

# An Algorithm for Source Coding

J. PIETER M. SCHALKWIJK, MEMBER, IEEE

**Abstract**—We derive a simple algorithm for the ranking of binary sequences of length  $n$  and weight  $w$ . This algorithm is then used for source encoding a memoryless binary source that generates 0's with probability  $q$  and 1's with probability  $p = 1 - q$ .

## I. INTRODUCTION

ASSUME a memoryless binary information source that generates 0's with probability  $q$  and 1's with probability  $p = 1 - q$ . According to Shannon's noiseless coding theorem [1], the source information can be encoded using on the average  $H(p)$  bits per source digit. The quantity  $H(p) = -p \log_2 p - (1 - p) \log_2 (1 - p)$  is called the entropy of the information source.

The first answer to the question as to how to optimally source encode  $M$  messages with probabilities  $p_1, p_2, \dots, p_M$  into a sequence of bits was given by Huffman [2]. (See also Abramson [3].) In Huffman coding the longest sequence of bits is used to encode the least probable message, while shorter sequences are used for more probable messages, thus minimizing the average number of bits per source symbol. Huffman coding could be applied to our memoryless binary information source. Treating the source digits one at a time, the Huffman procedure would assign one bit to each source digit. By encoding  $n$  source digits at a time, i.e.,  $M = 2^n$  possible messages, and letting  $n \rightarrow \infty$ , the average number of bits per source digit approaches  $H(p)$ . Although the Huffman procedure is optimum for each  $n$ , it does require a relatively large amount of computation for this particular application.

The Elias block-to-variable-length source coding algorithm [4] has been analyzed in detail by Jelinek [5]. A sequence of source digits is represented by a subinterval of the interval  $(0,1)$ . This subinterval, the length of which is equal to the probability of the corresponding source sequence, is constructed as follows. Start with the complete  $(0,1)$  interval. If the source generates a 0, the lower  $q$ th fraction of the current subinterval is retained; alternatively, if the source generates a 1, the upper  $p$ th fraction of the current subinterval is retained. The source encoding proceeds as follows. As soon as the remaining subinterval defining the source sequence lies in either  $(0, \frac{1}{2})$  or  $(\frac{1}{2}, 1)$ , a 0 or a 1 is generated, respectively. Suppose the subinterval defining the source sequence is in  $(0, \frac{1}{2})$ , i.e., the first source encoded digit is a 0, then if the subinterval defining the source sequence lies in either  $(0, \frac{1}{4})$  or  $(\frac{1}{4}, \frac{1}{2})$ ,

a 0 or a 1 is generated, respectively, etc. If the source sequence is  $n$  digits long, then for  $n \rightarrow \infty$  the average number of bits per source digit approaches  $H(p)$ . For large  $n$  we expect roughly  $qn$  zeros and  $pn$  ones. The length of the remaining subinterval will then be equal to

$$q^{qn} p^{pn} = 2^{-nH(p)}. \quad (1)$$

Note that size of the subinterval decreases exponentially with  $n$ . Hence we will have to divide the source sequence into relatively short blocks to prevent errors resulting from limited accuracy when implementing this algorithm on a digital computer. When we restrict the block length  $n$ , the data reduction will fall short of  $H(p)$  bits per source digit.

The algorithm to be discussed in this paper is basically a variable-length-to-block source coding algorithm.<sup>1</sup> The algorithm is asymptotically optimum in the sense that now we have on the average  $[H(p)]^{-1}$  source digits per bit put out as the size  $k$  of the output block approaches infinity. The amount of computation required by our algorithm is comparable to the amount of computation required by the Elias algorithm. The implementation of our algorithm seems slightly more transparent. Later in the paper more will be said as to the comparison of the latter two algorithms. The source-encoding algorithm discussed here is based on a theorem concerning the ranking of binary sequences of length  $n$  having a weight  $w$ , i.e., sequences with  $w$  1's. This theorem will be derived in the next section.

