DOCUMENT RESUME

ED 218 103 SE 038 104

AUTHOR Rice, Bart F.; Wilde, Carroll O.
TITLE Error Correcting Codes I. Applications of Elementary Algebra to Information Theory. Modules and Monographs in Undergraduate Mathematics and Its Applications Project. UMAP Unit 346.
INSTITUTION Education Development Center, Inc., Newton, Mass.
SPONS AGENCY National Science Foundation, Washington, D.C.
PUB DATE 79
GRANT SED-76-19615-A02
NOTE 36p.

EDRS PRICE MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS Answer Keys; *College Mathematics; *Communication Research; Higher Education; Information Science; *Information Theory; Instructional Materials; *Learning Modules; *Mathematical Applications; Problem Solving; Supplementary Reading Materials
IDENTIFIERS Coding; *Coding Theory

ABSTRACT

It is noted that with the prominence of computers in today's technological society, digital communication systems have become widely used in a variety of applications. Some of the problems that arise in digital communications systems are described. This unit presents the problem of correcting errors in such systems. Error correcting codes are developed as an application of linear algebra and finite field algebra. The models chosen for this module were selected for relative ease of comprehension. The material includes exercises and a model examination, with answers provided to all problems. (MP)

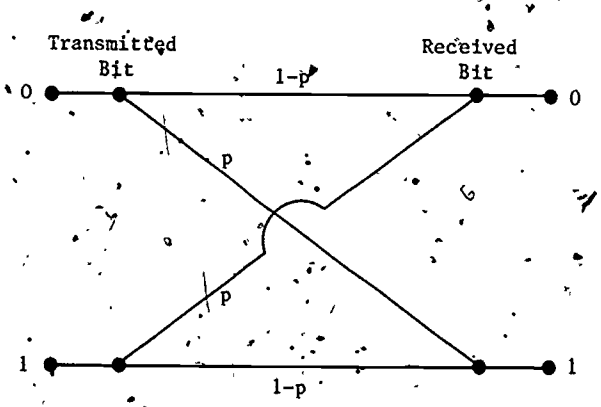
umap

UNIT 346

MODULES AND MONOGRAPHS IN UNDERGRADUATE MATHEMATICS AND ITS APPLICATIONS PROJECT

ERROR CORRECTING CODES I

by Bart F. Rice and Carroll O. Wilde



APPLICATIONS OF ELEMENTARY ALGEBRA TO INFORMATION THEORY

edc/umap / 55 chapel st. / newton, mass. 02160

ERROR CORRECTING CODES I

by

Bart F. Rice
Department of Defense
Washington, DC

and

Carroll O. Wilde
Department of Mathematics
Naval Postgraduate School
Monterey, CA

TABLE OF CONTENTS

Intermodular Description Sheet: UMAP Unit 346

Title: ERROR CORRECTING CODES I

Authors: Bart F. Rice
Department of Defense
Washington, DC 20755

Carroll O. Wilde
Department of Mathematics
Naval Postgraduate School
Monterey, CA. 93940

Classification: APPL ELEM ALG/INFORMATION THEORY

Suggested Support Materials:

Mac Lane, S. and Birkhoff, G. 1979. *A Survey of Modern Algebra*, Second Edition. New York: Macmillan.

Gilbert, W. 1976. *Modern Algebra with Applications*. New York: John Wiley and Sons.

Prerequisite Skills:

1. Ability to evaluate polynomials (see UMAP Unit 263).
2. Familiarity with binary addition and multiplication, and with the basic properties of the Galois field GF(2) (= Z_2).
3. Familiarity with fundamental concepts in the theory of vector spaces, in particular, with these concepts applied to the vector spaces $GF(2)^n$ over GF(2), for n a positive integer.
4. Ability to multiply and invert matrices over GF(2).
5. The basic concepts of a derivative and maximization of polynomial functions are useful, but not absolutely necessary for general comprehension.
6. Some familiarity with basic concepts of probability theory, including Bernoulli and binomial distributions, and statistical independence, is helpful. These concepts are used in an elementary way in this module, and it is hoped that this prerequisite will not deter you needlessly.

Output Skills:

1. Use simple repetition coding to decrease the probability of error in binary communication systems.
2. Apply the binary symmetric channel model to evaluate coding schemes, to find the improvement in the probability of error (or that of no error) in transmission.
3. Use Hamming codes to decrease the probability of error in binary communication channels (this includes use of the maximum likelihood decoding algorithm).
4. Find information rates for error correcting codes.

Other Related Units:

Cohen, S. *Aspects of Coding* (Unit 336)
Sherman, G. *A Double-Error Correcting Code* (Unit 337)

MODULES AND MONOGRAPHS IN UNDERGRADUATE MATHEMATICS AND ITS APPLICATIONS PROJECT (UMAP)

The goal of UMAP is to develop, through a community of users and developers, a system of instructional modules in undergraduate mathematics and its applications which may be used to supplement existing courses and from which complete courses may eventually be built.

The Project is guided by a National Steering Committee of mathematicians, scientists, and educators. UMAP is funded by a grant from the National Science Foundation to Education Development Center, Inc., a publicly supported, nonprofit corporation engaged in educational research in the U.S. and abroad.

PROJECT STAFF

Ross L. Finney	Director
Solomon Garfunkel	Associate Director/Consortium Coordinator
Felicia DeMay	Associate Director for Administration
Barbara Kelczewski	Coordinator for Materials Production
Paula M. Santillo	Administrative Assistant
Zachary Zevitas	Staff Assistant

NATIONAL STEERING COMMITTEE

W.T. Martin	M.I.T. (Chair)
Steven J. Brams	New York University
Llayron Clarkson	Texas Southern University
Ernest J. Henley	University of Houston
William Hogan	Harvard University
Donald A. Larson	SUNY at Buffalo
William F. Lucas	Cornell University
R. Duncan Luce	Harvard University
George Miller	Nassau Community College
Frederick Mosteller	Harvard University
Walter E. Sears	University of Michigan Press
George Springer	Indiana University
Arnold A. Strassenburg	SUNY at Stony Brook
Alfred B. Willcox	Mathematical Association of America

The Project would like to thank members of the UMAP Analysis and Computation Panel: Carroll O. Wilde, Chair, Naval Postgraduate School; Maurice D. Weir, Naval Postgraduate School; Roy B. Leipnik, University of California; Richard J. Allen, St. Olaf College; Louis C. Barrett, Montana State University, and; George Robert Blakley, Texas A & M University; and James Kaput of Southeastern Massachusetts University for their reviews, and all others who assisted in the production of this unit.

