Non-Binary Error Correction Codes*

By WERNER ULRICH

(Manuscript received April 19, 1957)

If a noisy channel is used to transmit more than two distinct signals, information may have to be specially coded to permit occasional errors to be corrected. If pulse amplitude modulation is used, the most probable error is a small one, e.g., 6 is changed to 7 or 5. Codes for correcting single small errors, and for correcting single small errors and detecting double small errors, in a message of arbitrary length, for an arbitrary number of different signals in the channel, are derived in this paper.

For more specialized situations, the error is not necessarily restricted to a small value. Codes are derived for correcting any single unrestricted error in a message of arbitrary length for an arbitrary number of different signals.

Finally, a set of codes based partially upon the Reed-Muller codes is described for correcting a number of errors in a more restricted class of message lengths for an arbitrary number of different signals.

The described codes are readily implemented. Many techniques are used which have an analog in a binary system. Other techniques are broadly analogous to binary coding techniques or are special adaptations of a binary code.

I. INTRODUCTION

1.1 *Use of Error Correction Codes*

One function of an error correction code is to aid in the correct transmission of digital information over a noisy channel. This process is illustrated in Fig. 1. An information source gives information to an encoder; the encoder converts the information into a message containing sufficient redundancy to permit the message to be slightly mutilated by the noisy channel and still be correctly interpreted at the destination. The message is then sent via the noisy channel to a decoder which will

* This paper was submitted to Columbia University in partial fulfillment of the requirements for the degree of Doctor of Engineering Science in the Faculty of Engineering.

reconstruct the original information if the mutilation has not been excessive. Finally, the information is sent to an information receptor.

One scheme for correcting errors in a binary system is to send each binary digit of information three times and to accept at the receiver that value which is represented by two or three of the received digits. Then, the encoder is simply an instrument for causing each digit to be sent three times, and the decoder consists of a majority organ. However, many methods are available which are considerably more elegant, and which will permit more information to be passed through a noisy channel in a given unit of time. This paper will deal with such methods for channels capable of sending b different symbols instead of the usual 1 and 0 of a binary channel.

The most convenient explanation of an error correction code has been made with respect to the transmission of correct digital information over a noisy channel. This does not imply the restriction of such codes

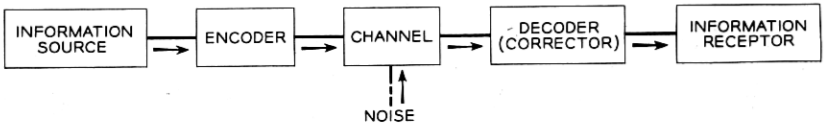

Fig. 1 — Transmission over a noisy channel.

to the noisy channel problem exclusively. Actually, the first application considered for such a code was with respect to computers.[1] Many large high speed computers stop whenever an error is detected in some calculation and must be restarted; with the use of an error correction code this could be avoided by permitting the computer to correct its own random errors directly. To the best knowledge of the author, error correction codes have not yet been used in any major computer. But the storage system of a computer may, in the future, lend itself to the use of error correction codes.

Frequently, very elaborate precautions must be taken in present storage systems to insure that they are free from errors. Magnetic tapes must be specially made and handled to guarantee the absence of defects, magnetic cores must be carefully tested to make sure that no defective cores get into an array, cathode ray tubes used in Williams Tube or Barrier Grid Tube storage systems must be perfect. Probably, there are other storage methods whose development is hampered because of a common requirement for error-free performance in all storage locations. With the use of error correction codes, such storage systems could be used, if they are sufficiently close to perfection, even though not perfect.

It is not unlikely that the near future will see the development of storage systems which will be able to store more than two states at every basic storage location.[2] If such systems are developed, it seems likely that they will be more erratic or noisy than binary storage systems, since each location must store one of b signals instead of one of two. If a cathode ray tube storage system were used, for example, different quantities of charge would have to be distinguished; in a binary storage system, only the presence or absence of charge must be detected. This suggests that error correction codes may become essential with certain types of non-binary storage systems. One object of this paper is to develop codes for this purpose and to discover which number systems are most easily correctable.

Some investigations have been made on the use of computer systems using multi-state elements.[3] A switching algebra has been developed similar to Boolean algebra for handling switching problems in terms of multi-state elements. Single device ring counters (the cold cathode gas stepping tube for example) already exist and might be useful in such systems. But currently, only limited steps in this direction have been made. Another object of this paper is to show the advantages and problems of error correction codes in multi-state systems; it is not unreasonable to predict that error correction codes may be more necessary in multi-state systems than in binary systems.

1.2 *Geometric Concept of Error Correction Codes*

A geometric model of a code was suggested by R. W. Hamming[1] which can be altered slightly to fit the non-binary case. For an n digit message, a particular message is a point in n dimensional space. A single error, however defined, will change the message, and will correspond to another point in n dimensional space. The distance between the original point and the new point is considered to be unity. Thus, the distance d between the points corresponding to any two messages is defined as the minimum number of errors which can convert the first message into the second.

With an error detection and/or correction code, the set of transmitted messages is limited so that those which are correctly received are recognizable; those messages which are received with fewer than a given number of errors are either corrected or the fact that they are wrong is recognized and some other appropriate action (such as stopping a computer) is taken.

In the case of binary codes, an error changes a 1 to 0 or a 0 to 1. In the non-binary case, two definitions of an error are possible and will be

used in this paper. A *small* error changes a digit to an adjacent value. In a decimal system, a change from 1 to 2 or 1 to 0 is a small error. An *unrestricted* error changes a digit to any other value. In a decimal system, a change from 1 to 5 is an unrestricted error.

1.3 *Material To Be Presented*

The various types of codes described in this paper and the sections in which they are to be found are summarized in Table I. The techniques which are described are summarized below.

The geometric model suggests the simplest approach to error correction codes. A transmitter has a "codebook" containing all members of the set of transmitted messages. If the message source gives to the encoder the signal that the information to be sent is k (that is to say, the kth

TABLE I — TYPES OF CODES

Type of Code	Distance	Type of Error	Described in Section
Single Error Detection	2	Small and Unrestricted	II
Single Error Correction	3	Small	III and 6.1
Single Error Correction	3		
Prime Number Base		Unrestricted	4.1
Composite Number Base		Unrestricted	4.2
Single Error Correction and Double Error Detection	4	Small	V and 6.1
Multiple Error Correction	—	Small	6.2

output of all the outputs associated with the message source), the encoder chooses the kth member of the set. The decoder will then look up the message it receives in its own codebook which contains all possible received messages, and corresponding to the entry of the received message will find the symbols corresponding to k. Or the receiver may compare the received message with every member of the set of transmitted messages, calculate the distance between the two, and correct the received message to whichever of the transmitted messages is separated from the received message by the smallest distance. (It has been shown by Slepian[4] that this is the message most likely to be correct in a symmetrical binary channel having the property that changes from 1 to 0 and from 0 to 1 as a result of noise in the channel are equally likely.)

The practical difficulty with such a code is the large size of the required codebooks. Most coding schemes try to eliminate such codebooks and substitute a set of rules for encoding, decoding and correcting messages.

One approach toward creating a simple association between the information and the message is to use some of the digits of the message for conveying information directly. The Hamming Code[1] uses this technique.

An *information digit* is a digit of a message that is produced directly by the information source; in a base b code, an information digit may have b different values, the choice between these values representing the information that is to be sent.

A *check digit* is a digit of a message that is calculated as a function of the information digits by the encoder. It is sometimes convenient to represent or calculate a check digit in terms of a recursive formula using previously calculated check digits as well as information digits. In a base b code, a check digit may have b *check states*. When more than one check digit is used, each different combination of check digits corresponds to a different check state for the message; a message with m check digits will have b^m *message check states*.

A *systematic*[5] *code* encoder generates messages containing only information digits and check digits. The information source generates only base b information digits. The Hamming Code is a systematic code.

Section II offers a general method for obtaining single error detection codes for both small and unrestricted errors. The idea of mixed digits (digits which are, in a sense, neither information nor check digits, but a combination of both) is introduced, and it is shown how mixed digits may lead to more efficient coding systems. This idea is believed to be novel. Code systems which use mixed digits are called semi-systematic codes. Semi-systematic codes are used extensively throughout this paper.

Section III offers a general method for obtaining single small error correction codes, including both systematic and semi-systematic codes.

Section IV offers a general method for obtaining the more complicated single unrestricted error correction codes. The problem is divided into two parts. Section 4.1 describes codes for correcting single unrestricted errors in case b, the base of the channel, is a prime number.* Section 4.2 describes a special technique for obtaining the more complex codes for correcting single unrestricted errors in the event b is a composite number.

Section V offers a general method for obtaining semi-systematic codes for correcting single small errors and detecting double small errors. No general solution has been found for obtaining single error correction or double error detection codes for the case of unrestricted errors. No gen-

* This class of codes was previously described in a brief summary by Golay.[6]

eral solution has been found for multiple error correction codes for the unrestricted error case.

In Section VI, a number of techniques are presented for using binary error correction coding schemes for non-binary error correction codes. Section 6.1 shows how such techniques may be used to obtain non-binary single error correction codes, and single error correction double error detection codes, for the small error case. Section 6.2 presents a special technique, involving the use of an adaptation of the Reed-Muller binary code, to obtain a class of non-binary multiple error correction codes, for the small error case.

Section VII shows that an iterative technique of binary coding can be directly applied to non-binary codes. It also shows how an adapted Reed-Muller code can be profitably used in such a system.

Section VIII summarizes the results obtained in Sections II–VII and shows the advantages and shortcomings of many of these codes.

Section IX presents general conclusions which may be drawn from this paper.

II. SINGLE ERROR DETECTION CODES

Single error detection codes require message points separated in n dimensional space by a distance of two.

For the binary case, the only two possible types of errors are the change from a 1 to a 0 and from a 0 to a 1.

A simple technique that is used frequently for binary error detection codes is to encode all messages in such a manner that every message contains an even number of 1's. This is accomplished by adding a *parity check digit* to the information digits of a message; this digit is a 1 if an odd number of 1's exist in the information digits of a message and is a 0 if an even number of 1's exist in the information digits. At least two errors must occur before a message containing an even number of 1's can be converted into another message containing an even number of 1's, since the first error will always cause an odd number of 1's to appear. A message with an odd number of 1's is known to be incorrect.*

An analogous technique may be used for the unrestricted error case in non-binary codes. We can obtain a satisfactory code by adding a complementing digit to a series of information digits to form a message.

A *complementing* digit, base b, is defined as a digit which when added to some other digit will yield a multiple of b.

* Parity check digits may be selected to make the number of 1's in a message always odd, but the principle is the same; in this case, an error is recognized if a received message contains an even number of 1's.

For a single unrestricted error detection code, the complementing digit complements the sum of the information digits. A complementing digit is a check digit. In the binary case, it is a parity check digit.

As an example, consider a decimal code of this type. A message 823 would require a complementing digit 7, making the total message 8237 ($8 + 2 + 3 + 7 = 20$, a multiple of 10). An error in any one digit will mean that the sum of the message digits will not be a multiple of 10.

For the small error case, it is sufficient to make certain that the sum of all digits is even since any error of ± 1 would destroy this property. For the binary case, all errors are small since the only possible error on any digit is a change by ± 1; a simple parity check is adequate. For a non-binary code, it would be wasteful to add a digit just to make sure that the sum of all digits is even. In a decimal code for example, if the sum of the message digits is even, the values 0, 2, 4, 6, 8 for the check digit will satisfy a check, or if the sum of the message digits is odd, the values 1, 3, 5, 7, 9 will satisfy the check. More information could be sent if a choice among these values could be associated with information generated by the information source.

This introduces the concept of a mixed digit; i.e., a digit which conveys both check information and message information.