## II. RANKING OF FIXED WEIGHT SEQUENCES

Consider the set  $T(n,w)$  of binary sequences  $t = (t_1, t_2, \dots, t_n)$  of length  $n$  and weight  $w$ , where  $0 \leq w \leq n$ . Let the weight  $w_k$  of the sequence  $(t_k, t_{k+1}, \dots, t_n)$  be

$$w_k = \sum_{i=k}^n t_i, \quad (2)$$

where  $t_i \in \{0,1\}$ ,  $i = 1, 2, \dots, n$ . Note that  $w_1 = w$  is the weight of  $t$ . Then we have the following theorem.<sup>2</sup>

*Theorem 1:* The binary sequences  $t \in T(n,w)$  of length  $n$  and weight  $w$  can be ranked according to

$$i(t) = \sum_{k=1}^n t_k \binom{n-k}{w_k}, \quad (3)$$

where  $\binom{n}{w} = 0$  for  $w > n$  and  $0 \leq i(t) \leq \binom{n}{w} - 1$ .

*Proof:* The proof is by induction on the length  $n$  of

<sup>1</sup> One of the reviewers pointed out that with a slight modification our algorithm can be used for block-to-variable-length source coding, like Elias' algorithm.

<sup>2</sup> It was pointed out by the same reviewer that ranking of this type is known to researchers working in the areas of sorting and searching. See, for example [6].

Manuscript received December 28, 1970; revised May 19, 1971. This research was supported by the U.S. Navy under contract N000123-70-C-15231.

The author is with the Department of Applied Physics and Information Science, University of California at San Diego, La Jolla, Calif. 92037.

the sequence  $t$ . For  $n = 1$  we can have either  $w = 0$  or  $w = 1$ . If  $w = 0$  the only allowed sequence is  $t = 0$  and (3) gives  $i(0) = 0$ . For  $w = 1$  the only allowed sequence is  $t = 1$  and (3) gives  $i(1) = \binom{0}{1} = 0$ . Assume that the theorem holds for each  $T(n, w)$ ,  $0 \leq w \leq n$ , with  $n \leq N$ . We will then prove that the theorem holds for  $T(N + 1, w)$ ,  $0 \leq w \leq N + 1$ . Consider the set  $T_0(N + 1, w) = \{t \in T(N + 1, w); t_1 = 0\}$  of all  $t \in T(N + 1, w)$  that have  $t_1 = 0$ . By the inductive hypothesis we can order  $T_0(N + 1, w)$  according to

$$i_0(t) = \sum_{k=1}^N t_{k+1} \binom{N-k}{w_{k+1}} = \sum_{k=2}^{N+1} t_k \binom{N+1-k}{w_k}.$$

Now consider the set  $T_1(N + 1, w) = \{t \in T(N + 1, w); t_1 = 1\}$  of all  $t \in T(N + 1, w)$  that have  $t_1 = 1$ . By the inductive hypothesis we can order  $T_1(N + 1, w)$  according to

$$i_1(t) = \sum_{k=1}^N t_{k+1} \binom{N-k}{w_{k+1}} = \sum_{k=2}^{N+1} t_k \binom{N+1-k}{w_k}.$$

Finally, the number of elements of  $T_0(N + 1, w)$  is  $\binom{N}{w}$  and  $0 \leq i_0(t) \leq \binom{N}{w} - 1$ . So, the total set  $T(N + 1, w)$  can be ordered according to

$$i(t) = (1 - t_1)i_0(t) + t_1 \left[ \binom{N}{w} + i_1(t) \right] \\ = \sum_{k=1}^{N+1} t_k \binom{N+1-k}{w_k}. \quad \text{Q.E.D.}$$

As an example, consider the sequence  $t = (010100)$  with four 0's and two 1's in the set  $T(6, 2)$ . According to Theorem 1 we can find the rank  $i(t)$  of this sequence  $t$  using the blocked array of binomial coefficients from Pascal's triangle (see Fig. 1). Start at the lower left-hand corner of the array, i.e., at  $\binom{6}{2} = 15$  in Fig. 1. The leftmost digit of the sequence  $t = (010100)$  is a 0. Move one step in the  $X$  direction, i.e., toward 10. The next digit is a 1. Move one step in the  $Y$  direction, i.e., toward 4, and record the number at a single step in the  $X$  direction from the current starting point 10, i.e., record 6. The next digit is a 0. Move one step in the  $X$  direction, i.e., toward 3. The next digit is a 1. Move one step in the  $Y$  direction, i.e., toward 1, and add the number at a single step in the  $X$  direction from the current starting point 3, giving  $6 + 2 = 8$ . The last two digits are 0's leading to no more additions. So, the result is  $i(010100) = 8$ .