This material was prepared with the support of National Science Foundation Grant No. SED76-19615 A02. Recommendations expressed are those of the author and do not necessarily reflect the views of the NSF, nor of the National Steering Committee.

ERROR CORRECTING CODES I

1. INTRODUCTION

With the prominence of computers in today's technological society, digital communication systems have become widely used in a variety of applications. For example, data gathered by space probes must be transmitted back to earth, where the information can be processed for subsequent use. This example includes satellite pictures that are transmitted and processed digitally to obtain reconnaissance and scientific information. Data links between computers provide enormous gains in computer applications. Military command and control systems also provide a broad range of examples.

In digital communications, information is transmitted in the form of "binary messages," that is, strings of 0's and 1's which are coded in some way to convey information. For example, in transmitting a satellite photograph, suppose that the onboard instrument package can distinguish 64 gray levels from white to black, and that each level is identified by a number from 0 to 63. The particular string 101011, which is the binary representation of the decimal number 43, could be transmitted to indicate that the light intensity at a particular point in the picture is at level 43.

You may easily imagine some of the problems that arise in digital communication systems. Errors may occur in the original encoding of data and in the transmitter. Channels may be "noisy," so that bits are lost or distorted. Security is often a problem—there may be compelling reasons to deny information to some individuals who have access to our channels.

In this unit we study the problem of correcting errors in digital communication systems. Error correcting codes are developed as an application of linear algebra and finite field algebra. The models chosen for this unit were selected for relative ease of comprehension; more complex models will be presented in subsequent units.

2. THE BINARY SYMMETRIC CHANNEL

Suppose we wish to send a binary message through a noisy channel which may corrupt the message by changing one or more of the bits. One or more zeros may be inverted to ones, or ones inverted to zeros. A simple model of such a channel is called the binary symmetric channel (BSC), and is shown in Figure 1. In this model, a bit is inverted with probability p, regardless of whether it is a 0 or a 1, and the corruption of any bit is statistically independent of what happens to any other bit. We assume that $0 < p < \frac{1}{2}$.

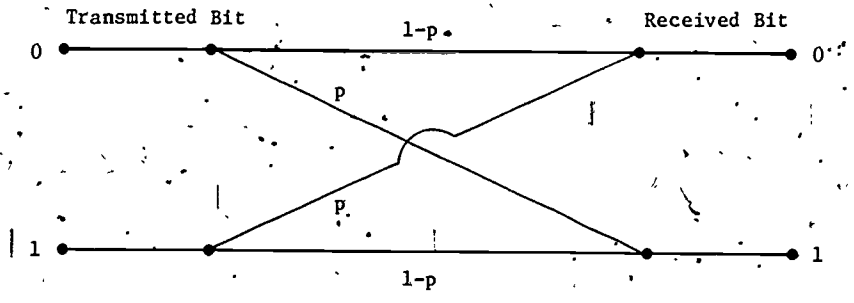


Figure 1. The binary symmetric channel (BSC). Transmitted bits are inverted with probability p.

We note that the BSC is not a good model for many real-life channels, although it does reasonably represent the errors in some computers during fast data transfers. The assumption of statistical independence is valid for certain systems in which encoded data are multiplexed, transmitted, and then demultiplexed before decoding.

Errors tend to occur in "bursts"—a relatively large number in a short time period, followed by a long error-free period. *Multiplexing* minimizes the damaging effects of bursts by selecting the successive bits in the transmitted signal cyclically from different codewords. Bursts of several errors in a relatively short span will then tend to corrupt bits in different codewords instead of being concentrated in a single word or two and rendering them unrecognizable. (See Appendix 2.)

Even though the BSC is an appropriate model for only certain real channels, we use it in this module for two primary reasons. The first, and more important reason, is that the BSC affords the most elementary discussion of the theory of error correcting codes available. The second is that despite its limitations, the BSC is the principal model by which code performance is evaluated, because it is the only one for which algebraic computations of code performance are tractable. Nevertheless, there are other models that are available and in current use, and we describe one briefly in Appendix 1.

Of course, it is desirable that a message be received correctly with high probability. For example, if the bit 1 signifies "by land" and 0 "by sea," an error in transmission could have serious consequences! In our first example we consider a scheme that substantially increases the probability of getting a single bit of information through correctly.

Example 1: (Repetition, with "majority" decoding). Suppose we wish to transmit a single bit of information, that is, a 0 or a 1. Instead of a single 0 we transmit the sequence 00000, and instead of 1 we send 11111. These two repetition strings are the *codewords* in our code. The receiver decodes the message by a simple majority vote. For example, if 10110 is received, the decoder decides by a 3 to 2 vote that 11111 was transmitted, hence the intended bit was 1. This decoding algorithm may produce

the correct result, but it could also produce an error, which would be the case here if 00000 had been transmitted and the channel had corrupted the 1st, 3rd and 4th bits. Under this scheme, the probability of a decoding error is given by:

$$\begin{aligned} P(E) &= \text{probability of 2 bits correct and 3 incorrect} \\ &\quad + \text{probability of 1 bit correct and 4 incorrect} \\ &\quad + \text{probability of 0 bits correct and 5 incorrect} \\ &= {}_5C_2(1-p)^2p^3 + {}_5C_1(1-p)p^4 + p^5 \\ &= p^3(10(1-p)^2 + 5(1-p)p + p^2) \\ &= p^3(6p^2 - 15p + 10). \end{aligned}$$

If we had simply transmitted the intended bit instead of using a code, the probability of an error would have been p. Using the code we have a new probability of error P(E), which depends on the original p. We tabulate a few values to show how these probabilities compare:

p	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45
P(E)	0.002	0.01	0.03	0.06	0.10	0.16	0.24	0.32	0.41

The table shows the gain we have made in reducing the probability of an error—by a factor of 10 when p = 0.1, for example. But we have paid a price for this improvement! For to transmit one bit of information we now need a block of five bits. In this case we would say that the *information rate*, or number of information bits divided by the number of message bits, is 1/5. We note also that our code will correct as many as 2 errors in a message. We have designed a 2-error correcting binary repetition code of block length 5, in which there are two possible codewords, 00000 and 11111.

Exercises

1. For a 1-error correcting binary repetition code of block length 3, with codewords 000 and 111,

   a. find a formula for P(E) in terms of p, as was done in Example 1;

   b. compare the values of P(E) and p for p = 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, and 0.45;

   c. find the information rate.