A *mixed digit* is defined as follows: a mixed digit x, base b, is composed of two components (y, z) where y represents an information component and z represents a check component. The number of information states of a mixed digit is β, with y taking the values $0, 1, \cdots, \beta - 1$; the number of check states of a mixed digit is α, the number base of z. In a message containing m check digits and h mixed digits, the number of check states for the *message* is $b^m \cdot \alpha_1 \cdot \alpha_2 \cdot \ldots \cdot \alpha_h$, where α_i is the number of check states of the i'th mixed digit.

If mixed digits are used as part of a code, information must be available in at least two number bases; b, the number base of the channel, and β, the number base of the mixed digit. A situation where this arises naturally is in the case of the algebraic sign of a number; this is a digit of information, base 2, which may be associated with other digits of any base. Similarly, any identification which must be associated with numerical information can be conveniently coded in a number base different from the number base of the numerical information. Thus, a mixed digit can sometimes be used conveniently in an information transmission system without complicating the information source and receptor.

An error detection code for single small errors suggests the use of a mixed digit. In the decimal code for example, the quibinary[7] representa-

TABLE II — QUIBINARY CODE

Quinary Component	Binary Component	Decimal Digit
0	0	0
0	1	1
1	0	2
1	1	3
2	0	4
2	1	5
3	0	6
3	1	7
4	0	8
4	1	9

tion of the mixed digit might be used, letting the quinary component of the mixed digit convey information and the binary component a check. (Table II.)

The information source generates blocks of decimal digits followed by one quinary digit. The messages are then generated in the following way: record all decimal information digits as information message digits and take their sum; if the sum is even, the binary component, z, of the mixed digit is 0, otherwise it is 1. The quinary component, y, of the mixed digit is taken directly from the information source and combined with the calculated binary part by the rules of the quibinary code to form the mixed decimal digit. Thus, x, the value of the mixed digit, is given by the formula:

$$x = 2y + z. \tag{1}$$

For example, if the decimal digits of a message are 289 and the quinary digit of the message is 3, the mixed digit is 7, and the message is 2897. The sum of the decimal information digits is 19, which is odd, so that the binary component of the mixed digit is 1; this is combined with the quinary component, 3, by the rules of the quibinary code table, to form decimal digit 7. The requirement that the sum of all digits be even is satisfied by the binary component of the mixed digit, and the information associated with the mixed digit is contained in the quinary component.

This method is easily extensible to any other number base and is also extensible to the case of slightly larger but still restricted errors (such as ± 1 or ± 2), provided that the maximum single error is less than $(b - 1)/2$.

From the preceding example, it is apparent that mixed digits can be usefully employed in error detection codes. The use of mixed, check and

information digits simplified the encoder and decoder. To differentiate among the classes of codes which will be described in this paper, the following terms will be used, in addition to those previously defined.

A *semi-systematic code* encoder produces messages containing only information, mixed and check digits. The information source generates information digits in base b for information digits, and in base β for mixed digits. (The example given above is a semi-systematic code.)

Of two coding schemes in the same channel base b, each working with messages of the same length, and each satisfying a given error detection or correction criterion, the more *efficient* scheme is defined as the one which produces the larger number of different possible messages.

III. SINGLE ERROR CORRECTION CODES, SMALL ERRORS (± 1)

The problems of error correction codes in nonbinary systems are extensive and must be treated in several distinct sections. The basic difference between the error correction problem in binary and non-binary codes is the fact that the sign of the error is important. In a binary code, if the message 11 is received and it is known that the second digit is incorrect, only one correction can be made, to 10. But in a decimal code with errors limited to ± 1, if the message 12 is received and it is known that the second digit is wrong, it can be changed to either 11 or 13.

Consider the following simple code for correcting single small errors. A decimal channel is used, and a message is composed of three information digits and one check digit. Let x_1 represent the check digit and x_2, x_3, x_4 the information digits. Here, x_1 is chosen to satisfy*

$$x_1 + 2x_2 + 3x_3 + 4x_4 = 0 \bmod 10. \tag{2}$$

The encoder calculates x_1, and transmits the message $x_1x_2x_3x_4$. This is received as $x_1'x_2'x_3'x_4'$. The decoder then calculates c given by

$$c = (x_1' + 2x_2' + 3x_3' + 4x_4') \bmod 10. \tag{3}$$

If the assumption is made that at most a single small error exists, then this error can be corrected by using the following rules, which may be verified by inspection.

If $c = 0$, no correction is necessary;

$5 > c > 0$, decrease the cth digit by one;

* By definition $a = c \ mod \ b$ is equivalent to $a = c + nb$, where a, b, c and n are integers. The equality notation is used in preference to the congruence notation throughout this paper, since an addition performed without carry occurs naturally in many circuits; in terms of such a circuit, the mod b signifies only the base of the addition, and a true equality exists between the state of two circuits, with the same output even though one has been cycled more often.

$5 < c$, increase the $(10 - c)$th digit by one;

$c = 5$ implies a multiple error or a larger error.

Since the value of c is used for correcting a received message, it is called the corrector.* For the general case, a corrector is defined as follows.

In a message encoded to satisfy m separate checks, the result of calculating the checks for the received message at the decoder is an m digit word called the *corrector*. There are as many possible values of the corrector as there are check states of the message, although all of the values of the corrector need not correspond to a correctable error.

It is important that, for a given transmitted message, every different error will lead to a different value of the corrector; otherwise there will be no way of knowing which correction corresponds to a particular value of the corrector. The number of correctable errors may be far less than the number of possible values of the corrector, so that not all of these values may be useful for a code to correct a particular class of errors. However, the number of corrector states sets an upper limit to the number of possible corrections.

For many codes, it is convenient to associate a particular value of a corrector for the condition that a particular digit has been received too high by a single increment, for example, a 7 received as an 8.

The *characteristic* of a digit for a particular code is defined as the value of the corrector if that digit is incorrectly received, the error having increased the value of the digit by $+1$, and all other digits are correctly received. Obviously, this definition only applies to those codes having the property that the value of the corrector is independent of the value of the incorrect digit and of the other digits.

A *simple characteristic code* encoder produces messages in which each digit has a distinct characteristic as defined above.

The Hamming code is an example of a simple characteristic code as is the code previously described. In that example, the characteristic of x_i is i.

The advantage of a simple characteristic code for single small error correction is obvious: the association between the calculated checks and the correction to be performed is simple and does not depend on the values of the digits of the message.

The following example of a simple characteristic code will illustrate this principle more fully.

Consider a single small error correction code, working with a quinary

* The terms corrector and characteristic were first used in a more restricted sense in an article on binary coding by Golay.[8]

(base 5) channel. Each message will consist of ten information digits and two check digits.

Let x_1 and x_2 represent the check digits, and $x_3, x_4, \cdots, x_{12}$ represent the information digits.

The equations for calculating x_1 and x_2 are:

$$1x_1 + 0x_2 + 0x_3 + 1x_4 + 1x_5 + 1x_6 + 1x_7 + 2x_8 + 2x_9 + 2x_{10} + 2x_{11} + 2x_{12} = 0 \bmod 5, \tag{4}$$

$$0x_1 + 1x_2 + 2x_3 + 1x_4 + 2x_5 + 3x_6 + 4x_7 + 0x_8 + 1x_9 + 2x_{10} + 3x_{11} + 4x_{12} = 0 \bmod 5. \tag{5}$$

At the decoder, the corrector terms, c_1 and c_2, are calculated using x_i', the received value of x_i, in the following formulas:

$$1x_1' + 0x_2' + 0x_3' + 1x_4' + 1x_5' + 1x_6' + 1x_7' + 2x_8' + 2x_9' + 2x_{10}' + 2x_{11}' + 2x_{12}' = c_1 \bmod 5, \tag{6}$$

$$0x_1' + 1x_2' + 2x_3' + 1x_4' + 2x_5' + 3x_6' + 4x_7' + 0x_8' + 1x_9' + 2x_{10}' + 3x_{11}' + 4x_{12}' = c_2 \bmod 5. \tag{7}$$

The values of c_1c_2 corresponding to the condition that one and only one digit is too high by 1, $x_i' = x_i + 1$, can be read by reading the coefficients of the ith digit in the corrector formulas. This quantity is therefore the characteristic of the ith digit. If $x_i' = x_i - 1$, then the fives complements of these coefficients will be the value of the corrector. Table III lists the characteristics and characteristic complements associated with each digit.

TABLE III — CHARACTERISTICS AND CHARACTERISTIC COMPLEMENTS SYSTEMATIC QUINARY CODE

Digit	Characteristic	Complement of Characteristic
x_1	10	40
x_2	01	04
x_3	02	03
x_4	11	44
x_5	12	43
x_6	13	42
x_7	14	41
x_8	20	30
x_9	21	34
x_{10}	22	33
x_{11}	23	32
x_{12}	24	31

In this code all the possible values of c_1c_2 correspond to the characteristic of a digit or the complement of this characteristic, except 00 which corresponds to the correct message. (An inspection of equations (4) through (7) reveals that if $x_i' = x_i$ for all values of i, the values of c_1 and c_2 are 0). Thus, we can assign a unique correction to each value of c_1c_2 .

The above techniques are extensible to other number bases and different length words provided b, the number base of the channel, is greater than 2. (The equivalent binary channel problem has been treated by Hamming.[1]) The following set of rules and conventions may be used for deriving a satisfactory set of characteristics for a simple characteristic systematic code used to correct single small errors for any length message, and any base, $b \geqq 3$. The rules must be followed, and the conventions (which represent one pair of conventions out of the set of pairs of conventions, which together with Rules 1 and 2 can be used for deriving a code of this class) if followed, will lead to a reasonably simple method for encoding and decoding messages.* Since the rules, not the conventions, limit the efficiency of the code, no set of conventions can be found which will lead to a more efficient code of this class.

Rule 1. For an n digit message (including check digits), m check digits are required and m must satisfy the following inequalities:

$$\text{if } b \text{ is odd}, \qquad \frac{b^m - 1}{2} \geqq n, \tag{8a}$$

$$\text{if } b \text{ is even}, \qquad \frac{b^m - 2^m}{2} \geqq n. \tag{8b}$$

Rule 2. No characteristic may be repeated; i.e., each digit must have a characteristic different from that associated with any other digit.

Convention 1. The various digits of a characteristic are arranged in a set order; i.e., C_{1i} , C_{2i} , $\cdots$, C_{mi} . The first digit which is neither zero, nor (in case b is even) $b/2$, must be less than $b/2$. There must be at least one such digit.

Convention 2. The characteristic of the jth check digit has a 1 in the jth position and 0's elsewhere.

Rule 1 is required since, for a code of this type, we must be prepared to correct any digit in one of two ways (± 1). This implies a minimum of $2n + 1$ values of the corrector, one for each possible correction, and one for the case of no corrections. This means that b^m, the number of possible

* The above distinction between rules and conventions will be observed throughout this paper.

values of the corrector, must be at least $2n + 1$, equation (8a). For even bases, we must reject all values of the corrector containing only the digits 0 and $b/2$ for representing error conditions for the following reasons: a positive error leads to a corrector that is the characteristic of the incorrectly received digit, and a negative error leads to the b-complement of such a characteristic. In order to have unique error correction, we must be able to distinguish between these two conditions. If a characteristic were to contain only the digits 0 and $b/2$, it would be equal to its own b-complement; such combinations of digits are therefore not useable as characteristics or characteristic complements.

Rule 2 is required to permit a unique identification of an incorrect digit in case of a single error.

Convention 1 allows us to distinguish between positive and negative errors. By observing this convention, a characteristic (corresponding to a positive error) can be distinguished from its complement (corresponding to a negative error) by inspecting the first digit of a corrector which is neither 0 nor $b/2$. A characteristic will have this digit less than $b/2$, a characteristic complement will have this digit greater than b/2. If the corrector is a characteristic, the correction is minus one; if it is a characteristic complement, it is plus one.

Once the characteristics have been chosen, the corresponding encoding procedure may be performed in the following manner: Let a_{ij} represent the jth digit of the characteristic of information digit x_i. Let z_j represent the check digit which has a characteristic containing a 1 in the jth position. If convention 2 has been observed, (9) can be used to calculate z_j:

$$\sum_{i=1}^{n-m} a_{ij}x_i = -z_j \bmod b. \tag{9}$$

An encoder calculates each z_j and inserts it into the message in those digit positions which have the characteristic of the jth check digit assigned to them.