Given the rank  $i(t) = 8$ , the sequence  $t$  can be found using the following reconstruction algorithm. Start again at the lower left-hand corner of the coding array, i.e., at  $\binom{6}{2} = 15$  in Fig. 1. The rank 8 of the sequence  $t$  is less than the number 10, at a single step in the  $X$  direction from the current starting point 15. Move one step in the  $X$  direction, i.e., toward 10, and record a 0. The rank 8 of the sequence  $t$  is not less than the number 6, at a single step in the  $X$  direction from the current starting point 10. Move one step in the  $Y$  direction, i.e., toward 4, subtract

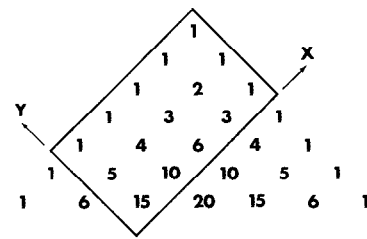


Fig. 1. Pascal's triangle gives an ordering of source sequences.

the number 6 used in the comparison from the rank 8 of the sequence  $t$  giving a new rank  $8 - 6 = 2$  and record a 1 giving 01. The current rank 2 of the sequence  $t$  is less than the number 3, at a single step in the  $X$  direction from the current starting point 4. Move one step in the  $X$  direction i.e., toward 3, and record a 0 giving 010. The current rank 2 of the sequence  $t$  is not less than the number 2, at a single step in the  $X$  direction from the current starting point 3. Move one step in the  $Y$  direction, i.e., toward 1, subtract the number 2 used in the comparison from the current rank 2 of the sequence  $t$  giving a new rank  $2 - 2 = 0$  and record a 1 giving 0101. The current rank of the sequence  $t$  is now 0. Thus, the last two steps are taken in the  $X$  direction, resulting in the desired sequence 010100.

The following table gives the complete set  $T(6, 2)$  with the sequences ranked according to (3).

0.	000011
1.	000101
2.	000110
3.	001001
4.	001010
5.	001100
6.	010001
7.	010010
8.	010100
9.	011000
10.	100001
11.	100010
12.	100100
13.	101000
14.	110000

The result  $i(t)$  produced by the ranking algorithm can be expressed in any desired number system. The binary number system would be used for an output in 0's and 1's. Any arbitrary member  $t \in T(n, w)$  would then be expressed as a sequence of  $k = \lceil \log_2 \binom{n}{w} \rceil$  bits, i.e., the smallest integer equal to or greater than  $\log_2 \binom{n}{w}$ . In this example 010100 would be encoded as 1000. The number of input digits per bit output, i.e.,  $r = n / \lceil \log_2 \binom{n}{w} \rceil$ , approaches  $[H(w/n)]^{-1}$  as  $n \rightarrow \infty$ . This can easily be shown by substituting Stirling's asymptotic result, i.e.,  $n! \approx (2\pi n)^{1/2} n^n e^{-n}$ , for the factorials in the binomial coefficient  $\binom{n}{w}$ .

Fixed weight sequences with an alphabet size  $q$  can be ranked according to a generalization of Theorem 1. Let  $T(n, w^0, w^1, \dots, w^{q-1})$ , be the set of sequences  $t = (t_1, t_2, \dots, t_n)$  of length  $n$ , where  $w^d$  components of  $t$  are equal to  $d$ ,  $d = 0, 1, \dots, q - 1$ . Let

$$w_k^d = \sum_{i=k}^n \delta(t_i, d), \quad (4)$$

where  $d = 0, 1, \dots, q - 1$ , and

$$\delta(x, y) = \begin{cases} 0, & x \neq y \\ 1, & x = y. \end{cases}$$

Note again that  $w_1^d = w^d$ ,  $d = 0, 1, \dots, q - 1$ . Now Theorem 1 generalizes as follows.