3. HAMMING CODES

In the repetition code we used redundancy to increase the probability that a message will be interpreted correctly. The general theory of error correcting codes addresses the question, "Can we add redundancy in such a way that the probability of error will be decreased to an acceptable level and the information rate will remain relatively high?" (The terms "acceptable" and "relatively high" are imprecise; they must be defined by the communication system designer in light of available equipment and transmission channels.)

In this section we introduce a class of codes in which redundancy is added more intelligently than it was in the simple repetition code. These codes provide a better trade-off balance between the decrease in the probability of error and the information rate loss.

Also in this section we apply some concepts and results from linear algebra to obtain practical error correcting codes. We perform our calculations over GF(2), the field with two elements 0 and 1, with binary addition and multiplication. If n is a positive integer, let $GF(2)^n$ denote the vector space whose elements are the n-tuples with entries 0 or 1, and whose scalar field is

GF(2). For example, with n = 4, the vectors (1,0,1,1) and (0,0,0,1) are two of the sixteen elements of $GF(2)^4$.

We may now offer a formal definition of a concept which we have already used loosely in describing the simple repetition code.

Definition. A *binary code* of block length n is simply a subset C of $GF(2)^n$. Each element of C is called a *codeword* of the code.

Elements of $GF(2)^n$, including codewords, may be represented as row vectors, as column vectors, or as "words," which are strings of 0's or 1's. Thus, the symbols

$$(1,0,1,1), \quad \begin{bmatrix}1\\0\\1\\1\end{bmatrix}, \quad \text{and } 1011$$

represent the same element of $GF(2)^4$. We shall be somewhat cavalier in passing between these representations, and between the terms "vector" and "word."

An early impetus to coding theory was provided by Richard W. Hamming,* when he developed what are now called *Hamming codes*. These are binary codes of block length 2^m-1, where m is a positive integer. In Examples 2 and 3 we describe the code and a corresponding decoding algorithm for the case m = 3.

Example 2 (A (7,4) Hamming code**). To obtain a (7,4) Hamming code C, we first form the $3\times(2^3-1)$, or 3×7, matrix whose columns are the binary representations of the

*R.W. Hamming did much of his pioneering work on coding theory at the Bell Telephone Laboratory, Murray Hill, New Jersey. He is currently a Professor of Computer Science at the Naval Postgraduate School, Monterey, California.

**The indefinite article is used because any permutation of the columns in the fundamental matrix H in (3.1) yields another (7,4) Hamming code.

integers from 1 to 7. Since these representations are, in order, 001, 010, 011, 100, 101, 110, 111, this matrix, which is called the *parity check matrix* of the code, is:

$$(3.1) \qquad H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

A quick inspection reveals that the rank of H over GF(2) is 3. Hence, the nullspace of H is a 4-dimensional subspace of $GF(2)^7$ (see Exercise 2), and this nullspace is the set we take for C. Thus, a vector x in $GF(2)^7$ is a codeword if and only if x satisfies the condition $Hx = 0$ (here 0 denotes a zero vector), when x is represented as a column vector. For example, if

$$x = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \quad \text{and} \quad y = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix},$$

then since

$$Hx = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad \text{and} \quad Hy = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix},$$

we have $x \in C$ and $y \notin C$. That is, x is a codeword and y is not.

Since C is a 4-dimensional vector space over GF(2), there are precisely $2^4 = 16$ codewords in the (7,4) Hamming code. So far we have seen only one element of C (the vector x that, as a word, is represented by the string 1010101) in the last display. But we know that 0000000 is also in C, since $H0 = 0$ by elementary properties of matrix multiplication. In Exercise 3 you will be challenged to find all the codewords in this code.

In Example 3 below we present a decoding algorithm for the (7,4) Hamming code of Example 2. In this code, four of the seven bits in a codeword may be designated as the information bits. The remaining three are called the

parity check bits or, simply, the check bits. In this case the information rate is 4/7. For a more general Hamming code, with m any positive integer, the parity check matrix H will have dimensions $m \times (2^m-1)$ and rank 2^m-m-1, and the information rate will be $(2^m-m-1)/(2^m-1)$.

Before we present Example 3, we offer several exercises and some additional discussion designed to get you better acquainted with the (7,4) Hamming codes. This material will help you understand the decoding algorithm in Example 3.

Exercises

2. Consider the following elements of $GF(2)^7$ (regarded as column vectors):

$$x_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad x_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad x_3 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \quad x_4 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

   a. Show that for $i = 1,2,3,4$, x_i is a codeword in the (7,4) Hamming code of Example 2. (That is, show that each $x_i \in C$, the nullspace of the matrix H in (3.1).)

   b. Show that the set $\{x_1,x_2,x_3,x_4\}$ is linearly independent in the vector space $GF(2)^7$ over the scalar field GF(2). (Hence the subspace C has dimension 4; this agrees with our earlier observation that the rank of H is 3.)

3. Use the result of Exercise 2 to list all 16 codewords in our (7,4) Hamming code.

Although we now have a list of the codewords, the approach used to obtain it in Exercises 2 and 3 is not particularly instructive. We offer a better technique using elementary matrix manipulation. We first introduce two additional 3×7 matrices:

$$H_1 = \left[\begin{array}{cccc|ccc} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right], \quad H_2 = \left[\begin{array}{cccc|ccc} 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{array}\right].$$

Then by matrix addition we have

$$H = H_1 + H_2,$$

and for any element x (represented as a column vector) in $GF(2)^7$ we have

$$Hx = H_1x + H_2x.$$

Therefore, if $x \in C$, the nullspace of H, then

$$H_1x + H_2x = 0,$$

$$H_1x = -H_2x.$$

But, since we are calculating in GF(2), we have

(3.2) $$H_1x = H_2x.$$

Now let $x_0, x_1, \ldots, x_6$ be the coordinates of x. Then, because of the blocks of zeros in H_1, H_2, we obtain the following reductions:

(3.3) $$H_1x = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}, \quad H_2x = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_4 \\ x_5 \\ x_6 \end{bmatrix}.$$

Substitute (3.3) into (3.2) to obtain

(3.4) $$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}.$$

Equation (3.4) can be solved to obtain x_4, x_5, x_6 in terms of x_0, x_1, x_2, x_3 by a matrix inversion. Since

(3.5) $$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

(see Exercise 4), we may multiply both sides of Equation (3.4) by this matrix to obtain

(3.6) $$\begin{bmatrix} x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$= \begin{bmatrix} x_1 + x_2 + x_3 \\ x_0 + x_2 + x_3 \\ x_0 + x_1 + x_3 \end{bmatrix} \quad \text{(after two matrix multiplications)}.$$

Now, the steps used to derive Equation (3.6) are all reversible (see Exercise 5). Hence,

$$\text{for } x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} \in GF(2)^7,$$

(3.7) $x \in C$ if and only if $\begin{cases} x_4 = x_1 + x_2 + x_3 \\ x_5 = x_0 + x_2 + x_3 \\ x_6 = x_0 + x_1 + x_3. \end{cases}$

We note that x_i, x_j, x_k may be taken as the check bits as long as the columns i, j, k form linearly independent vectors so that the 3×3 matrix consisting of these columns is invertible.