In more general terms, we use implicit relations that are equivalent to the explicit equations given by (9). Letting x_i represent an information or a check digit, and letting C_{ij} represent the jth digit of the characteristic of the ith information or check digit, these formulas may be rewritten as

$$\sum_{i=1}^{n} C_{ij}x_i = 0 \bmod b. \tag{10}$$

At the receiver, the decoder calculates m different check sums. Let c_j

represent the check sum corresponding to the jth corrector term, and x_i' represent the received value of x_i : Then,

$$\sum_{i=1}^{n} C_{ij}x_i' = c_j \bmod b. \tag{11}$$

The difference between equations (10) and (11) is the result of any mutilations caused by the channel. If no error has occurred, all the c_j's are 0; if an error of ± 1 has occurred, the m c_j's will form the characteristic or the characteristic complement, respectively, of the incorrectly received digit.

One disadvantage of a systematic code is the discontinuity in the number of check states as a function of m, the number of check digits. For example, in decimal code one check digit is required for a message of up to four digits, and two check digits for up to forty-eight digits. Obviously, for a message of intermediate length, for example, twelve digits, many of the corrector states cannot be used for single error correction since they will not correspond to any single error. A more efficient code would be obtained if the check states were limited to a smaller number.

One method of reducing the number of check states is to perform the check in a different modulus than the modulus of the channel. In the single error detection code using a mixed digit, binary check information and quinary message information was conveyed by this digit. This code was more efficient than a systematic code because each message contained the minimum number of check states which is 2.

If a mixed digit, x, is composed of the two components (y, z) where y is the information state of the digit and z the check state, it is convenient to combine these two components to form x by means of the formula

$$x = \alpha y + z. \tag{12}$$

We calculate z by using a linear congruence equation modulo α.

The use of this formula permits a decoder to act on x', the received value of x, directly, without first resolving x' into y' and z', because (12) insures that $x' = y' \bmod \alpha$. This permits x' to be corrected directly and then resolved into its components.

As an example, consider a semi-systematic code for correcting a single small error in a decimal system, using a twelve digit message; ten of the digits are information digits and two are mixed digits, each conveying binary message information and quinary check information. (One of these binary digits might represent the sign of the number.)

With two quinary checks, twenty-five different check states are possible; for correcting single small errors in a twelve digit message, twenty-

TABLE IV — CHARACTERISTICS AND CHARACTERISTIC COMPLEMENTS, SEMI-SYSTEMATIC DECIMAL CODE

Digit	Characteristics	Characteristic Complements
x_1 (mixed digit)	1 0	4 0
x_2 (mixed digit	0 1	0 4
x_3	0 2	0 3
x_4	1 1	4 4
x_5	1 2	4 3
x_6	1 3	4 2
x_7	1 4	4 1
x_8	2 0	3 0
x_9	2 1	3 4
x_{10}	2 2	3 3
x_{11}	2 3	3 2
x_{12}	2 4	3 1

five corrector states are required, one for each of the two possible corrections (± 1) for each digit, and one for the case of a correctly received message. Characteristics may be chosen for the various digits in accordance with the rules and conventions outlined above in this case, since the check modulus is the same for both check digits. Consequently, it is no accident that these characteristics, shown in Table IV, are the same as those shown in Table III.

Let C_{i1} and C_{i2} represent the characteristic of the ith digit, and let y_1 and y_2 represent the two binary information digits. Then:

$$\sum_{i=3}^{12} C_{i1}x_i = -z_1 \bmod 5, \qquad x_1 = z_1 + 5y, \tag{13}$$

$$\sum_{i=3}^{12} C_{i2}x_i = -z_2 \bmod 5, \qquad x_2 = z_2 + 5y_2\,. \tag{14}$$

Because $x_1 = z_1 \bmod 5$ and $x_2 = z_2 \bmod 5$, these relations can be rewritten implicity to resemble equation (10):

$$\sum_{i=1}^{12} C_{i1}x_i = 0 \bmod 5, \tag{15}$$

$$\sum_{i=1}^{12} C_{i2}x_i = 0 \bmod 5. \tag{16}$$

At the decoder, the corrector c_1c_2 is calculated by:

$$\sum_{i=1}^{12} C_{i1}x_i' = c_1 \bmod 5, \tag{17}$$

$$\sum_{i=1}^{12} C_{i2}x_i' = c_2 \bmod 5. \tag{18}$$

If the corrector is 00, the message has been correctly received; otherwise, the corrector is either the characteristic or characteristic complement of the incorrect digit, from which plus one or minus one respectively must be subtracted as a correction.

Consider the general case. Let $x_1, x_2, \cdots, x_k$ represent the k information digits; $y_1, y_2, \cdots, y_m$ represent the information state of the m mixed digits, and $z_1, z_2, \cdots, z_m$ represent the check state of the m mixed digits. In addition, let $\alpha_1, \alpha_2, \cdots, \alpha_m$ represent the number base of $z_1, z_2, \cdots, z_m$ respectively; $\beta_1, \beta_2, \cdots, \beta_m$ represent the number of possible states of $y_1, y_2, \cdots, y_m$ respectively, and $x_{k+1}, x_{k+2}, \cdots, x_{k+m}$ represent the values of the mixed digits after the message has been encoded. (Note that for simplicity, a check digit is considered as a special case of a mixed digit; its information state is permanently 0.) The following encoding procedure may be used in which $x_1, x_2, \cdots, x_k$ are used directly as part of the transmitted message. This is a semi-systematic code, which means that information digits are not changed in coding. To derive the mixed digits, the following formulas are used:

$$a_{11}x_1 + \cdots + a_{1k}x_k = -z_1 \bmod \alpha_1 \qquad (19\text{-}1)$$

$$x_{(k+1)} = y_1\alpha_1 + z_1 \qquad (20\text{-}1)$$

$$a_{12}x_1 + \cdots + a_{2k}x_k + a_{2(k+1)}x_{(k+1)} = -z_2 \bmod \alpha_2 \qquad (19\text{-}2)$$

$$x_{(k+2)} = y_2\alpha_2 + z_2 \qquad (20\text{-}2)$$

$$\vdots \qquad\qquad \vdots$$

$$a_{j1}x_1 + \cdots + a_{jk}x_k + \cdots + a_{j(k+j-1)}x_{(k+j-1)} = -z_j \bmod \alpha_j \qquad (19\text{-j})$$

$$z_{(k+j)} = y_j\alpha_j + z_j \qquad (20\text{-j})$$

$$\vdots \qquad\qquad \vdots$$

$$a_{m1}x_1 + \cdots + a_{mk}x_k + \cdots + a_{m(k+m-1)}x_{(k+m-1)} = -z_m \bmod \alpha_m \qquad (19\text{-m})$$

$$x_{(k+m)} = y_m\alpha_m + z_m\,. \qquad (20\text{-m})$$

In each case, the value of the check component z_j, of a mixed digit $x_{(k+j)}$ is determined by a formula involving the information digits and previously calculated mixed digits. Immediately after z_j has been determined, $x_{(k+j)}$ is calculated for possible use in calculating $z_{(j+1)}$. After the message has been completely encoded the following equations, analogous to (10), will be satisfied.

Let C_{ij} represent a_{ji} in equation (19-j). Then,

$$\sum_{i=1}^{k+m} C_{ij}x_i = 0 \bmod \alpha_j . \tag{21}$$

(Since $x_{(k+j)} = z_j \bmod \alpha_j$, substitution of $x_{(k+j)}$ for z_j in equation (19-j) will continue to satisfy the equation.)

At the decoder, equation (21) is changed to

$$\sum_{i=1}^{k+m} C_{ij}x_i' = c_j \bmod \alpha_j \tag{22}$$

In (22), x_i' represents the received value of x_i, and c_j represents the jth digit of the corrector. If all the digits have been correctly received, i.e., $x_i' = x_i$ for all values of i, then $c_1 = c_2 = \cdots = c_m = 0$; [see equation (21)]. If x_h had been received incorrectly so that $x_h' = x_h + 1$, but all other digits had been correctly received, then the value of c_j (the jth digit of the corrector) would be calculated in the following manner:

$$c_j \bmod \alpha_j = \sum_{i=1}^{k+m} C_{ij}x_i'$$

$$c_j \bmod \alpha_j = \sum_{i=1}^{k+m} C_{ij}x_i + C_{hj} = C_{hj} \tag{23}$$

Equation (23) proves that C_{hj} is actually the jth digit of the characteristic of x_h, because by definition, the characteristic of x_h is the value of the corrector when $x_h' = x_h + 1$, and all other digits have been correctly received. This means that the general term, C_{ij} of (21), is actually the jth digit of the characteristic of the ith digit and that this is a simple characteristic code.

For the case that $x_h' = x_h - 1$, the value of the corrector is such that if it were incremented, digit by digit, by the characteristic of x_h, the corrector would be composed only of zeros. Incrementing the corrector by the characteristic of x_h is equivalent to recalculating the corrector with x_h' increased by one, which in this case would amount to calculating the corrector for the case of a correctly received message. The latter is composed of all zeros [see (21)]. Thus, for the case of a single error of -1, the corrector is the characteristic complement of the digit which is incorrectly received. For a semi-systematic or systematic code, the characteristic complement is an m digit word whose jth digit is the complement modulo a_j of the jth digit of the characteristic.

Equation (20-j) shows that generally $\alpha_j\beta_j$ cannot exceed b. (An exception is given below.) The maximum value of y_j is $\beta_j - 1$ since y is a

digit in the number base β_j. The maximum value of z_j is usually $\alpha_j - 1$, since z_j is a digit in the number base α_j. Thus,

$$x_{k+j} = y_j\alpha_j + z_j \leqq b - 1, \tag{24}$$

$$(\beta_j - 1)\alpha_j + \alpha_j - 1 \leqq b - 1, \tag{25}$$

$$\alpha_j\beta_j \leqq b. \tag{26}$$

Equation (24) restates (19-j), and also states that the maximum value of any digit x, is $b - 1$, where b is the number base of the channel. In (25), the maximum values of y_j and z_j are substituted to yield the result shown in (26).

It was stated above that the maximum value of z_j is usually $\alpha_j - 1$. An exception occurs only in case z_j checks only itself and other mixed digits, the latter being restricted to fewer than $b - 1$ states. Under such circumstances, the value of z is sometimes restricted, so that even though z is calculated to satisfy a check, modulo α_j [see equation (19-j)], it cannot assume $\alpha_j - 1$ values. For example, a code for transmitting a single digit message over a decimal channel and permitting the correction of small errors, might use as the set of transmitted messages the digits, 0, 3, 6, 9. In this case, $\alpha = 3$ (any correct message satisfies the check $x = 0 \bmod 3$) and $\beta = 4$ since four different messages may be transmitted. In this case, z is restricted to the value 0 because the mixed digit checks only itself.

In order to correct single errors of ± 1, using a simple characteristic code, it is necessary and sufficient that every characteristic be different from every other characteristic, and that it also be different from the complement of every other characteristic.

The following rules and conventions may be used to derive a set of characteristics which meet the requirements for a simple characteristic semi-systematic or systematic code for correcting small errors for any base $b \geqq 3$ and an arbitrary length message. No set of conventions can be found which will lead to a more efficient code of this class, since the rules, not the conventions limit the efficiency of the code.

Rule 1. For an n digit message, including mixed digits, containing m mixed or check digits of which m_1 are associated with an even modulus, α, the inequality

$$(\alpha_1 \cdot \alpha_2 \cdot \ldots \cdot \alpha_m - 2^{m_1})/2 \geqq n \tag{27}$$

must be satisfied.

Rule 2. No characteristic may be repeated, i.e., each digit must have a characteristic different from that associated with any other digit.

Rule 3. Since the mth check is the last one to be calculated, and the

characteristic of the mth mixed digit must therefore contain only a single digit which is not 0, α_m must be greater than 2.