*Theorem 2:* The  $q$ -ary sequences  $t \in T(n, w^0, w^1, \dots, w^{q-1})$  of length  $n$  and with a weight distribution  $w^0, w^1, \dots, w^{q-1}$  can be ranked according to

$$i(t) = \sum_{k=1}^n \sum_{d=0}^{t_k-1} \left\{ (n-k)! / \left[ (w_k^d - 1)! \prod_{\substack{i=0 \\ i \neq d}}^{q-1} (w_k^i!) \right] \right\}, \quad (5)$$

where  $\sum_{d=0}^{-1} = 0$ , and  $(-1)! = 0$ .

The proof of Theorem 2 is identical to the proof of Theorem 1 and will hence not be given. An implementation similar to the ranking algorithm for Theorem 1 would use a  $q$ -dimensional array of multinomial coefficients  $n! / (w^0! w^1! \dots w^{q-1}!)$ , where  $w^0 + w^1 + \dots + w^{q-1} = n$ .

In this section we considered the ranking of fixed weight sequences of  $n$  digits. In the next section these results will be applied to the encoding of memoryless binary (discrete) information sources.

### III. VARIABLE-LENGTH-TO-BLOCK CODING

Consider a binary source generating independent digits. The probability of a 0 being generated is  $q$  and, hence, the probability of a 1 being generated is  $p = 1 - q$ . We proceed as before taking an array of size  $(qn + 1) \times (pn + 1)$  from Pascal's triangle (Fig. 2). When the source generates a 0 we take a unit step in the  $X$  direction, otherwise, when the source generates a 1 we take a unit step in the  $Y$  direction. If each source block of length  $n$  has exactly  $pn$  1's, as in the previous section, the source digits would specify a two-dimensional random walk in Fig. 2 that always terminates in point  $A$ . So we would always be able to encode each of our  $\binom{n}{pn}$  possible source blocks of length  $n$ . However, in the present case  $pn$  is only the expected number of 1's. So, suppose that our source sequence has many 1's in its initial part. Then the array boundary may be reached early, for example, at point  $B$  in Fig. 2. At this point no new source digits can be introduced as another 1 would take us outside of the coding array. So, if point  $B$  has coordinates  $(x, pn)$ , then only  $k = x + pn$  source digits have been introduced. We can now add  $qn - x$  dummy 0's to the source sequence. The resulting sequence has exactly  $qn$  0's and  $pn$  1's and is encoded as before. Suppose, instead, that our source sequence has many 0's in its initial part. Then the boundary may be reached early too, for example, at point  $C$  in Fig. 2. At this point no new source digits can be introduced as another 0 would take us outside of the coding array. So, if point  $C$  has coordinates  $(qn, y)$  then only  $k = qn + y$  source digits could have been introduced. We can now add  $pn - y$  dummy 1's to the source sequence. The resulting sequence has again  $qn$  0's and  $pn$  1's and is encoded as before. Note that point  $A$  (Fig. 2) can never be reached before the boundary is encountered. Hence the longest

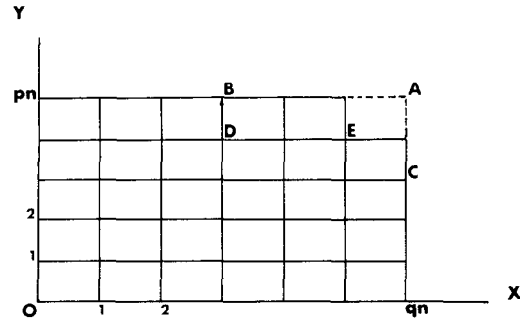


Fig. 2. Random walk in coding array.

source sequence that can be encoded as one unit is  $n - 1$  binary digits long. In general, for a  $q$ -ary source, the longest sequence is  $n - q + 1$  digits long.