Exercises.

4. Verify that the two matrices (3.5) are indeed mutually inverse.

5. Show that if the coordinates of a vector in $GF(2)^7$ satisfy the condition in (3.7), then $Hx = 0$. (The verification may be accomplished by reversing the steps that led to Equation (3.6), or by performing the matrix operation needed to find Hx.)

6. Again list the 16 codewords (as in Exercise 3), this time using the condition (3.7).

7. For the word $x = 1000000$ in $GF(2)^7$ (you may want to represent x as a column vector),

   a. find Hx, and compare Hx with the individual columns of H (to which column does Hx correspond?);

   b. list the 16 elements in the coset $C+x$.

8. Carry out the instructions of both parts of Exercise 7 for the word $y = 0100000$.

By now we have a good understanding of the space C. We know that of the $2^7 = 128$ elements in $GF(2)^7$, exactly 16 are codewords; we know which ones they are, and we have a way to identify them (Condition (3.7)). We also have a good feeling for the coset structure: we know that there are eight cosets, each with 16 elements, and that the elements of any one of the cosets other than C itself can be obtained by choosing a fixed bit (or coordinate) position and inverting the bit in that position, for each element of C. (In Exercise 7 we chose the first coordinate, in Exercise 8 the second.)

We call special attention to Condition (3.7). This condition shows that while four of the seven bits in a codeword may be assigned as we wish (here the indicated bits are x_0, x_1, x_2, x_3), there is no choice in the remaining three. For if we assign values to x_0, x_1, x_2, x_3, then

the values of x_4, x_5, x_6 are then predetermined by (3.7). Thus, each 7-bit codeword contains only 4 information bits, which confirms our earlier observation that the information rate is 4/7.

We now present a decoding algorithm for our (7,4) Hamming code. This algorithm is called a *maximum likelihood decoder*, although the reason for this name will not become apparent until the discussion after the example.

Example 3 (Maximum likelihood decoder). Suppose that a certain 7-bit codeword c from our (7,4) Hamming code C in $GF(2)^7$ is transmitted over a noisy channel, and that the word $r = 1001010$ is received at the other end. If we represent r as a column vector, we may calculate Hr, for H the parity check matrix (3.1):

$$(3.8) \qquad Hr = H\begin{bmatrix}1\\0\\0\\1\\0\\1\\0\end{bmatrix} = \begin{bmatrix}0\\1\\1\end{bmatrix} \neq \begin{bmatrix}0\\0\\0\end{bmatrix}.$$

Since $Hr \neq 0$, the message recipient knows that $r \notin C$, and hence $r \neq c$, and therefore an error has been made in transmission. What word was originally sent?

Suppose we represent the received word r as the sum of the transmitted codeword c and an error e:

$$(3.9) \qquad r = c + e.$$

Now let $Hr = s$; then also $He = s$, because

$$s = Hr = He + Hc = He + 0 = He.$$

Therefore, while the message recipient does not know c or e, he or she does know that e is contained in the same coset as r, i.e., that $e \in C + r$. The problem of determining c therefore reduces to that of making an "intelligent" choice for e from the coset $C + r$.

In seeking the best choice of e in C + r, we first note that the column vector Hr in (3.8) coincides with the third column of H. But by elementary properties of matrix algebra (see also Exercises 7 and 8), we know also that the column vector given by the product

$$H\begin{bmatrix}0\\0\\1\\0\\0\\0\\0\end{bmatrix}$$

also equals the third column of H. Hence the word 0010000 is also in the coset C + r. Moreover, if we call the number of 1's in a word the *weight* of that word, then no single coset may contain more than one vector of weight one. (This fact can be seen from the results of Exercises 7 and 8 and the observation that, as vectors, the columns of H are all distinct.) The word we take from the coset C + r as our choice for the error when the received word is r = 1001010 is the word

(3.10) e = 0010000.

We now use Equation (3.9) to find our "best" guess for the original c:

$$r = c + e,$$

from which (since e + e = 0000000)

$$\begin{aligned} c &= r + e \\ &= 1001010 + 0010000 \\ &= 1011010. \end{aligned}$$

We note that our choice of c can be obtained by inverting the third bit in the received word r.

Now suppose that instead of a word (such as 1001010) from a coset of C, a codeword c' had been received. Then,

the error c - c' would be a codeword, so the choice of an error would be made from C itself. In this case we choose the zero word 0000000 for the error, that is, we assume that no error occurred, so we take c' itself as the choice of c. (The reason for this choice will be explained after this example.)

We summarize the procedure above as our decoding algorithm for a (7,4) Hamming code.

Maximum likelihood decoder. For a (7,4) Hamming code, when a 7-bit word r is received, we:

- represent r as a 7×1 column vector;
- find the column vector Hr by matrix multiplication over GF(2);
- test Hr to determine whether Hr is the zero vector, or a column of H;
- choose r to be c itself if Hr = 0, for then $r \in C$;
- choose c to be the codeword obtained from r by inverting the jth bit, if Hr is the jth column of H.

Exercises

9. For the (7,4) Hamming code of Example 2 with maximum likelihood decoder, decode each of the following received words r:
   a. 1101011;
   b. 0011110;
   c. 1010101;
   d. 0100000.

Two immediate questions arise in connection with Example 3. The first is: since we are using 7-bit codewords to convey 4 bits of information, which means an information rate of 4/7, or 0.57, what do we gain in

trade for this 43-percent loss? The second concerns the name of the algorithm—why is it called the "maximum likelihood" decoder?

We begin with the first of these questions. The basic response is that by using the code we reduce the probability of error in conveying each 4 bits of information. We shall measure this gain, as we measured the corresponding gain in Example 1, but we emphasize that in studying gains, a realistic user must never lose sight of the losses involved.

For an analysis of the gain made by using a (7,4) Hamming code we use the binary symmetric channel (BSC) model as in Example 1. In this case, however, it is more convenient to study the increase in the probability of "no error," P(N), instead of the decrease in the probability of error, P(E).