Convention 1. The various digits of a characteristic are arranged in a set order, i.e., C_{i1}, C_{i2}, $\cdots$, C_{im}. The first digit which is neither 0 nor $\alpha_j/2$ must be less than $\alpha_j/2$. There must be at least one such digit.

Convention 2. The characteristic of the jth mixed digit has a 1 in the jth position and 0's elsewhere, provided that $\alpha_j \neq 2$. If $\alpha_j = 2$, the characteristic of this mixed digit has a 1 in the jth and mth positions, and 0's elsewhere.

Rule 1 is required because the number of possible corrector states is $\alpha_1 \cdot \alpha_2 \cdot \ldots \cdot \alpha_m$, of which only those containing at least one digit which is neither 0 nor $\alpha/2$ can be associated with the $2n$ possible errors. The same reasons used for Rule 1 for the systematic code case are equally applicable here; a characteristic containing only the digits 0 or $\alpha_j/2$ in the jth position is not distinguishable from its complement.

Rule 2 is required to permit a unique identification of an incorrect digit.

Rule 3 is necessary to derive the sign of an error on the mth mixed digit.

The reasons for using Conventions 1 and 2 in the case of the systematic code are equally applicable in this case. For the case $\alpha = 2$, however, a special convention must be used to avoid a conflict with Convention 1.

The procedure for converting a set of characteristics into an error correcting code system is the same for a semi-systematic code as for a systematic code except that the following additional functions must be performed: the encoder must combine check states with information states to derive mixed digits, and the decoder must resolve mixed digits into information and check digits *after* it has performed its corrections.

By using these rules and conventions, the most efficient simple characteristic code can be determined. For messages of length n (including mixed or check digits), the following relations must be satisfied:

Let

$$P = \alpha_1 \cdot \alpha_2 \cdot \ldots \cdot \alpha_m ,$$

$$Q = \beta_1 \cdot \beta_2 \cdot \ldots \cdot \beta_m ,$$

$$m_1 = \text{number of even } \alpha\text{'s.}$$

Then:

$$(P - 2^{m_1})/2 \geqq n, \tag{28}$$

$$\alpha_i \beta_i \leqq b. \tag{29}*$$

* For exceptions, see above.

TABLE V — DECIMAL ERROR CORRECTION CODES

n	P	$\frac{10^m}{Q}$	$\alpha_1, \alpha_2, \ldots$	$\beta_1, \beta_2, \ldots$	$2n+1$
1	3	2.5	3	4	3*
2	5	5	5	2	5
3	10	10	10	1	7
4	10	10	10	1	9
5	15	16.7	5, 3	2, 3	11
6	15	16.7	5, 3	2, 3	13
7	15	16.7	5, 3	2, 3	15
8	20	20	10, 2	1, 5	17
9	25	25	5, 5	2, 2	19
10	25	25	5, 5	2, 2	21
11	25	25	5, 5	2, 2	23
12	25	25	5, 5	2, 2	25
13	30	33.3	10, 3	1, 3	27
14	30	33.3	10, 3	1, 3	29
15	40	40	10, 2, 2	1, 5, 5	31
16	40	40	10, 2, 2	1, 5, 5	33
17	50	50	10, 5	1, 2	35
18	50	50	10, 5	1, 2	37
19	50	50	10, 5	1, 2	39
20	50	50	10, 5	1, 2	41

* The single digit message containing the points 0, 3, 6, 9 is an exception to the inequality $\alpha\beta \leqq b$, because the mixed digit checks only itself.

For the most efficient code b^m/Q should be minimized. This term represents the ratio of the number of possible messages for an n digit message with and without error correction. This is normally at least as great as $2n + 1$, the number of possible corrections on such a message.

Table V shows the most efficient decimal codes of this type for an n digit message, for values of n from 1 to 20. Where two or more different codes are equally efficient, the code with the fewest mixed digits is shown. It is easy to convert from a code using two mixed digits with $\alpha_1 = 5$, $\alpha_2 = 2$, to one using a check digit with $\alpha = 10$, or to make the inverse conversion, and to show that both codes are equally efficient.

IV. SINGLE ERROR CORRECTION CODES, UNRESTRICTED ERROR

The problem of correcting an unrestricted error on one digit of a message must be divided into two categories, depending on whether b is a prime number or a composite number. As will be seen, the error correction problem for prime bases is considerably simpler than that for composite bases. The method for correcting errors in prime number systems was discovered by Golay,[6] although this did not come to the author's attention until after he had worked out the same method. The

adaptation to non-prime channel bases is believed to be novel. Since the adaptation makes use of the code for prime bases, both will be described.

4.1 *Prime Number Base, Single Unrestricted Error Correction Code*

This code depends upon a fundamental property of prime numbers, well known in number theory.[9] Let p represent a prime number and d, c, and w represent non-negative integers less than p, related by the expression:

$$dw = c \bmod p. \tag{30}$$

If $d \neq 0$, then d and c uniquely determine w.

In order to have a simple characteristic systematic code for correcting unrestricted errors, it is necessary and sufficient that the set of characteristics shall have the property that all multiples of all characteristics are distinct. Equation (30) implies a unique correspondence between multiples of a characteristic and the characteristic itself, if we consider c to be the multiple, d the multiplying factor and w a digit of the characteristic. An error, d, is simply identifiable if a known digit of a characteristic is always 1. If each characteristic is distinct from every other and if a sufficient number of check digits are available, a simple characteristic code can be obtained. In the following set of rules and conventions which may be used for deriving a set of characteristics for a simple characteristic systematic code for correcting single unrestricted errors, p represents the prime number base of the channel. The number base of the channel must be prime, and the length of the message is arbitrary. Since the rules and not the conventions limit the efficiency of the code, no other set of conventions may be found which will lead to a more efficient code of this class.

Rule 1. For an n digit message, m check digits are required and m must satisfy the inequality

$$n \leqq \frac{p^m - 1}{p - 1}. \tag{31}$$

Rule 2. Each digit must have a different characteristic.

Convention 1. The digits of a characteristic are arranged in a set order, i.e., $C_{i1}C_{i2} \cdots C_{im}$. The first digit which is not 0 must be 1.

Convention 2. The characteristic of the jth check digit has a 1 in the jth position and 0's elsewhere.

Rule 1 is required for a code for correcting single unrestricted errors since any digit must be correctable in one of $p - 1$ ways. This implies a minimum of $n(p - 1) + 1$ states for the corrector, one for each cor-

rection and one for the correct message. When m check digits are used, p^m corrector states are obtained.

Rule 2 and Convention 2 are the same for the single small error correction systematic codes. The same reasons apply for both cases.

Convention 1 is changed from the equivalent convention for the small error correction code, because the magnitude of the error, not only its sign, must be derivable for a code for correcting single unrestricted errors.

An encoder first encodes the message according to (32), where C_{ij} represents the jth digit of the characteristic of x_i,

$$\sum C_{ij}x_i = 0 \bmod b. \tag{32}$$

The decoder calculates the corrector using the following formula where x_i' represents the received value of x_1;

$$\sum C_{ij}x_i' = c_j \bmod b. \tag{33}$$

The decoder then examines the digits of the corrector in order. The first digit which is not 0 shows the magnitude, d, of the error. All digits are then divided by d (provided $d \neq 0$). (That division is unique, as shown by (30).) The result of this division is the characteristic of the incorrect digit, which is then corrected by subtracting d.

Consider a code for correcting a single unrestricted error in a six digit message for a base 5 channel:

$$6 \leqq \frac{5^m - 1}{4}. \tag{34}$$

A value of 2 for m will satisfy equation (34). The characteristics are 14, 13, 12, 11, 10 and 01, the last two being check digit characteristics, for x_1, x_2, x_3, x_4, x_5, and x_6 respectively. Here, x_1, x_2, x_3, and x_4 are information digits. The encoding formulas are:

$$x_1 + x_2 + x_3 + x_4 = -x_5 \bmod 5, \tag{35}$$

$$4x_1 + 3x_2 + 2x_3 + x_4 = -x_6 \bmod 5. \tag{36}$$

The decoding and correcting formulas are: (x_i' is the received value of x_i)

$$x_1' + x_2' + x_3' + x_4' + x_5' = c_1 \bmod 5, \tag{37}$$

$$4x_1' + 3x_2' + 2x_3' + x_4' + x_6' = c_2 \bmod 5. \tag{38}$$

The corrector is c_1c_2.

Suppose that a message 221321 is received as 224321. Then:

$$c_1 = 13 = 3 \bmod 5, \tag{39}$$

$$c_2 = 26 = 1 \bmod 5. \tag{40}$$

To find the characteristic of the digit, x_h, that was incorrectly received from the value of the corrector, (41) and (42) must be solved:

$$d\, C_{h1} = c_1 = 3 \bmod 5, \tag{41}$$

$$d\, C_{h2} = c_2 = 1 \bmod 5. \tag{42}$$

Because the first non-zero digit of any characteristic is 1, (41) can be solved for d since $C_{h1} = 1$. This yields the result, $d = 3$. Using this result, (42) is solved for C_{h2}; by inspection, $C_{h2} = 2$, since $3 \cdot 2 = 6 = 1 \bmod 5$. Thus the characteristic of the incorrect digit, $C_{h1}\, C_{h2}$, is 12, and the error d, is 3; x_3' must therefore be reduced by 3 to get the correct value. Since the message was received with x_3' too high by an amount 3, this result confirms our expected correction.

Any correction that is applied must be applied on a modulo b basis. For example, if a correction of -2 is indicated on a digit whose received value is 1, $1 - 2 = 4 \bmod 5$, which means that the digit is corrected to 4.

Codes of this type are restricted in their construction. No mixed digits may be used, and the number base must be prime. For the case of $n = [(p^g - 1)/(p - 1)] + 1$, $g + 1$ check digits are required [see (31)]. This means that the number of information digits for a message of this length is the same as for a message one digit shorter, which requires only g check digits. A comparable binary case is the Hamming Code example of an eight binary digit message (four information digits) compared with a seven digit message (also four information digits). In the binary case, the extra digit is useful for double error detection, but unfortunately, this is not the case for non-binary codes.

4.2 *Composite Number Base, Single Unrestricted Error Correcting Code*

The problem of correcting an unrestricted error on a single digit, working with a number base b, that is not a prime is much more difficult. Many relatively inefficient techniques exist. For example, characteristics containing only binary numbers (0 and 1) might be used; (this would amount to using the Hamming Code directly). This is obviously inefficient since the corrector associated with any single digit error of amount

d, would contain only the digits 0 and d, thus wasting most of the possible corrector values.*

It is possible to encode and decode using the prime factors of the number base, performing separate and independent corrections on each factor. This is also inefficient, since for many cases, information as to which digit is in error is found independently in two or more ways, while for certain values of the error, it can be found in only one way. Working with mixed digits and check bases, α lower than b, is not satisfactory since certain values of the error (α in particular) will never show up in a particular check. The technique used for primes will not work since multiples of two different characteristics may be identical; for example, base 10, characteristics 11 and 13, error 5, will both yield correctors of 55.

Another technique that is relatively efficient is, however, available. It involves performing all check, encoding and decoding operations in a number base p, where p is some prime number (usually, the lowest) that is equal to or greater than b. (In case b is a prime, we use the procedure outlined above, which is a special case of the procedure to be described below.)