For binary source blocks, as in the previous section, with exactly  $pn$  1's we obtain

$$r = n / \left\lceil \log_2 \binom{n}{pn} \right\rceil \quad (6)$$

source digits per bit output, which is asymptotically equal to  $[H(p)]^{-1}$ . We have seen before that with independently generated binary source digits there is a finite probability of the source sequence specifying a two-dimensional random walk that hits the coding array boundary early. This results in a number  $k$  of encoded source digits that can be smaller than  $n$  and, thus, the number of source digits per bit output falls short of  $[H(p)]^{-1}$ . So far we only know that  $pn \leq k \leq n - 1$  for  $0 < p < \frac{1}{2}$ . The lower bound on  $k$  results if the source sequence starts with  $pn$  1's. The upper bound for  $k$  results if the first  $n - 2$  source digits contain exactly  $qn - 1$  0's and  $pn - 1$  1's, i.e., if the random walk passes through point  $E$  in Fig. 2. The array boundary will then be reached on either of the two boundary points adjacent to  $A$ .

We will now consider the statistical behavior of the number of source digits  $k$  corresponding to an encoded block that can take on  $\binom{n}{pn}$  different possible values. What is the probability of the source sequence specifying a random walk that gets absorbed at the top boundary point  $B$  of the coding array (Fig. 2)? First observe that if the random walk gets absorbed at point  $B$ , it has to come from point  $D$ , otherwise it would have been previously absorbed. Let point  $B$  have coordinates  $(x, pn)$ . The number of paths from the origin to point  $D$  is the number of ways of choosing  $x$  objects out of  $x + pn - 1$  objects, i.e.,  $\binom{x+pn-1}{x}$ . The chosen objects here are the locations of the  $x$  steps in the  $X$  direction. Each path terminating in  $B$  has  $x$  steps in the  $X$  direction and  $pn$  steps in the  $Y$  direction. Hence each such path has a probability  $q^x p^{pn}$ . So the total probability of the two-dimensional random walk being absorbed at point  $B$  is

$$\Pr(B) = \binom{x + pn - 1}{x} q^x p^{pn}. \quad (7)$$

Similarly, for a point  $C = (qn, y)$  on the right-hand boundary of the array (Fig. 2) we get

$$\Pr(C) = \binom{qn-1+y}{y} q^{qn} p^y. \quad (8)$$

For point  $B$  we have  $k = x + pn$  and for point  $C$  we have  $k = qn + y$ . Hence,

$$\begin{aligned} \Pr(K \leq k) &= \sum_{x=0}^{k-pn} \Pr(B) + \sum_{y=0}^{k-qn} \Pr(C) \\ &= p^{pn} \sum_{x=0}^{k-pn} \binom{x+pn-1}{x} q^x \\ &\quad + q^{qn} \sum_{y=0}^{k-qn} \binom{qn-1+y}{y} p^y, \end{aligned} \quad (9)$$

where  $0 \leq k \leq n-1$ , and the summations are zero if the upper limit is negative. The distribution function (9) for the number of source digits  $k$  has been plotted in Figs. 3 and 4. From Fig. 3 we see that as the parameter  $n$  increases, the probability weight shifts toward  $k/n = 1$ . Fig. 4 shows that an increase in the parameter  $p < \frac{1}{2}$  also causes a shift of the probability weight toward  $k/n = 1$ .

An important statistic that follows easily from (7) and (8) is the average number of source digits  $\bar{k}$  per encoded block. We have

$$\begin{aligned} \bar{k} &= \sum_{x=0}^{qn-1} (x+pn) \Pr(B) + \sum_{y=0}^{pn-1} (qn+y) \Pr(C) \\ &= n \left[ p^{pn+1} \sum_{x=0}^{qn-1} \binom{x+pn}{x} q^x \right. \\ &\quad \left. + q^{qn+1} \sum_{y=0}^{pn-1} \binom{qn+y}{y} p^y \right] \\ &= n \left[ 1 - \binom{n}{pn} p^{pn} q^{qn} \right]. \end{aligned} \quad (10)$$