If we simply broadcast 4 bits of information across a BSC without using a code, then we would have $P(N) = (1-p)^4$. Using a (7,4) Hamming code with maximum likelihood decoder, we make no error precisely when either one or none of the 7 transmitted bits is corrupted. In this case we have

(3.11) P(N) = probability of 7 bits correct and 0 incorrect
+ probability of 6 bits correct and 1 incorrect

$$= (1-p)^7 + {}_7C_6(1-p)^6 p$$

$$= (1-p)^4(1+4p-11p^2+6p^3).$$

Thus, the polynomial $f(p) = 1 + 4p - 11p^2 + 6p^3$ provides a measure of the gain in the probability of no error.

Since $f(0) = f(1/2) = 1$, and an easy analysis of the derivative shows that $f'(p) > 0$ for $0 < p < 2/9$ and $f'(p) < 0$ for $2/9 < p < 1$, our best *percentage* gain in the probability of no error occurs at $p = 2/9$. Since $f(2/9) \approx 1.41$, this maximum gain is around 41 percent.

We tabulate the values of $(1-p)^4$, P(N), and the percentage gain for several values of p.

TABLE 1

Percent Gain in the Probability of No Error

(for selected values of p)

p	$(1-p)^4$	P(N)	Percent Gain (f(p)-1)
0.01	0.96	1.00	4
0.05	0.81	0.96	17
0.10	0.66	0.85	30
0.15	0.52	0.72	37
0.20	0.41	0.58	41
2/9	0.37	0.52	41
0.25	0.32	0.44	41
0.30	0.24	0.33	38
0.35	0.18	0.23	31
0.40	0.13	0.16	22
0.45	0.09	0.10	12

The table indicates that this coding scheme may very well be appealing, depending on requirements for accuracy and other factors, such as constraints on the information rate. In addition, it may be that this scheme would be more desirable for values of p that do *not* yield maximum percentage gain in the probability of no error. For example, if a high degree of accuracy is required, then the increase in the probability of no error from 0.96 to 0.998 (rounded to 1.00 in the table) for p = 0.01 could be attractive to the user, and well worth the trade in the information rate.

The second question indicated above also has an interesting answer. In the decoding algorithm of Example 3, there is a test to be made, namely a determination of

whether the received word r is a codeword or not. If $r \in C$, then the error vector e is also in C, and the basis of our choice of e = 0 for the error is that under certain conditions this choice maximizes the probability that the chosen vector is the error vector. Similarly, if $r \notin C$, the choice of the vector of weight one from the coset C + r maximizes the probability of the error vector (again, under certain conditions).

Let us explore these probabilities, again using the BSC model. When we transmit a 7-bit word over a noisy channel, the error vector component is 0 if the bit is correctly received, and 1 if the bit is inverted in the channel. Thus, the error vector may be regarded as the outcome of 7 repeated Bernoulli trials with probability p of a 1 at each trial and 1-p of a 0. Hence, for a an integer with $0 \le a \le 7$, the probability that the error vector will have weight a (i.e., a entries 1 and 7-a entries 0) is binomial:

$$(3.12) \qquad P(a) = {}_7C_a p^a (1-p)^{7-a}, \qquad a = 0,1,2,\ldots,7.$$

Thus, the probability of any error vector depends on its weight. When $r \in C$, we seek the vector in C which maximizes (3.12). We note that the zero vector is in C, and that no vector of weight 1 is in C. But C cannot have a vector of weight 2 either; for example, if $1100000 \in C$, then we would have

$$0 = H\begin{bmatrix}1\\1\\0\\0\\0\\0\\0\end{bmatrix} = H\begin{bmatrix}1\\0\\0\\0\\0\\0\\0\end{bmatrix} + H\begin{bmatrix}0\\1\\0\\0\\0\\0\\0\end{bmatrix} = H\begin{bmatrix}1\\0\\0\\0\\0\\0\\0\end{bmatrix} - H\begin{bmatrix}0\\1\\0\\0\\0\\0\\0\end{bmatrix},$$

which would imply that the words 1000000 and 0100000 are in the same coset. Since $1111111 \in C$, we see that no vector of weight 5 or 6 can be a codeword, either. (These facts are also known from the results of Exercises 3 and 6.) Thus,

to maximize the function P(a) in (3.12), we need consider only a = 0,3,4,7. But since $0 < p < 1/2$, we need consider only 0 and 3. From (3.12) we have

$$P(0) = (1-p)^7, \qquad P(3) = 35(1-p)^4 p^3.$$

Therefore,

$$\frac{P(0)}{P(3)} = \frac{(1-p)^3}{35p^3},$$

and $(1-p)^3/35p^3 > 1$ when $0 < p < 1/(1+\sqrt[3]{35})$, or $0 < p < 0.234$ (approximately). Thus, when the received word r is a codeword, i.e., when $r \in C$, the choice of e = 0000000 for the error (with the concomitant choice of r for c) is the choice that maximizes the probability that the chosen vector is indeed the error, provided only that $0 < p < 0.234$. In practice, real-world digital communication channels satisfy this modest requirement, with room to spare.

When the received word r is not a codeword, we would like to know the condition(s) under which the choice of the vector of weight 1 from C + r maximizes (3.12). Since C + r has vectors of weights 1,2,3,4,5 and 6 only, and since $0 < p < 1/2$, we must maximize P(a) in (3.12) for a = 1,2,3, only. From (3.12) we have

$$P(1) = 7(1-p)^6 p; \quad P(2) = 21(1-p)^5 p^2; \quad P(3) = 35(1-p)^4 p^3.$$

It follows that

P(1) > P(2) whenever $0 < p < 1/4$, and

P(1) > P(3) whenever $0 < p < 0.309$ (approximately).

The verification is similar to the one above for the case $r \in C$. Therefore, in all cases the algorithm in Example 3 calls for the choice of e that will maximize P(a) in (3.12) whenever $0 < p < 0.234$ (approximately); hence the name, "maximum likelihood decoder."

We note that if the channel noise causes 1 error in a 7-bit message, i.e., inverts 1 bit, then the decoding algorithm will correct the error. Thus, in Examples 2 and 3 we have a 1-error correcting binary code of block length 7, with 16 possible codewords.

We close this section with an exercise that represents an invitation to share the experience of working through a Hamming code model from the beginning. This model corresponds to the value 4 of the parameter m (we considered the case m = 3 above). No doubt you will find that the calculations rapidly become tedious as m increases. Larger models are best handled by computers.