The obvious difficulty in such a procedure is that while the information channel can only handle b levels, the check digits may assume p levels, corresponding to the required p check states. This dilemma can be resolved by adding an *adjustment digit*. The object of this digit is to permit check information to be transmitted in a base greater than b, the channel base. The idea of an adjustment digit can best be illustrated by an example. Suppose for a decimal channel, checks are performed in a unodecimal (base 11) code. Let γ represent the value corresponding to ten. (The consecutive integers in a unodecimal system are then 0, 1, 2, 3, $\cdots$, 9, γ, 10, 11, $\cdots$, 19, 1γ, 20, etc.) Suppose in a particular message, four check digits, z_1, z_2, z_3, z_4, calculated modulo 11 from decimal information digits are used, whose values are 1, 0, γ, 8. A fifth digit, z_0 is added such that the sums modulo 11 of $z_1 + z_0$, $z_2 + z_0$, $z_3 + z_0$, $z_4 + z_0$ are kept constant at 1, 0, γ, 8 respectively. There are eleven different words satisfying the condition: $[1, 0, \gamma, 8] = [(z_1 + z_0), (z_2 + z_0), (z_3 + z_0), (z_4 + z_0)]$. These are shown in Table VI. Of these words, six do not contain the digit γ, and so may be transmitted over a decimal channel. Thus, an adjustment digit permits check digits which are calculated in a number system of a higher base than b, to be transmitted over a base b channel. When an adjustment digit is used in base p for adjusting m digits so that transmission over a channel in base b is possible, a mini-

* A waste of corrector values is equivalent to an excessive number of check states for a message, which in turn implies an excessive number of check digits.

mum of $b - m(p - b)$ states are allowed for the adjustment digit. (For certain values of the check digits, more states could be allowed, but a code for utilizing these extra states becomes unwieldy.) For the case $b = 10$, $p = 11$, this turns out to be $10 - m$. At least one state must be available for each adjustment digit, to have a workable code.

The characteristic of an adjustment digit is determined in the following way: if an adjustment digit adjusts the jth check digit, then the jth digit of the characteristic of the adjustment digit is 1; otherwise, it is 0. The characteristic of all other digits may be derived using the rules described above for the prime number base channel, except that p, the prime number base of the code must be used instead of b, the number

TABLE VI — ILLUSTRATION OF ADJUSTMENT DIGIT

z_0	z_1	z_2	z_3	z_4
0	1	0	γ	8
1	0	γ	9	7
2	γ	9	8	6
3	9	8	7	5
4	8	7	6	4
5	7	6	5	3
6	6	5	4	2
7	5	4	3	1
8	4	3	2	0
9	3	2	1	γ
γ	2	1	0	9

base of the channel, for generating characteristics. A message is initially encoded using a value of 0 for an adjustment digit. Subsequently, if the adjustment digit always has at least q allowable states, it may be used to transmit one additional information digit, base q, of information. If the value of this information digit is y, the $(y + 1)$st lowest possible value of the adjustment digit (making the lowest value equivalent to $y = 0$) meeting the requirement that all adjusted check digits are no greater than $b - 1$ is transmitted. The adjustment digit in conjunction with its associated check digits conveys a digit, base q, of information.

In the example given above, $q = 6$ and if y is 4, the fifth lowest value of z_0, 7, is transmitted. The lowest value must be associated with $y = 0$. The values of $z_0z_1z_2z_3z_4$ that are sent over the decimal channel are 75431.

An example of such a code is one using a decimal channel working in a unodecimal base for the purposes of encoding and error correction. The word length, n, is twelve, nine decimal information digits, one octal (base

8) information digit associated with the adjustment digit, and two check digits. The characteristics are the following:

x_1 1γ	x_5 16	x_9 12
x_2 19	x_6 15	x_{10} 11 (adjustment digit)
x_3 18	x_7 14	x_{11} 10 (check digit)
x_4 17	x_8 13	x_{12} 01 (check digit)

Let z_{11} and z_{12} represent the values of the check digits x_{11} and x_{12}, originally derived from $x_1, x_2, \cdots, x_8, x_9$:

$$x_1 + x_2 + x_3 + \cdots + x_9 = -z_{11} \bmod 11, \tag{43}$$

$$\gamma x_1 + 9x_2 + 8x_3 + \cdots + 2x_9 = -z_{12} \bmod 11. \tag{44}$$

From z_{11} and z_{12}, the ten different words $(0, z_{11}, z_{12})$, $(1, z_{11} - 1, z_{12} - 1)$, $(2, z_{11} - 2, z_{12} - 2)$, $\cdots$, $(9, z_{11} - 9, z_{12} - 9)$ are formed. If y is the value of the octal information digit, the $(y + 1)$st such word, that does not contain the digit γ, is selected and transmitted as the last three digits of the message. For example, if $z_{11} = 2$, $z_{12} = 1$ and $y = 6$, the ten words are $(0, 2, 1)$, $(1, 1, 0)$, $(2, 0, \gamma)$, $(3, \gamma, 9)$, $(4, 9, 8)$, $(5, 8, 7)$, $(6, 7, 6)$, $(7, 6, 5)$, $(8, 5, 4)$, $(9, 4, 3)$; the word $(8, 5, 4)$ is selected since it is the seventh in the sequence that does not contain any γ's. Table VII shows the choice of the three last digits as a function of y, given $z_{11} = 2$, $z_{12} = 1$.

Formula (45) is used for calculating the corrector. Let C_{ij} represent the jth digit of the characteristic of x_i, c_j the jth digit of the corrector, and x_i' the received value of x_i. Then,

$$c_j = \sum_{i=1}^{12} C_{ij} x_i' \bmod 11. \tag{45}$$

The translation from corrector to correction is the same as if the original

TABLE VII — RELATION BETWEEN ADJUSTED DIGIT AND ASSOCIATED INFORMATION

y	x_{10}	x_{11}	x_{12}
0	0	2	1
1	1	1	0
2	4	9	8
3	5	8	7
4	6	7	6
5	7	6	5
6	8	5	4
7	9	4	3

message had been in a unodecimal code. (This has been illustrated in Section 4.1.)

The first step of the encoding procedure is to calculate the unadjusted check digits. Next, the adjusted check digits and adjustment digit are selected according to the value of y, the information digit associated with the adjustment. The message is then ready for transmission.

At the decoder, the message is first corrected as if it had been received as a unodecimal message. The information digits are then in their corrected states. Next, the adjustment digit and the check digits are examined and the inverse of the encoding process used to select a particular set of check and adjustment digits is used to reconstruct the value of y which originally controlled the selection. In the example given above, the values of x_{10}, x_{11}, x_{12} are 8, 5, 4 respectively; the decoder recognizes that this is the seventh lowest value of x_{10}, which means that the value of y, used in selecting x_{10} and the adjusted values of x_{11} and x_{12}, was 6.

The code described above is fairly efficient; about 90 per cent of the corrector values can be associated with corrections; the product of the information states and the check states is about 97 per cent of the total number of states of a twelve decimal digit word. Each of the above factors reduces the efficiency of the code below a possibly unattainable maximum. It will be noted, however, that this reduction is relatively small in both cases, and is very much lower than would be the case for any of the rejected schemes. The scheme is not difficult to instrument; relatively little additional equipment is required in addition to the basic equipment for instrumenting a simple prime number base channel, unrestricted single error correcting code system.

The method of adjustment digits is general and can be used for deriving a single error correction code for correcting unrestricted errors for any channel base. Any convenient prime check base, p, at least as great as b may be used, although the lowest will generally be the most efficient. The only requirements which must be fulfilled are that the number of states of the adjustment digit must be at least 1, and that at least two check digits must be associated with each adjustment digit. An adjustment digit associated with m check digits, working with a channel base b and a check base p, may have $b - m(p - b)$ different states.

V. SINGLE ERROR CORRECTION, DOUBLE ERROR DETECTION CODES FOR CORRECTING SMALL ERRORS

Single error correction, double error detection codes are very useful in situations where a message may occasionally be repeated. In order

for a correction code to be reasonably useful in a system with random noise or errors, the errors must be relatively infrequent, which makes double errors still more infrequent. If means are available for an occasional but very infrequent repetition of a message, a single error correction, double error detection code will increase the reliability of a digital system, since a message may be repeated if a double error is recognized.

This section will show how the ideas of the single error correction, double error detection Hamming Code may be combined with the ideas of semi-systematic single small error correction codes (described in Section III) to derive simple and efficient codes for correcting single small errors and detecting double small errors.

In order to derive a simple characteristic code for correcting single small errors, and detecting double small errors, a set of characteristics must be found having the property that the sum or difference of two characteristics or their complements or double the value of one characteristic or its complement be distinguishable from the value of any single characteristic or its complement. The sum of two characteristics represents the value of the corrector for a message with two errors of $+1$, $+1$, the difference represents two errors of $+1$, -1, the sum of their complements represents two errors of -1, -1; double a characteristic represents an error of $+2$, and double a complement represents an error of -2. To have a true single error correction, double error detection code for small errors, all these cases must be distinguished from the case of a single error or no error by making certain that the value of the corrector for any of these cases is different than the value of the corrector corresponding to any single error and no error.

Table VIII gives the characteristics used in the single error correction Hamming Code and the single error correction, double error detection Hamming Code for conveying four digits of information in a message containing seven or eight binary digits respectively.

An inspection ot Table VIII shows that the sum (performed without carries from column to column) of any two characteristics in the right part of the table is distinguished by having at least one 1 in the first three places and a 0 in the last place. This distinguishes it from any single characteristic since all characteristics have a 1 in their last place.

Some difficulties arise in trying to adopt such a scheme directly in a non-binary system. For the code to be efficient, an over-all check would have to be performed using a mixed digit; only two check states are required for an over-all parity check, and if $b > 3$, (b representing the number base of the channel) at least two information states are possible. But the over-all check digit, which performs a binary check, is not checked by any other digit. This means that although errors might be

detected in an over-all check digit, difficulties would be encountered in determining the direction of the correction, so that the information conveyed by the mixed digit could be used. Actually, means are available, for accomplishing an adaptation of binary techniques. These methods are described in Section VII but they are less straightforward than the ones described below.

For channels with base b, greater than 3, at least one check may be made using a check base, α_m, that is 4 or greater. If characteristics are used whose last digit (the digit associated with the α_m check) is always 1, and whose only other limitation is that each characteristic is different from every other characteristic, a satisfactory code is obtained. Single errors are corrected in the normal way. If the last digit of the corrector is 1 or $\alpha_m - 1$, the error is ± 1 respectively on the digit whose characteristic or whose characteristic complement is indicated by the corrector. If the last digit of the corrector is 2 or $\alpha_m - 2$, or the last digit is 0 and other digits are not all 0, a double error is indicated. If the entire corrector is made up of 0's, the message is correct as received.

TABLE VIII — CHARACTERISTICS FOR HAMMING CODES

Single Error Correction			Single Error Correction Double Error Detection
001	Check Digit	x_1	0011
010	Check Digit	x_2	0101
011	Information Digit	x_3	0111
100	Check Digit	x_4	1001
101	Information Digit	x_5	1011
110	Information Digit	x_6	1101
111	Information Digit	x_7	1111
	Over-all Check Digit	x_8	0001

An example is a code for a ten digit message, decimal base channel; eight decimal information digits, one mixed digit conveying binary message information (such as the sign of the decimal number) and quaternary (base 4) check information, and one check digit are transmitted in each message. Let x_1 and x_2 represent the mixed and check digit respectively, x_3 through x_{10} the information digits, y_1 the binary information conveyed by x_1, and z_1 the quaternary check information conveyed by x_1. The encoding formulas are:

$$2x_3 + 3x_4 + 4x_5 + 5x_6 + 6x_7 + 7x_8 + 8x_9 + 9x_{10} = -x_2 \bmod 10, \quad (46)$$

$$x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} = -z_1 \bmod 4, \quad (47)$$

$$x_1 = z_1 + 4y_1 . \quad (48)$$

Note that (46), (47) and (48) must be applied consecutively, in that order, since (47) cannot be applied without knowing x_2 obtained from (46), and (48) requires z_1, obtained from (47).