The last equality in (10) follows easily if we realize that the terms  $\binom{x+pn}{x} q^x p^{pn+1}$  and  $\binom{qn+y}{y} q^{qn+1} p^y$ , within the first pair of square brackets, represent the probability of a two-dimensional random walk in a rectangle of size  $(qn+1) \times (pn+1)$  reaching points on the boundaries  $y = pn+1$  and  $x = qn+1$ , respectively. Hence the total expression within the first pair of square brackets is the probability of the random walk not going through the point  $(qn, pn)$ , i.e.,  $1 - \binom{n}{pn} p^{pn} q^{qn}$ . Using Stirling's result in (10) we get the following asymptotic equation for the average number  $\bar{k}$  of source digits:

$$\bar{k} \approx n[1 - (2\pi pqn)^{-1/2}]. \quad (11)$$

In other words,  $\bar{k}$  is asymptotically equal to  $n$  and, hence, the data reduction  $r = \bar{k}/[\log_2 \binom{n}{pn}]$  approaches  $[H(p)]^{-1}$  for large  $n$ .

Finally, note that for  $q \gg p$  the coding array in Fig. 2

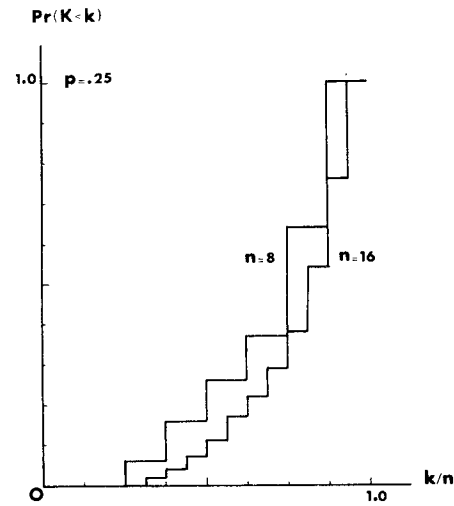


Fig. 3. Distribution function of the length of the source sequence,  $n$  being the parameter.

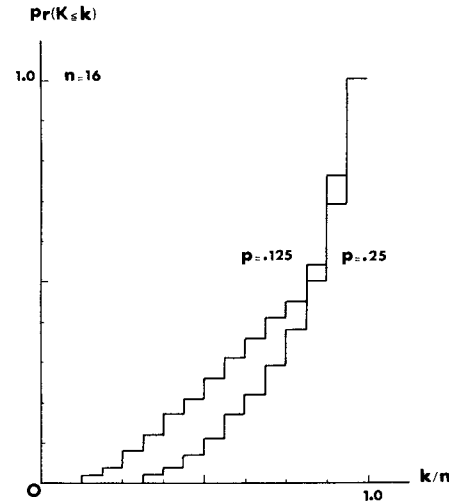


Fig. 4. Distribution function of the length of the source sequence,  $p$  being the parameter.

becomes an  $(n-1) \times 1$  rectangle and our coding scheme is then equivalent to run-length encoding. It is a well-known fact that run-length coding is quite efficient for  $q \gg p$ .

#### IV. CONCLUSIONS

In the Introduction our variable-length-to-block source coding algorithm was compared to Elias' block-to-variable-length source coding algorithm. We want to point out that with a slight modification<sup>1</sup> our algorithm can also be used for block-to-variable-length source coding. A code-word prefix could state the number of 1's,  $w$ , in the source block, and the suffix could be the encoding of the block in terms of the Pascal triangle appropriate to the relative frequency  $w/n$ .

Finally, also in the Introduction, we alluded to a comparison of Elias' algorithm and our algorithm as far as computational complexity is concerned. It was discovered that Elias' algorithm can be implemented using the follow-

ing triangle, that in a sense is dual to Pascal's triangle:

$$\begin{array}{cccc}
 & & 1 & \\
 & & p & q \\
 & p^2 & pq & q^2 \\
 p^3 & p^2q & pq^2 & q^3
 \end{array}$$

A detailed comparison of the two algorithms is presently being made at the University of California, San Diego.