Exercises

10. For the (15,11) Hamming code with columns of the parity check matrix in natural order, and with maximum likelihood decoder:
    a. find the parity check matrix H;
    b. find the information rate;
    c. decode the received word 111100101100010;
    d. find conditions corresponding to (3.7) that characterize codewords in $GF(2)^{15}$;
    e. carry out an analysis to compare the gain in the probability of conveying information correctly under this coding scheme with the corresponding probability of no error under no coding scheme, which is $(1-p)^{11}$.

4. MODEL EXAMINATION

1. For a 3-error correcting binary repetition code of block length 7, with codewords 1111111 and 0000000,
    a. find the information rate;
    b. decode the received word 0011000;
    c. find the probability of an error P(E), using the BSC model.

2. Using our (7,4) Hamming code with maximum likelihood decoder, decode the words:
    a. r = 0110011;
    b. r = 1101101.

In problems 3 and 4 you will need results from Exercise 10.

3. Using the (15,11) Hamming code of Exercise 10, with maximum likelihood decoder, decode the words:
    a. r = 101010101011111;
    b. r = 110111000101101.

4. In our (7,4) Hamming code we found the following distribution in C: 1 word of weight 0, 7 of weight 3, 7 of weight 4, 1 of weight 7, and 0 of weights 1,2,5,6. In the (15,11) Hamming code, find:
    a. the number of words of weight 0;
    b. the number of words of weight 1;
    c. the number of words of weight 2;
    d. at least one word of weight 3 (there are 24 of them);
    e. the total number of codewords.

5. SOLUTIONS TO EXERCISES

1. a. P(E) = probability of 1 bit correct and 2 incorrect + probability of 0 bits correct and 3 incorrect
$= {}_3C_1(1-p)p^2 + p^3 = p^2(3-2p)$.

b.

p	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45
P(E)	0.007	0.028	0.061	0.104	0.156	0.216	0.282	0.352	0.425

c. 1/3.

2. a. The verification can be accomplished by carrying out the matrix multiplication to show that $Hx_i = 0$ for $i = 1,2,3,4$. For example,

$$Hx_1 = \begin{bmatrix} 0&0&0&1&1&1&1 \\ 0&1&1&0&0&1&1 \\ 1&0&1&0&1&0&1 \end{bmatrix} \begin{bmatrix} 1\\1\\1\\0\\0\\0\\0 \end{bmatrix} = \begin{bmatrix} 0+0+0+0+0+0+0 \\ 0+1+1+0+0+0+0 \\ 1+0+1+0+0+0+0 \end{bmatrix} = \begin{bmatrix} 0\\0\\0 \end{bmatrix}.$$

b. Suppose that a linear combination of the x_i's vanishes, i.e., that $\alpha_1x_1 + \alpha_2x_2 + \alpha_3x_3 + \alpha_4x_4 = 0$. (Note that the α's are all 0 or 1 and that all arithmetic is carried out in GF(2).) Then from the first and second rows we have

$$\alpha_1 + \alpha_2 + \alpha_4 = 0, \quad \text{and}$$

$$\alpha_1 + \alpha_3 + \alpha_4 = 0.$$

Upon adding we obtain $\alpha_2 + \alpha_3 = 0$, so $\alpha_2 = \alpha_3$. The third row shows that $\alpha_1 = \alpha_4$, the fifth that $\alpha_2 = \alpha_4$, and the seventh that $\alpha_4 = 0$. Thus, $\alpha_i = 0$, $i = 1,2,3,4$, so x_1,x_2,x_3,x_4 are linearly independent.

3. We may find all elements in C by forming all possible linear combinations $\alpha_1x_1 + \alpha_2x_2 + \alpha_3x_3 + \alpha_4x_4$, where the α_i's may assume the values 0,1. We list the 16 elements in C as words:

0000000, 1110000, 1001100, 0101010, 0010110, 0100101, 1000011, 1011001, 0001111, 0110011, 1010101, 1101001, 1011010, 0111100, 1100110, 1111111.

4. Verification can be accomplished by carrying out the indicated matrix multiplication to obtain the identity matrix.

5. We perform the suggested matrix multiplication:

$$\begin{bmatrix} 0&0&0&1&1&1&1 \\ 0&1&1&0&0&1&1 \\ 1&0&1&0&1&0&1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_1+x_2+x_3 \\ x_0+x_2+x_3 \\ x_0+x_1+x_3 \end{bmatrix} = \begin{bmatrix} x_3+x_1+x_2+x_3+x_0+x_2+x_3+x_0+x_1+x_3 \\ x_1+x_2+x_0+x_2+x_3+x_0+x_1+x_3 \\ x_0+x_2+x_1+x_2+x_3+x_0+x_1+x_3 \end{bmatrix}$$

$$= \begin{bmatrix} 0\\0\\0 \end{bmatrix}.$$

6. The answers, which are given in the solution for Exercise 3, can be obtained by making all possible assignments for x_0,x_1,x_2,x_3, then applying (3.7). For example, if we take $x_0 = 1$, $x_1 = 1$, $x_2 = 1$, $x_3 = 0$, then $x_4 = 1 + 1 + 0 = 0$, $x_5 = 1 + 1 + 0 = 0$, $x_6 = 1 + 1 + 0 = 0$, and we obtain the codeword 1110000.

7. a. $Hx = \begin{bmatrix} 0\\0\\1 \end{bmatrix}$, which corresponds to the first column of H.

b. The words in C + x can be obtained from the codewords by inverting the first bit in each word. Referring to the solution for Exercise 3 we obtain the 16 words in C + x:

1000000, 0110000, 0001100, 1101010, 1010110, 1100101, 0000011, 1011001, 1001111, 1110011, 0010101, 0101001, 0011010, 1111100, 0100110, 0111111.

8. a. $Hy = \begin{bmatrix} 0\\1\\0 \end{bmatrix}$, which corresponds to the second column of H.

b. The words in C + y can be obtained from the codewords by inverting the second bit in each word:

0100000, 1010000, 1101100, 0001010, 0110110, 0000101, 1100011, 0111001, 0101111, 0010011, 1110101, 1001001, 1111010, 0011100, 1000110, 1011111.