The characteristics are 01, 11, 21, 31, 41, 51, 61, 71, 81, 91 respectively; the complements of the characteristics are 03, 93, 83, 73, 63, 53, 43, 33, 23, 13 respectively. The corrector, c_1c_2, is calculated at the decoder by the following formulas (x_i' is the received value of x_i):

$$c_1 = \sum_{i=2}^{10} (x_i')(i - 1) \bmod 10 \tag{49}$$

$$c_2 = \sum_{i=1}^{10} x_i' \bmod 4 \tag{50}$$

Consider the example of a message with decimal information digits 3 7 5 2 0 6 5 2 and binary information digit 1. Then $x_2 = 3$, $z_1 = 3$, and $y_1 = 1$, yielding a value of 7 for x_1. The message is sent as 7 3 3 7 5 2 0 6 5 2. Suppose that the sixth digit is changed to 1 in transmission. Then the corrector has a value 53; this is the complement of the characteristic of the sixth digit and indicates that the sixth digit should be incremented by 1 according to the rules previously stated. If the sixth digit had been received as 1 and the seventh digit also received as 1 (an error of +1), then the corrector value would be 10, indicating a double error (see rules stated above).

If a multiple of 4 is used as α_m, the last digit of a characteristic may assume all odd values below $\alpha_m/2$. The rule then is that an even value of the last digit of the corrector, or a 0 for the last digit and other digits of the corrector not all 0, indicates a double error.

The following set of rules and conventions may be used with any base $b \geqq 4$, and any length of message, for deriving a set of characteristics for a semi-systematic code for correcting single small errors and detecting double small errors. Since the conventions restrict the efficiency of the code, it is conceivable that a different set of conventions will yield a more efficient code in some cases; (51) may be modified through the use of an alternate set of conventions.

Rule 1. No two digits may have identical characteristics.

Convention 1. Choose for α_m a multiple of 4. Let $\alpha_m/4 = g$.

Convention 2. The characteristic of the mixed digit associated with α_m contains a single 1 in the last position; the rest of its digits are 0.

Convention 3. The characteristics of the jth mixed or check digit contains a 1 in the last position, a 1 in the jth position and 0's elsewhere.

Convention 4. The characteristic of an information digit has an odd

number less than $\alpha_m/2$ in its last position. The rest of its digits are arbitrary.

Convention 5. The above conventions restrict the choice of characteristics. In order to have n distinct characteristics, m mixed or check digits, using check bases α_1, α_2, $\cdots$, α_m, are required, and inequality (51) must be satisfied:

$$n \leqq \alpha_1\alpha_2 \cdot \ldots \cdot \alpha_{m-1} \cdot g. \tag{51}$$

Codes may be derived using the above conventions only if $b \geqq 4$. For the ternary case, a relatively efficient code may be obtained by using one ternary digit as an over-all parity check digit. The rest of the message is in a single small error correction code, derived using the rules and conventions of Section III. Any single small error will lead to a failure of the parity check, and a double small error will lead to a failure of other checks but not the parity check.

No general solution has been found for deriving an efficient single error correction double error detection code for the unrestricted error case. Also, no general solution has been found for deriving an efficient multiple error correction code for the unrestricted error case. A reasonably efficient method has been found for correcting multiple errors in the more important small error case; this is discussed in Section 6.2.

VI. THE USE OF BINARY ERROR CORRECTION TECHNIQUES IN NON-BINARY SYSTEMS

In this section, methods for using binary codes for the correction of errors in a non-binary system are described. Although the single small error correction codes obtained in this manner are generally less flexible than the codes obtained in Section III, the class of multiple error correction codes described in Section 6.2 is the only reasonably satisfactory class of such codes that has been found. The codes described in this section are semi-systematic but are not simple characteristic codes.

6.1 *Single Small Error Correction Codes*

Binary codes are most conveniently used for correcting small errors (± 1). Suppose any digit, base b, has an associated pair of binary digits, arranged in such a way that a change of ± 1 in the base b digit will change only one of the two binary digits. For $b = 10$, an association such as the one shown in Table IX might be used. For example, if a 6 is received as a 7, the associated binary message would indicate that the second of the binary digits is incorrect; a 7 can be corrected

TABLE IX — ASSOCIATED BINARY DIGITS FOR CORRECTION OF SMALL ERRORS

Decimal Digit	Associated Binary Digits
0	00
1	01
2	11
3	10
4	00
5	01
6	11
7	10
8	00
9	01

TABLE X — REFLECTED QUIBINARY CODE

Decimal Digit	Quinary Component	Binary Component	Associated Binary Digits
0	0	0	00
1	0	1	01
2	1	1	11
3	1	0	10
4	2	0	00
5	2	1	01
6	3	1	11
7	3	0	10
8	4	0	00
9	4	1	01

to an 8 or a 6, but only the correction to 6 would correspond to a change in the second binary digit of the associated binary message.

If the first of the associated binary digits is the odd or even indication of a quinary component of a decimal digit, a decimal digit can convey ten states rather than the four states of the associated binary digits. The combination of binary and quinary digits shown in Table X may be called a reflected quibinary code because of its analogy with the reflected binary code.*

If a method were available for transmitting without error (e.g., by using an error correcting code) a message composed of the associated binary digits in a base b code, small errors could be corrected in the base b digits.

An examination of Table X for resolving a decimal digit into binary and quinary components, reveals that a change of ± 1 on any decimal

* The reflected binary code has the property that each increment changes only one binary digit; for example, the eight successive words of a three binary digit reflected binary code are 000, 001, 011, 010, 110, 111, 101, 100.

digit will change only one of these two components. Further, an error corresponding to a change in the quinary component can be uniquely corrected if the error in the decimal digit is assumed to be ± 1. For example, if a received 6 is discovered to have an incorrect quinary component, only a decrease in the quinary component making the decimal digit 5 is a possible correction, since an increase in the quinary component would correspond to the decimal digit 9, a change of more than ± 1 from 6.

A system is shown in Fig. 2 for taking advantage of these properties.

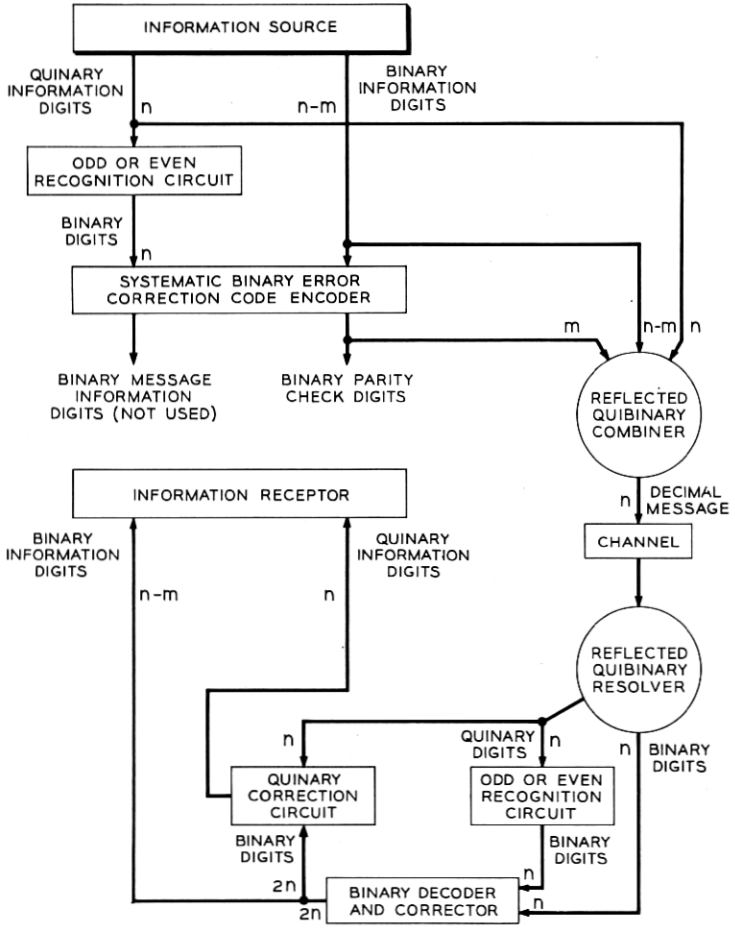

Fig. 2 — Use of binary codes with a decimal channel.

In this example, an information source generates n quinary and $n - m$ binary information digits for each message. All quinary digits go through an odd or even recognition circuit to be converted into binary digits for the purpose of generating a binary error correction code message. These binary digits and the binary digits generated by the information source are fed into a systematic binary error correction code encoder whose output is a binary message containing $2n$ digits, of which m are parity check digits. This output is divided into two parts, $2n - m$ original inputs to the encoder unchanged by the encoding process (this is a systematic encoder which does not change information digits in encoding), and m parity check digits.

The m parity check digits are then combined with m of the quinary information digits through the use of the reflected quibinary combiner to form m of the decimal digits of the decimal message that is transmitted; the other decimal digits are formed by combining the $n - m$ binary information digits with the rest of the quinary information digits.

The decimal message is transmitted over the noisy channel and arrives with one or more (a number limited by the choice of the binary code) errors of ± 1 on decimal digits. It is fed into a reflected quibinary resolver which resolves decimal digits into binary and quinary components in accordance with the reflected quibinary code (Table X). The quinary digits are then fed into an odd or even recognition circuit to form binary digits; these and the binary outputs of the resolver are fed into a binary decoder and corrector, working with the same code as the binary encoder. The output of this corrector should correspond to the output of the original binary encoder.

In the decoder, the binary digits are corrected. When the binary digit derived from a quinary digit is corrected, however, the quinary digit is not yet correct. The correction of the quinary digit is performed by examining both the corrected binary digit derived from the quinary digit and the corrected binary digit which was derived from the same decimal digit as the quinary digit in question. The rules for correcting the quinary digit are given in Table XI.

As an example, consider the application of a Hamming Code for transmitting ten binary digits in a fourteen binary digit message.

Using a code of this type, single errors of ± 1 may be corrected in a seven digit decimal message, transmitting seven quinary digits of information and three binary digits of information. The characteristics required for a fourteen binary digit Hamming Code message are shown in the first column of Table XII.

TABLE XI — CORRECTING QUINARY DIGITS

Q	B_1	B_2	Correction of Quinary Digit
Even	0	0	None
Even	0	1	None
Even	1	0	−1
Even	1	1	+1
Odd	0	0	+1
Odd	0	1	−1
Odd	1	0	None
Odd	1	1	None

TABLE XII — BINARY CODE USED FOR CORRECTING DECIMAL MESSAGE

Binary Characteristics					a	b	Position in Decimal Message
0	0	0	1	Parity Check Digt	(0)	(0)	Binary comp. of 1st digit
0	0	1	0	Parity Check Digit	(0)	(0)	Binary comp. of 2nd digit
0	0	1	1		1	1	Binary comp. of 3rd digit
0	1	0	0	Parity Check Digit	(1)	(1)	Binary comp. of 4th digit
0	1	0	1		0	0	Binary comp. of 5th digit
0	1	1	0		0	0	Binary comp. of 6th digit
0	1	1	1		1	3	Quinary comp. of 7th digit
1	0	0	0	Parity Check Digit	(1)	(1)	Binary comp. of 7th digit
1	0	0	1		1	3	Quinary comp. of 6th digit
1	0	1	0		0	2	Quinary comp. of 5th digit
1	0	1	1		0	4	Quinary comp. of 4th digit
1	1	0	0		1	1	Quinary comp. of 3rd digit
1	1	0	1		1	3	Quinary comp. of 2nd digit
1	1	1	0		0	0	Quinary comp. of 1st digit

To illustrate the method completely, a strictly binary example will first be illustrated, then a related decimal example. In column a of Table XII, the digits of a binary message are indicated and in column b, the binary and quinary information digits. The values of the parity check digits, which are shown in parentheses, are calculated by the usual formula. Let C_{ij} represent the jth digit of the characteristic of the ith digit (including parity check digits):

$$\sum_{i=1}^{14} x_i C_{ij} = 0 \bmod 2. \tag{52}$$

This formula applies for all values of j and in this case will yield four implicit equations each with one unknown term, the value of the parity check digit. Using the given values of the binary information digits, the values of the parity check digits are calculated. These are shown in parentheses in Table XII.