REFERENCES

[1] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379-423, 623-656, 1948.  
 [2] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proc. IRE*, vol. 40, pp. 1098-1101, Sept. 1952.  
 [3] N. Abramson, *Information Theory and Coding*. New York: McGraw-Hill, 1963.  
 [4] P. Elias, unpublished result.  
 [5] F. Jelinek, "Buffer overflow in variable length coding of fixed rate sources," *IEEE Trans. Inform. Theory*, vol. IT-14, pp. 490-501, May 1968.  
 [6] E. F. Beckenbach, *Applied Combinatorial Mathematics*. New York: Wiley, 1964, sect. 1.10.

# On the Existence of Group Codes for the Gaussian Channel

EZIO BIGLIERI AND MICHELE ELIA

**Abstract**—A number of theorems are given, which give insight into the algebraic structure of group codes for the Gaussian channel, as defined by Slepian. The problem of the existence of group codes for a given pair of defining parameters (i.e., number of codewords  $M$ , and dimension of the signal space  $n$ ) is partially solved.

I. INTRODUCTION

GROUP codes for the Gaussian channel have been introduced in a recent paper by Slepian [1], who first described their remarkable properties. In his paper, however, a number of questions are left unanswered, one of these being the problem of the existence of group codes for a given pair of defining parameters (i.e., the number of codewords  $M$  and the dimension of the signal space  $n$ ).

In this paper a number of theorems are given that partially solve this problem. The only question left unanswered concerns the existence of group codes for  $n$  odd and  $M$  odd but not prime; for this case, we show that additional constraints on  $n$  and  $M$  are needed, but we do not find them explicitly.

II. DEFINITIONS AND PROPERTIES

**Definition 1:** A pseudogroup code  $[M, n]$  is a set  $X = \{X_i\}_{i=1}^M$  of  $M$  unit vectors in Euclidean  $n$ -dimensional space  $E^n$  such that there is a set  $G^* = \{O_i^*\}_{i=1}^M$  of  $M$  orthogonal  $n$ -by- $n$  matrices for which

$$G^*X_i = X, \quad \forall X_i \in X. \tag{1}$$

**Definition 2:** A group code  $[M, n]$  is a pseudogroup code  $[M, n]$  spanning  $E^n$ .

Let now  $G^*$  be a multiplicative group of orthogonal matrices—the set  $X = G^*X_1$  ( $X_1 \in X$ ) is obviously a pseudogroup code. Generally, however,  $X$  is not a group code for all choices of the starting vector  $X_1$ ; only if the group  $G^*$  is real irreducible (see Appendix) does  $X$  span  $E^n$  for every  $X_1$  [1].

Moreover, the set  $G^*$  of Definition 1 is certainly a subset of the group  $G = \{O_i\}_{i=1}^g$  of all real orthogonal  $n$ -by- $n$  matrices such that  $GX_i = X$  ( $\forall X_i \in X$ ). So, we can substitute for Definition 1 the following definition.

**Definition 3:** We define a pseudogroup code  $[M, n]$  to be a set  $X = \{X_i\}_{i=1}^M$  of  $M$  unit vectors in Euclidean  $n$ -dimensional space  $E^n$  such that there is a group  $G = \{O_i\}_{i=1}^g$  of  $g$  orthogonal  $n$ -by- $n$  matrices for which

$$GX_i = X, \quad \forall X_i \in X.$$

In the conditions of Definition 3, we say that  $X$  is generated by  $G$ , and we think of  $G$  as a real representation of an abstract group ([3]–[5] and Appendix).

Let us now state two theorems that give insight into the algebraic structure of group codes.

**Theorem 1** [1]:

- i)  $g \geq M$  and  $g \leq M!$ ;
- ii) the subset  $H \subseteq G$  formed by the matrices carrying a given starting vector, say  $X_1$ , into itself (i.e.,  $HX_1 = \{X_1\}$ ) is a subgroup of  $G$ ;
- iii) if  $g > M$ , then  $M \mid g$ .

The second part of the first statement in Theorem 1 can

Manuscript received February 1, 1971; revised July 13, 1971.  
 The authors are with the Istituto di Elettronica e Telecomunicazioni, Politecnico di Torino, Turin, Italy.