9. a. 1101001

b. 0010110

c. 1010101

d. 0000000

10. a.
$$\begin{bmatrix} 0&0&0&0&0&0&0&1&1&1&1&1&1&1&1 \\ 0&0&0&1&1&1&1&0&0&0&0&1&1&1&1 \\ 0&1&1&0&0&1&1&0&0&1&1&0&0&1&1 \\ 1&0&1&0&1&0&1&0&1&0&1&0&1&0&1 \end{bmatrix}.$$

b. 11/15.

c. 111100101100000.

d. Some care must be taken in using the technique that resulted in Condition (3.7). For example, if we try to echo the method to find the last 4 variables in terms of the first 11, we obtain a singular matrix. We must select a set of unknowns with a matrix of rank 4. There are many choices, of which one is the following:

$$H_1 = \left[\begin{array}{ccccccccc|cccc|ccc} 0&0&0&0&0&0&0&0&1&0&0&0&0&1&1&1 \\ 0&0&0&1&1&1&1&1&0&0&0&0&0&1&1&1 \\ 0&1&1&0&0&1&1&0&0&0&0&0&0&0&1&1 \\ 1&0&1&0&1&0&1&0&0&0&0&0&0&1&0&1 \end{array}\right],$$

$$H_2 = \left[\begin{array}{ccccccccc|cccc|ccc} 0&0&0&0&0&0&0&0&0&1&1&1&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&0 \end{array}\right].$$

If x is a codeword then, as in Equation (3.2), when we represent x as a column vector we find $H_1x = H_2x$. If we denote the bits (or the coordinates) of x by $x_0, x_1, \ldots, x_{14}$, we obtain a result similar to Equation (3.4):

$$\begin{bmatrix} 1&1&1&1 \\ 0&0&0&1 \\ 0&1&1&0 \\ 1&0&1&0 \end{bmatrix}\begin{bmatrix} x_8 \\ x_9 \\ x_{10} \\ x_{11} \end{bmatrix} = \begin{bmatrix} 0&0&0&0&0&0&0&1&1&1&1 \\ 0&0&0&1&1&1&1&0&1&1&1 \\ 0&1&1&0&0&1&1&0&0&1&1 \\ 1&0&1&0&1&0&1&0&1&0&1 \end{bmatrix}\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_{12} \\ x_{13} \\ x_{14} \end{bmatrix}.$$

Since

$$\begin{bmatrix} 1&1&1&1 \\ 0&0&0&1 \\ 0&1&1&0 \\ 1&0&1&0 \end{bmatrix}^{-1} = \begin{bmatrix} 1&1&1&0 \\ 1&1&0&1 \\ 1&1&1&1 \\ 0&1&0&0 \end{bmatrix},$$

we obtain

$$\begin{bmatrix} x_8 \\ x_9 \\ x_{10} \\ x_{11} \end{bmatrix} = \begin{bmatrix} 0&1&1&1&1&0&0&1&0&1&1 \\ 1&0&1&1&0&1&0&1&1&0&1 \\ 1&1&0&1&0&0&1&1&1&1&0 \\ 0&0&0&1&1&1&1&0&1&1&1 \end{bmatrix}\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_{12} \\ x_{13} \\ x_{14} \end{bmatrix}.$$

Therefore, x is a codeword if and only if

$$x_8 = x_1 + x_2 + x_3 + x_4 + x_7 + x_{13} + x_{14}$$
$$x_9 = x_0 + x_2 + x_3 + x_5 + x_7 + x_{12} + x_{14}$$
$$x_{10} = x_0 + x_1 + x_3 + x_6 + x_7 + x_{12} + x_{13}$$
$$x_{11} = x_3 + x_4 + x_5 + x_6 + x_{12} + x_{13} + x_{14}$$

e. Using the code, we have for the probability of no error:

P(N) = probability of 0 bits incorrect and 15 correct
+ probability of 1 bit incorrect and 14 correct

$= (1-p)^{15} + 15(1-p)^{14}p$

$= (1-p)^{11}(1 + 11p - 39p^2 + 41p^3 - 14p^4)$.

The following table shows the gain for a few values of p:

p	0.001	0.005	0.010	0.05
$(1-p)^{11}$	0.989	0.946	0.895	0.569
P(N)	0.99990	0.9975	0.990	0.829

6. ANSWERS TO MODEL EXAMINATION

1. a. 1/7;
   b. 0000000; $p^4(35 - 84p + 70p^2 - 20p^3)$.
2. a. 0110011;
   b. 1101001.
3. a. 1110101010111111;
   b. 110111000101101.
4. a. 1;
   b. 0;
   c. 0;
   d. Use the result of Exercise 10d: for example, if we assign $x_0 = 1$, and $x_k = 0$ for k = 1,2,3,4,5,6,7,12,13,14; then $x_8 = 0$, $x_9 = 1$, $x_{10} = 1$, $x_{11} = 0$, and we obtain the codeword 100000000110000;
   e. 2^{11}, or 2,048.

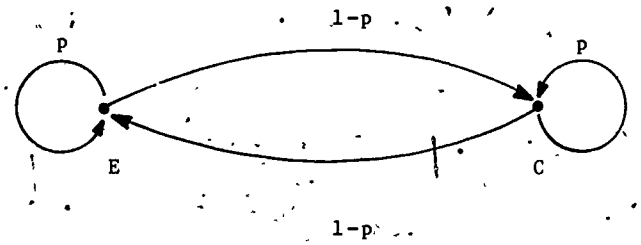
APPENDIX 1

AN ALTERNATE TO THE BSC

We describe briefly a model that is applicable in some situations where the BSC model fails. Note first that the BSC model may be represented as a finite Markov chain with two states, an "error state" E and a "correct state" C, with matrix of transition probabilities given by

$$\begin{array}{c} \\ C \\ E \end{array}\begin{array}{c} \begin{array}{cc} C & E \end{array} \\ \begin{bmatrix} 1-p & p \\ p & 1-p \end{bmatrix} \end{array}$$

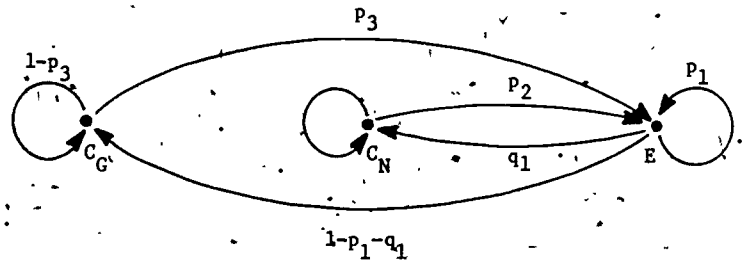
A graph-theoretic model is also useful in visualizing this chain. The states may be represented as vertices, and the transition probabilities are shown on the edges.