The binary message is

$$0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 1\ 1\ 0.$$

For this example, the quinary components (quinary information digits) of decimal digits are chosen odd if the corresponding digit of the binary example is 1, even if that digit is 0. The binary and quinary components are then combined by the rules of the reflected quibinary code to form the decimal digits 0 7 2 9 4 7 6. For example, the quinary and binary components of the fifth digit are 2 and 0, respectively; the decimal digit which has these components is 4, the fifth decimal digit of the message.

Consider the binary case. Suppose that the message is mutilated in transmission so that the tenth digit is received incorrectly. The message is mutilated from

$$0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 1\ 1\ 0$$

to

$$0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 0.$$

The decoder and corrector calculates the corrector by

$$c_j = \sum_{i=1}^{14} x_i' C_{ij} \bmod 2. \tag{53}$$

In this formula, c_j is the jth digit of the corrector and x_i' the received value of x_i. In this example the corrector is 1 0 1 0, which means that the tenth digit, which has this characteristic, is wrong and should be changed to 0.

The corresponding error in the decimal example is a change in the fifth digit from 4 to 3. If the message 0 7 2 9 3 7 6 is received, the resolver and quinary to binary converter delivers the message

$$0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 0$$

to the decoder instead of

$$0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 1\ 1\ 0$$

corresponding to the correct message. The corrected binary message is produced at the output of the decoder and corrector. When the quinary and binary components of the fifth digit are examined by the quinary correction circuit, the following inputs exist:

Received quinary digit	1 (Odd) (quinary component of received decimal 3)
Corrected binary digit derived from quinary	0 (B_1)
Corrected binary digit from same decimal number	0 (B_2).

Table XI shows that the quinary digit must be increased by 1 to 2, which combined with the binary 0 conveyed by the same decimal digit yields a decimal value of 4, the original transmitted value.

The best semi-systematic simple characteristic code for correcting single small errors in a seven digit message allows 6×10^5 possible messages in a seven digit message (see Table V), whereas this code allows 6.25×10^5. This code is therefore slightly more efficient. In addition, this code has the special advantage that any error of ± 2 on one digit is recognizable since the corrector will have a value of 1111 for the associated binary message. (An inspection of the choice of characteristics and assignment of characteristics to the two components of any decimal digit will confirm this.)

This general technique can be applied to any base b channel, provided

TABLE XIII — COMPONENTS OF QUINARY DIGITS

Mixed Digit			Information Digit		
Quinary Digit	Info. Comp.	Check Comp.	Quinary Digit	Binary Comp.	Ternary Comp.
0	0	0	0	0	0
1	0	1	1	1	0
2	1	1	2	1	1
3	1	0	3	0	1
4	not used	not used	4	0	2*

* If quinary information is initially generated, the combination (1, 2) will not occur.

that b is greater than 3. For odd bases, the digits which convey a parity check component and an information component cannot be utilized efficiently since one state of the base b digit is not available. For example, using a base 5, (see Table XIII), only two information and two parity check states may be conveyed by one digit, since the use of a third information state would require at least six states for the mixed digit. In the case of information digits, however, all states can be used. In the quinary example, the resolution of a digit into two components and the subsequent recombination is subject to the restraint that one of the combinations (1, 2) will not occur, which can be assured if the information source generates quinary digits.

For the case of high redundancy codes having the property that the associated binary code contains more than 50 per cent parity check digits (corresponding to a negative value of $n - m$ in Fig. 2), at least some of the base b digits must convey two or more parity check digits.

This can be easily accomplished: a decimal digit can convey three

TABLE XIV — DECIMAL DIGIT CONVEYING THREE BINARY DIGITS

Decimal Digit	Binary Components
0	0 0 0
1	0 0 1
2	0 1 1
3	0 1 0
4	1 1 0
5	1 1 1
6	1 0 1
7	1 0 0
8	not used
9	not used

parity check digits if a simple reflected binary code correspondence between binary and decimal digits is maintained as shown in Table XIV.

An extension of this idea is the encoding of the original information (i.e., the information that is shown coming out of the information source in Fig. 2) in some error detection or correction code. For example, the decimal to reflected quibinary code resolver will cause both components to be incorrect if an error of ± 2 in a decimal digit occurs. In this case, the system shown in Fig. 2 will automatically make a correction on the decimal digit of either $+2$ or -2 depending upon the value of the received decimal digit, and provided a double error correction binary code is used. Such a correction will be incorrect about half the time. If the received binary digit is compared to the corrected binary digit and the received quinary digit is compared to its corrected odd or even digit, an error of ± 2 can be detected without changing the code. If one extra binary check digit, treated as an information digit by the encoder and decoder, is transmitted in the message, this binary digit can convey the information necessary for determining the sign for a correction of ± 2, provided that only one such correction is required for any one message. A rule for determining the value of this digit is:

$$\begin{aligned} B_c &= 0 \quad \text{if} \quad \sum_{i=1}^{n} q_i = (0 \text{ or } 1) \bmod 4, \\ B_c &= 1 \quad \text{if} \quad \sum_{i=1}^{n} q_i = (2 \text{ or } 3) \bmod 4, \end{aligned} \tag{54}$$

where q_i represents the ith quinary information digit, and B_c represents the special check digit. If the received message contains one error of ± 2 on a digit, two possible corrections may be made on the quinary component of this digit; ± 1. Obviously, only one of these corrections will satisfy the equation for determining B_c since the two possible corrected values of q are two units apart.

Note that the associated binary codes for performing such a correction must have the property that two binary digits may be corrected since an error of ± 2 corresponds to incorrect values for two associated binary digits. If the noise is such that errors of ± 2 are not very unlikely, it may be desirable to place the binary and the quinary components of any one decimal digit in a different binary error correction code word so as to make the errors independent. In a seven decimal digit message, as an example, the quinary components of the first four decimal digits can be used to generate parity check digits which are conveyed by the binary components of the last three decimal digits. The binary component of the fourth decimal digit (this might be B_c) and the quinary components of the last three decimal digits generate parity check digits conveyed by the binary components of the first three decimal digits. Two separate binary error correction code messages are then conveyed by a single seven digit decimal code message. Each message is in a four information digit, three parity check digit Hamming Code. Through the use of this code, one error in the binary component of any decimal digit, and one error in the quinary component of any decimal digit may be corrected.

In certain cases, the quinary digits themselves might be encoded in an error correction code for single unrestricted errors before the binary process is carried out. This is helpful chiefly for occasional large errors, leading to initial miscorrections.

The variations based upon the principles described, which can be applied to any channel, provided $b \geqq 4$, including the pyramiding of one code scheme upon another, are almost endless. Generally, the last encoding and first decoding step should be able to correct many more errors than the first encoding step. For example, if quinary components are encoded in single unrestricted error correction quinary code, the binary code should probably be a triple or quadruple error correction code; otherwise a correction may not correspond to the most probable error condition, and the correction scheme loses its effectiveness.

These techniques cannot be conveniently applied to the ternary channel, since a ternary digit cannot be resolved into two components efficiently.

6.2 *Multiple Small Error Correction Codes*

One limitation of the above techniques is the requirement for a systematic binary code; i.e., a code in which some of the binary information digits are transmitted directly, and others are determined by parity checks on information and previously calculated check digits. These

TABLE XV — REED-MULLER CODES — 256 DIGIT MESSAGE

Number of Digits of Information per Message	Number of Errors Correctable per Message
255	0
247	1
219	3
163	7
93	15
37	31
9	63
1	127

systematic codes are conveniently applicable only to the correction of single errors and a few special cases of multiple errors.

The Reed-Muller[10] codes are not systematic codes, ("systematic" being used in the narrow sense indicated above, not in the sense of Hamming[11]), but offer the advantage that multiple error correction is relatively straightforward. For this reason, it is desirable to find some way of adapting the binary Reed-Muller codes for correcting a number of small errors in non-binary codes.

To explain the nature of the Reed-Muller codes completely is beyond the scope of this paper; a list of their important features is sufficient. This is:

1. The length of a message is 2^k binary digits for the simpler versions of the code.

2. If C_c^d represents the number of combinations of d items taken c at a time, and $C_c^d = d!/[c!(d-c)!]$, then $2^k - \sum_{i=0}^{m} C_{k-i}^k$ information digits may be transmitted correctly in a message containing 2^k digits, if no more than $2^m - 1$ errors occur in the messages; 2^m errors are detected but they are not always correctable. The Reed-Muller codes for correcting a large number of errors will frequently correct more than $2^m - 1$ errors, and will always correct $2^m - 1$ or fewer errors.

These values are given for a 256 digit message in Table XV.

3. Each digit of the transmitted message is a parity check of a group of digits from the information source; the message cannot be broken down into information digits and check digits.

4. The decoding is accomplished by a number of majority decisions among different groups of message digits.

A technique will be described for using a Reed-Muller code efficiently to correct a number of small (± 1) errors for any code base b that is a multiple of 2, and also, at a small sacrifice of efficiency, a number of larger errors.

A theorem, stating that any code which is generated by a set of parity

checks will contain the same set of allowable messages as some systematic code, was proved by Hamming.[11] In particular, such a theorem indicates that a Reed-Muller code will contain the same set of allowable messages as some systematic code. This was also proved by Slepian,[12] who has given a simple method of deriving a systematic code generating the same set of messages as a Reed-Muller code. For convenience, such a code will be called an SERM code (Systematic Equivalent Reed-Muller code).

A Reed-Muller decoder serves to derive the information digits from a message in Reed-Muller code which may have been mutilated by noise. If a Reed-Muller decoder is followed by a Reed-Muller encoder, the combination serves as a noise eliminator (provided the noise is within the correction bounds of the code), since the output of the encoder is the noiseless Reed-Muller code message that is equivalent to the noisy message that entered the decoder. This property is useful since it means that any message, drawn from the set of Reed-Muller code messages, which has not been mutilated outside the bounds set up by a particular Reed-Muller code, will be restored to its original form, by a Reed-Muller decoder followed by a Reed-Muller encoder. Since an SERM code will produce only messages included in the set of messages of the corresponding Reed-Muller code, the SERM code can be used in conjunction with a Reed-Muller decoder and encoder to permit transmission over a noisy channel in a systematic code.

The two systems shown in Fig. 3 are therefore equivalent in their error correction properties. In both cases, messages from the set of Reed-Muller code messages are sent, and since the same decoder is used initially, both systems will correct errors in the received message in the same manner. The Reed-Muller encoder in the second system is required because a Reed-Muller decoder does not correct a message but derives information digits from the received message directly. The derived information digits, however, necessarily correspond to some corrected form of the received message and, in effect, the decoder performs the same correction as it would perform by deriving the corrected form of the message first.

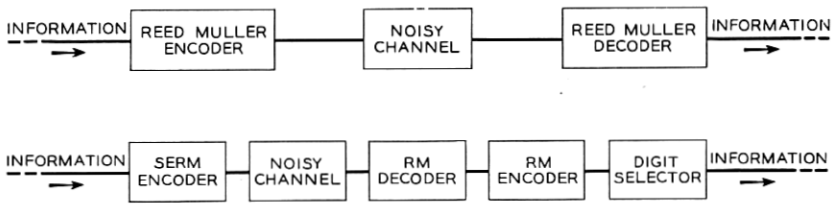

Fig. 3 — Equivalent systems using SERM and Reed-Muller codes.

TABLE XVI — MULTIPLE SMALL ERROR CORRECTION CODE USING SERM CODES WITH DECIMAL CHANNEL

Message Length	Information Digits (Equivalent Decimal Digits)	Check Digits (Equivalent Decimal Digits)	No. of Small Errors Correctable per mes.
128	127.7	.3	0
128	125.3	2.7	1
128	116.9	11.1	3
128	100.0	28.0	7

This means that a Reed-Muller code can be adapted to the system shown in Fig. 2. The Systematic Binary Error Correction Code Encoder is simply an SERM encoder; this is permissible since the SERM codes are systematic. The Binary Decoder and Corrector is simply a Reed-Muller decoder followed by a Reed-Muller encoder. Everything else remains unchanged.

This scheme offers flexibility for the correction of large numbers of small errors. Proper initial error correction encoding of the original information digits will permit correction of a small number of large errors.

