F_X(\cdot)$ is the cdf of X and $f_Y(\cdot)$ is the marginal pdf of Y. But

$$\begin{aligned} F_{X_{i+1}}(z) &= 1 - P[X_{i+1} > z] \\ &= 1 - P\ [\text{all components of } \boldsymbol{R} \text{ in } (i+1)\text{th segment} > z] \\ &= 1 - \prod_{j=1}^{p_{i+1}} (1-\Phi(z-\alpha_{i+1,j}))^{n_{i+1,j}} \end{aligned}$$

where

$$\Phi(\gamma) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\gamma} e^{-(1/2)x^2}\, dx.$$

To find the marginal pdf of Y_i, we first note

$$\begin{aligned} F_{Y_i}(z) &= P[Y_i \leqslant z] \\ &= P_i\ [\text{all components of } \boldsymbol{R} \text{ in } i\text{th segment} \leqslant z] \\ &= \prod_{j=1}^{p_i} [\Phi(z-\alpha_{ij})]^{n_{ij}}. \end{aligned}$$

Differentiating to obtain the pdf we obtain

$$f_{y_i}(z) = \sum_{k=1}^{p_i} n_{ik}\frac{\phi(z-\alpha_{ik})}{\Phi(z-\alpha_{ik})}\prod_{j=1}^{p_i}[\Phi(z-\alpha_{ij})]^{n_{ij}}$$

where $\phi(z)=(1/\sqrt{2\pi})\exp^{-(1/2)z^2}$. Combining these results we obtain

$$P_{2/\boldsymbol{u}_1} = \sum_{i=1}^{L-1}\int_{-\infty}^{\infty}\sum_{k=1}^{p_i} n_{ik}\frac{\phi(z-\alpha_{ik})}{\Phi(z-\alpha_{ik})}\prod_{j=1}^{p_i}(\Phi(z-\alpha_{ij}))^{n_{ij}} \cdot \left[1-\prod_{l=1}^{p_i+1}(1-\Phi(z-\alpha_{i+1,l}))^{n_{i+1,l}}\right] dz.$$

Since all codewords are sent with equal *a priori* probability, the overall probability of error for the second decoder (for Variant I codes) is given as

$$P_2 = \frac{1}{M_1M_2}\sum_{i=1}^{M_1M_2} P_{2/\boldsymbol{u}_i}.$$

But since all codewords differ only in the order in which the symbols are transmitted, by reordering the symbols, we can convert any codeword to $\boldsymbol{u}_1$. Then the upper bound for the error probability assuming $\boldsymbol{u}_1$ was transmitted is also an upper bound for $P_{2/\boldsymbol{u}_i}$ for every i. Thus the upper bound for $P_{2/\boldsymbol{u}_1}$ is also an upper bound for P_2.

Variant II Codes

For Variant II codes, we will only require the second decoder to assign signs to components in segments 2 through L. That is, from the standpoint of assigning signs, we will treat all the components in the first segment as if they were all zeros. This strategy of course reduces the rate of information to the second decoder.

Using the notation introduced for Variant I codes and assuming $\boldsymbol{u}_1$ was transmitted (which has all positive signs), a union bound for $P_{2/\boldsymbol{u}_1}$ yields

$$P_{2/\boldsymbol{u}_1} \leqslant \sum_{i=2}^{L-1} P(Y_i > X_{i+1}) + P(E_2).$$

Here E_2 is the event that $Z_1 > X_2$, where Z_1 is the maximum of the absolute value of the components of $\boldsymbol{R}$ in segment 1. We now find that

$$P(E_2) = P[Z_1 > X_2] = \int_{-\infty}^{\infty} f_{Z_1}(z) F_{X_2}(z)\, dz$$

where

$$F_{X_2} = \prod_{j=1}^{p_2}(\Phi(z-\alpha_{2j}))^{n_{2j}}.$$

To find the pdf of Z_1, we first note that

$$\begin{aligned} F_{Z_1}(z) &= P\ [\text{magnitude of all components in segment } 1 < z] \\ &= \prod_{j=1}^{p_1}[\Phi(z-\alpha_{1j}) - \Phi(-z-\alpha_{1j})]^{n_{1j}}. \end{aligned}$$

Differentiating to obtain the pdf,

$$f_{Z_1}(z) = \sum_{l=1}^{p_1} n_{1l}\frac{(\phi(z-\alpha_{1l})+\phi(z+\alpha_{1l}))}{(\Phi(z-\alpha_{1l})-\Phi(-z-\alpha_{1l}))} \cdot \prod_{j=1}^{p_1}[\Phi(z-\alpha_{1j})-\Phi(-z-\alpha_{1j})]^{n_{1j}}.$$

Combining these, we obtain

$$P(E_2) = \int_{-\infty}^{\infty}\sum_{l=1}^{p_1} n_{1l}\frac{(\phi(z-\alpha_{1l})+\phi(z+\alpha_{1l}))}{(\Phi(z-\alpha_{1l})-\Phi(-z-\alpha_{1l}))} \cdot \prod_{j=1}^{p_1}[\Phi(z-\alpha_{2j})]^{n_{2j}}(\Phi(z-\alpha_{1j})-\Phi(-z-\alpha_{1j}))^{n_{1j}}\, dz.$$

Finally we obtain the upper bound for $P_{2/u1}$ and thus P_2 as

$$P_2 \leq \sum_{i=2}^{L-1} \int_{-\infty}^{\infty} \sum_{k=1}^{p_i} n_{ik} \frac{\phi(z-\alpha_{ik})}{\Phi(z-\alpha_{ik})} \prod_{j=1}^{p_i} (\Phi(z-\alpha_{ij}))^{n_{ij}} \cdot \left[1 - \prod_{l=1}^{p_{i+1}} (1-\Phi(z-\alpha_{i+1,l}))^{n_{1+i,l}}\right] dz$$

$$+ \int_{-\infty}^{\infty} \sum_{l=1}^{p_1} n_{1l} \frac{(\phi(z-\alpha_{1l})+\phi(z+\alpha_{1l}))}{(\Phi(z-\alpha_{1l})-\Phi(-z-\alpha_{1l}))} \cdot \prod_{j=1}^{p_1} (\Phi(z-\alpha_{2j}))^{n_{2j}} (\Phi(z-\alpha_{1j})-\Phi(-z-\alpha_{1j}))^{n_{1j}}\, dz.$$

References

[1] D. Slepian, "Permutation modulation," *Proc. IEEE*, Mar. 1965, pp. 228–236.

[2] T. M. Cover, "Broadcast channels," *IEEE Trans. Inform. Theory*, vol. IT-18, no. 1, Jan. 1972, pp. 2–14.

[3] A. D. Wyner, "Recent results in the Shannon theory," *IEEE Trans. Inform. Theory*, vol. IT-20, no. 1, Jan. 1974, pp. 2–10.

[4] P. P. Bergmans, "A simple converse for broadcast channels with additive white Gaussian noise," *IEEE Trans. Inform. Theory*, vol. IT-20, no. 2, Mar. 1974, pp. 279–280.

[5] ——, "Random coding theorem for broadcast channels with degraded components," *IEEE Trans. Inform. Theory*, vol. IT-19, no. 2, Mar. 1973, 197–207.

[6] C. Heegard, "Permutation modulation for the two-user broadcast channel," M.S. dis., Dep. of Electrical and Computer Engineering, Univ. Massachusetts, Amherst, June 1976.