The Markov chain approach can be used to obtain a model which represents reasonably well some channels in which errors occur in bursts, for example, certain HF channels. In this model there are three states: C_G is a "guard state," and the transmission is error free while the system is in this state; the remaining states C_N and E are "bursty" (noisy) states. The matrix of transition probabilities is given by

$$\begin{array}{c} \\ C_G \\ C_N \\ E \end{array} \begin{array}{c} \begin{array}{ccc} C_G & C_N & E \end{array} \\ \begin{bmatrix} 1-p_3 & 0 & p_3 \\ 0 & 1-p_2 & p_2 \\ 1-p_1-q_1 & q_1 & p_1 \end{bmatrix} \end{array}$$

This chain may also be represented as a graph.



This model can be interpreted in a straightforward way. For example, suppose that the system is in state C_G, so that error-free transmission is in progress. If p_3 is small, then there is a high probability of continuing in state C_G, but it is also possible to make an error. We move to the error state E with probability p_3, and from this state it is possible to continue in state E, return to state C_G, or enter the error-free state C_N. Analytically, the difference between states C_G and C_N is that p_3 is much smaller than p_2. Thus when the channel is in state C_G, it has a tendency to remain there. When the error burst is over, we move back to state C_G for another period of error-free transmission.

This 3-state Markov model can be generalized to a model with any number of states. Increasing the number of states to 5, say, with three error-free states and two error states may enable us to model a wider class of channels accurately. However, there are two major difficulties. First, as the number of states increases, the physical significance of these states may be difficult to ascertain; and, second, even in the case of the 3-state model, computation of P(N) in Table 1 in terms of p_1, p_2, p_3, q_1 is quite complicated.

The customary method for constructing an analog to Table 1 for a multi-state Markov model is to use a Monte Carlo approach, that is, to generate an "error stream" of 0's and 1's according to the model. The error stream is then divided into n-bit blocks and decoded. If decoding produces any result other than 00...0, the all-0 block, then a decoding error has occurred.

It is worthwhile to note that if errors tend to occur in bursts, then a code, such as a Hamming code, which is designed to correct single, isolated errors may be worse than useless, for two reasons. First, most of the time, while the channel is in the guard state, no errors occur at all, so that all the code is doing in this state is adding unnecessary redundancy; and second, when errors do occur, the chances are that any block with at least one error will contain more than one, which results in a decoding error, thus producing even more errors. Therefore, channel considerations are very important in designing error correcting codes. Our next two modules on error-correcting codes will contain some examples of codes that are designed to correct burst errors.

APPENDIX 2

A NOTE ON MULTIPLEXING, OR "INTERLEAVING" OF CODES

Suppose m binary k-tuples, representing (perhaps) measurements from m different instruments, are encoded using an (n,K) code into m codewords $C_i = (C_{i1}, C_{i2}, \ldots, C_{in})$, $1 \leq i \leq m$. We could represent these codewords in an m×n array

$$C = \begin{bmatrix} C_1 \\ \vdots \\ C_m \end{bmatrix} = (C_{ij}), \quad 1 \leq i \leq m, \quad 1 \leq j \leq n.$$

Now, rather than transmit C_1, then C_2, then C_3, etc., we transmit the *columns* of C: first $(C_{11}, C_{21}, \ldots, C_{m1})$, then $(C_{12}, C_{22}, \ldots, C_{m2})$, then $(C_{13}, C_{23}, \ldots, C_{m3})$, etc. The recipient of the transmission receives $(r_{11}, r_{21}, r_{31}, \ldots, r_{m1}, r_{12}, r_{22}, \ldots, r_{mn})$, which are written by *column* into an m×n array $R = (r_{ij})$. The rows of R are (possibly) corrupted codewords. If a "burst" of errors of length $b < m$ has occurred in transmission over the channel, the errors are spread through successive rows of R. Therefore, in a given row, whatever errors occur seem to be isolated, independent errors. In such a situation, the binary symmetric channel may give a reasonable model for the errors appearing in the received words to be decoded.

STUDENT FORM 1

Request for Help

Return to:
EDC/UMAP
55 Chapel St.
Newton, MA 02160

Student: If you have trouble with a specific part of this unit, please fill out this form and take it to your instructor for assistance. The information you give will help the author to revise the unit.

Your Name _____ Unit No. _____

Page _____
- ○ Upper
- ○ Middle
- ○ Lower

OR

Section _____
Paragraph _____

OR

Model Exam
Problem No. _____
Text
Problem No. _____

Description of Difficulty: (Please be specific)

Instructor: Please indicate your resolution of the difficulty in this box.

○ Corrected errors in materials. List corrections here:

○ Gave student better explanation, example, or procedure than in unit. Give brief outline of your addition here:

○ Assisted student in acquiring general learning and problem-solving skills (not using examples from this unit.)

Instructor's Signature _____

Please use reverse if necessary.

STUDENT FORM 2

Unit Questionnaire

Return to:
EDC/UMAP
55 Chapel St.
Newton, MA 02160

Name _____ Unit No. _____ Date _____

Institution _____ Course No. _____

Check the choice for each question that comes closest to your personal opinion.

1. How useful was the amount of detail in the unit?

___Not enough detail to understand the unit
___Unit would have been clearer with more detail
___Appropriate amount of detail
___Unit was occasionally too detailed, but this was not distracting
___Too much detail; I was often distracted

2. How helpful were the problem answers?

___Sample solutions were too brief; I could not do the intermediate steps
___Sufficient information was given to solve the problems
___Sample solutions were too detailed; I didn't need them

3. Except for fulfilling the prerequisites, how much did you use other sources (for example, instructor, friends, or other books) in order to understand the unit?

___A Lot ___Somewhat ___A Little ___Not at all

4. How long was this unit in comparison to the amount of time you generally spend on a lesson (lecture and homework assignment) in a typical math or science course?

Much Longer	Somewhat Longer	About the Same	Somewhat Shorter	Much Shorter
___	___	___	___	___

5. Were any of the following parts of the unit confusing or distracting? (Check as many as apply.)

___Prerequisites
___Statement of skills and concepts (objectives)
___Paragraph headings
___Examples
___Special Assistance Supplement (if present)
___Other, please explain _____

6. Were any of the following parts of the unit particularly helpful? (Check as many as apply.)

___Prerequisites
___Statement of skills and concepts (objectives)
___Examples
___Problems
___Paragraph headings
___Table of Contents
___Special Assistance Supplement (if present)
___Other, please explain _____

Please describe anything in the unit that you did not particularly like.

Please describe anything that you found particularly helpful. (Please use the back of this sheet if you need more space.)