Table XVI shows some typical cases of the correction of many small errors in a decimal message as a function of the number of information and check digits in a message of constant length. For convenience, everything is shown in equivalent decimal digits, even though in the actual code, binary and quinary information digits are used. Only the first few entries are considered, since the message composed exclusively of the digits 0, 3, 6, 9 in which any number of small errors in a decimal channel may be corrected (this code is described by the first entry of Table V) is more efficient than the codes corresponding to subsequent entries on Table XVI. This code, which is very easy to instrument, will transmit the equivalent of 77 decimal digits in a 128 decimal digit message.

One problem not efficiently solved by these techniques is the multiple-error correction ternary channel problem. A technique which can be used is a code identical to the regular binary Reed-Muller Code, except that all equations will be modulo 3 instead of modulo 2. In decoding, this will sometimes require subtraction instead of addition; in modulo 2 equations there is no difference between these operations, but in modulo 3 equations, the two operations are distinct. The same procedure can be used for correcting multiple unrestricted errors in any base.

VII. ITERATIVE CODES

All the codes described above have one disadvantage; occasional excessive noise will yield a non-correctable message. In order to approach

error free transmission, some iterative coding procedure may be used. This problem has been solved by Elias.[13] His methods are directly applicable to non-binary codes, since nothing restricts the digits to binary values.

In order to minimize the complexity of an iterative coding procedure, systematic codes are desirable. The advantages of the Reed-Muller code are significant however, especially for the case of a relatively noisy channel. A sound procedure for a binary channel would therefore be to use SERM codes, (see Fig. 3); such codes are more efficient than iterated Hamming Codes in a relatively noisy channel.

VIII. SUMMARY AND ANALYSIS

Many codes have been presented in this paper, all constructed by some combination of procedures involving linear congruence or modulo equations.

In most cases, more efficient codes exist. Exhaustive procedures exist for deriving maximum efficiency codes, although the codes derived in this manner usually require an extensive codebook, both at the encoder and at the decoder. Even for simple single error correction binary codes, the most efficient code is not always a systematic code. For example, the best systematic single error correction binary code working with an eight digit message has only 16 different allowable messages; it is known[14] that a non-systematic code with at least 19 allowable messages exists.

In the case of non-binary codes, the situation is somewhat worse. Very few of the codes given in this paper take advantage of the fact that, for most situations, a digit that is incorrectly received as 0 or $b - 1$ is usually corrected only in one direction and no need exists to specify whether the correction is ± 1. Most of the codes are arranged so that any received digit may be corrected either positively or negatively. No codes have been found which take full advantage of such a property, other than codebook codes, except for isolated instances of short message codes having symmetrical properties. For example, the single digit, single small error correction decimal code having 0, 3, 6, 9 as the allowable messages takes full advantage of this property, and is, at the same time, a true semi-systematic code.

It is extremely difficult to find the ultimate limits of efficiency of codebook codes. The exhaustive procedures are totally impractical except for very short messages. If an analysis is restricted to codes which do not take advantage of the property that certain values of digits may be corrected in only one direction, and it is assumed that each possible message is mutilated to the same number of incorrect messages, one

limit to the efficiency of codes may be found. This limit can be derived from the fact that an error correction code decoder and correction circuit must be able to convert any message which contains errors within the bounds of the correction performed by the code, into the value of the message as originally transmitted, or must be able to derive the original information which was fed into the encoder. Thus, if each message may be mutilated in w ways, and still be corrected, then at least w messages must be associated with each allowed message. This is indicated diagrammatically in Fig. 4. The messages produced by the encoder are shown at the left; each one fans out to $w - 1$ mutilated messages plus the original message. The decoder converts any of these w messages into the original message.

The value of w can be determined by taking all possible combinations of errors that can be corrected by a coding system. For example, for a code system which can correct up to $(d - 1)/2$ small errors in different digits in an n digit message, w is given by

$$w = \sum_{i=0}^{(d-1)/2} C_i^n 2^i, \tag{55}$$

where d is the minimum distance between messages, and

$$C_i^n = \frac{n!}{(n - i)!i!}.$$

This equation merely signifies that w is the sum of all combinations of positive and negative (accounting for the 2 term) errors in up to $(d - 1)/2$ different digits out of n digits. For single errors, $w = 2n + 1$.

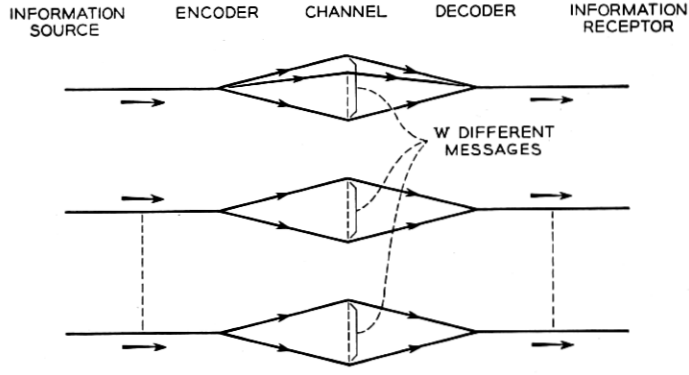

Fig. 4 — Graphical representation of an error correction code.

The number of different messages that can be produced by the encoder must be no greater than b^n/w, subject to the above restriction, b^n representing the maximum number of messages that the decoder may receive as an input. If only systematic and semi-systematic codes are considered, the number of messages is limited to multiples of powers of b and of the information component base β of mixed digits. The number of check states must be at least as large as w, so that w different correctors may be calculated and associated with w different corrections.

Subject to the above restrictions, the following statements may be made.

1. The systematic single small error correction codes derived using the rules of Section III are the most efficient systematic single small error correction codes possible. For those codes in which the two sides of inequality (8a) are equal, no code, not even a non-systematic code, is more efficient.

2. The systematic single unrestricted error correction codes derived using the rules of Section 4.1 are the most efficient systematic single unrestricted error correction codes. For those codes in which the two sides of inequality (31) are equal, no code is more efficient.

3. No codes are more efficient than those semi-systematic codes, derived using the rules of Section III, for which the two sides of inequalities (28) and (29) are equal and $m_1 = 0$. It is difficult to make more general statements about semi-systematic codes, because special techniques (such as those of Section VI), not all of which are known, may be used with these codes.

For multiple error correction codes, other techniques are both simpler and more efficient than the straight systematic and semi-systematic techniques described in Sections III, IV and V. One such scheme has been described in detail in Section VI. No codes have been found which approach the limit set by w, but the codes described in Section 6.2 are moderately efficient.

Throughout this paper, all techniques which involve vast complications at the expense of slight additional efficiency have been avoided. Codebook methods are always possible. If a technique is almost as complicated as a codebook technique with only slightly greater efficiency than a simple technique, the simple technique would always be used in practice, and the codebook satisfies the mathematical and theoretical requirements. In a sense, a really complicated technique is only useful for deriving a better lower limit for the maximum efficiency of a codebook code. In the non-binary case, however, a codebook system is considerably more efficient than any code system which does not take ad-

vantage of the fact that all transmitted messages are not mutilatable to an equal number of correctable received messages.

From the point of view of deriving lower limits to the maximum efficiency of a codebook technique, such a consideration is vital. Except for a few relatively trivial cases, no codes have been found which take significant advantage of the above consideration, for deriving such a limit.*

IX. CONCLUSION

In this paper, techniques have been presented for deriving error correction codes for non-binary systems. None of the methods presented are overly complicated, nor do they require excessive storage capacity for either the encoding or decoding and correction system.

The codes are sufficiently simple so that their use with a non-binary storage system may be considered, and the development of such a storage system should not be stopped because a system without flaws or not subject to noise cannot be realized.

An important disadvantage of using error correction codes with such a system is the time requirement. Correction usually requires a significant amount of time. This is probably one reason why the Hamming Code is not used more extensively. The more advanced and complicated codes, such as the Reed-Muller Codes, suffer particularly from the amount of time required for a correction. The codes described in this paper are therefore probably best suited to medium or low speed storages, which are not read too frequently.

A study of this type may be of some interest to those who have been considering the use of multi-state devices for building switching systems and computers, since this paper represents a study of a typical problem. Certain lessons may be derived from this study:

1. Restriction to a single number base for all operations is a severe handicap. The more advanced codes presented in this paper, require extensive use of different number base operations. The ability, inside the computer, to change number bases for different operations, may well be useful.

2. Different problems are best solved using different number bases. For example, the use of an even number base is desirable for multiple small error correction codes, while the use of a prime number base is desirable for correcting single large errors. It is the author's opinion that

* Note that this restriction has less significance in the case of binary codes. In a symmetrical channel with only two available signals, each value of a digit may be changed in as many ways, namely, one, as every other.

number bases which are the product of several small factors are best. Suggested values are six, ten and twelve. Number bases with two different prime factors, may offer an advantage, since they permit simple translation and change of number base among at least three different numbers.

In the comparison between binary and non-binary error correction codes, the following observations may be made:

1. Keeping the amount of information per message fixed, a binary single error correction code is less efficient than a non-binary single small error correction code, provided b, the channel base, is greater than three, but is more efficient than a non-binary single unrestricted error correction code.

2. Non-binary codes are slightly more complicated to implement than binary codes; this applies to multiple error correction codes as well as to single error correction codes. The amount of added complication is in no case really extensive.

It was initially hoped that this study might also produce some additional binary error correction techniques. One such technique was discovered: the use of a systematic equivalent Reed-Muller code to approach error free coding (see Section VII).

Finally, the author wishes to express the hope that further work on non-binary systems will be encouraged by this study.

ACKNOWLEDGEMENTS

This work was performed under the part-time Graduate Study Plan of Bell Telephone Laboratories at the Columbia University School of Engineering under the guidance of Prof. L. A. Zadeh. The author wishes to acknowledge the help of Prof. Zadeh, both in the selection of a dissertation topic and in the subsequent guidance of the study. In addition, a number of helpful discussions with C. Y. Lee and A. C. Rose helped to guide the research into a study of the most significant problems in the field.

REFERENCES

1. R. W. Hamming, Error Detecting and Error Correcting Codes, B.S.T.J., **29**, p. 147–160, April, 1950.
2. Of Current Interest, Elec. Engg., p. 871, Sept., 1956.
3. C. Y. Lee and W. H. Chen, Several-Valued Combinational Switching Circuits, Trans. A.I.E.E., **75**, Part I, p. 278–283, July, 1956.
4. D. Slepian, A Class of Binary Signaling Alphabets, B.S.T.J., **35**, p. 203–234, Jan., 1956.

5. Hamming, *op. cit.*, Section 1.
6. M. J. E. Golay, Notes on Digital Coding, Proc. I.R.E., **37,** p. 657, June, 1949.
7. W. Keister, A. E. Ritchie, and S. H. Washburn, *The Design of Switching Circuits*, D. Van Nostrand Co., Inc., New York, 1951, p. 316.
8. M. J. E. Golay, Binary Coding, 1954 Symposium on Information Theory, Trans. I.R.E., **PGIT-4,** p. 23–28, Sept., 1954.
9. G. H. Hardy and E. M. Wright, *An Introduction to the Theory of Numbers*, Oxford University Press, Oxford, 1954, p. 51, Theorem 57.
10. I. S. Reed, A Class of Multiple-Error-Correcting Codes and the Decoding Scheme, 1954 Symposium on Information Theory, Trans. I.R.E., **PGIT-4,** p. 38–49, Sept., 1954.
11. Hamming, *op. cit.*, Section 7.
12. D. Slepian, A Note on Two Binary Signaling Alphabets, Trans. I.R.E., **IT-2,** p. 84–86, June, 1956.
13. P. Elias, Error Free Coding, 1954 Symposium on Information Theory, Trans. I.R.E., **PGIT-4,** p. 29–38, Sept., 1954.
14. V. I. Siforov, On Noise Stability of a System with Error-Correcting Codes, Trans. I.R.E., **IT-2,** p. 109–115, Dec., 1956. See Table II, Column 8.