University of Kent

Kent Academic Repository

Khadem-Zadeh, A. (1980) *Anticodes and error-correcting for digital data transmission.* Doctor of Philosophy (PhD) thesis, University of Kent.

Downloaded from
https://kar.kent.ac.uk/94459/ The University of Kent's Academic Repository KAR

The version of record is available from
https://doi.org/10.22024/UniKent/01.02.94459

This document version
UNSPECIFIED

DOI for this version

Licence for this version
CC BY-NC-ND (Attribution-NonCommercial-NoDerivatives)

Additional information
This thesis has been digitised by EThOS, the British Library digitisation service, for purposes of preservation and dissemination. It was uploaded to KAR on 25 April 2022 in order to hold its content and record within University of Kent systems. It is available Open Access using a Creative Commons Attribution, Non-commercial, No Derivatives (https://creativecommons.org/licenses/by-nc-nd/4.0/) licence so that the thesis and its author, can benefit from opportunities for increased readership and citation. This was done in line with University of Kent policies (https://www.kent.ac.uk/is/strategy/docs/Kent%20Open%20Access%20policy.pdf). If you ...

Versions of research works

Versions of Record
If this version is the version of record, it is the same as the published version available on the publisher's web site. Cite as the published version.

Author Accepted Manuscripts
If this document is identified as the Author Accepted Manuscript it is the version after peer review but before type setting, copy editing or publisher branding. Cite as Surname, Initial. (Year) 'Title of article'. To be published in ***Title of Journal*** , Volume and issue numbers [peer-reviewed accepted version]. Available at: DOI or URL (Accessed: date).

Enquiries
If you have questions about this document contact ResearchSupport@kent.ac.uk. Please include the URL of the record in KAR. If you believe that your, or a third party's rights have been compromised through this document please see our Take Down policy (available from https://www.kent.ac.uk/guides/kar-the-kent-academic-repository#policies).

ANTICODES AND ERROR-CORRECTING CODES FOR DIGITAL DATA TRANSMISSION

by

A. Khadem Zadeh

A dissertation submitted for the degree of Doctor of Philosophy in the Faculty of Natural Sciences at the University of Kent at Canterbury.

The Electronics Laboratories,
September, 1980

To my father-in-law

Hosain Namazian

May this be a small sign of
my gratitude for his
encouragement and support
during the time spent on my
research.

ACKNOWLEDGEMENTS

I would like to express my grateful thanks to Dr. P. G. Farrell for his helpful supervision, guidance, encouragement and friendship; Professor R. C. Jennison for the use of the facilities of the Digital Communications Laboratories, and to Professor M. J. Lanigan for providing me with the use of computer facilities.

I also would like to thank my brother, Majid, for his support, Mrs. Joan Elmes who kindly and efficiently typed this thesis, and all my colleagues in the Digital Communications Laboratories.

Finally, I wish to express most gratitude to my dear wife and son Ali for their love, patience and understanding.

ABSTRACT

The work reported in this thesis is an investigation in the field of error-control coding. This subject is concerned with increasing the reliability of digital data transmission through a noisy medium, by coding the transmitted data. In this respect, an extension and development of a method for finding optimum and near-optimum codes, using N.m digital arrays known as anticodes, is established and described.

The anticodes, which have opposite properties to their complementary related error-control codes, are disjoined from the original maximal-length code, known as the parent anticode, to leave good linear block codes. The mathematical analysis of the parent anticode and as a result the mathematical analysis of its related anticodes has given some useful insight into the construction of a large number of optimum and near-optimum anticodes resulting respectively in a large number of optimum and near-optimum codes. This work has been devoted to the construction of anticodes from unit basic (small dimension) anticodes by means of various systematic construction and refinement techniques, which simplifies the construction of the associated linear block codes over a wide range of parameters. An extensive list of these anticodes and codes is given in the thesis.

The work also has been extended to the construction of anti-codes in which the symbols have been chosen from the elements of the finite field GF(q), and, in particular, a large number of optimum and near-optimum codes over GF(3) have been found. This generalises the concept of anticodes into the subject of multilevel codes.

LIST OF CONTENTS

LIST OF CONTENTS

LIST OF CONTENTS (Continued)

LIST OF CONTENTS (Continued)

LIST OF CONTENTS (Continued)

LIST OF CONTENTS (Continued)

LIST OF CONTENTS (Continued)

SYMBOLS AND ABBREVIATIONS

$\underline{D}$	D-minus
$\equiv$	Has the same construction as
$\equiv$ mod-m	Congruent mod-m
θ_q	Subtraction mod-q
$\simeq$	Equivalent
$\sim$	Nearly equal
$\in$	Belongs to
	Is contained in
$\alpha\|+$	α-plus
$\alpha\|\ \alpha'$	α-plus-α prime
+	Plus,
	Concatenation
$\longrightarrow$	Results in,
	Mapping
$\xrightarrow{\alpha}$	Results in, with a mapping α
$(i)_q$	q-ary representation of the number i
δ	Maximum distance of an anticode
ψ	Iteration degree of an anticode
$[A]$	Matrix A
$[A]^T$	Transpose of matrix A
AC(m, k, δ, ψ)	Anticode with parameters m, k, δ, ψ
BAC(m, k, δ)	Basic anticode with parameters m, k, δ
B.C.H.	Bose-Chaudhuri-Hocquenghem
B.S.C.	Binary Symmetric Channel
$[C]$	Binary parent anticode-book
C	Number of check symbols in a block code-deletion code

$[C_o]$	Binary parent anticode including an all zero column
$[C_q]$	q-ary parent anticode
$[C]_{(i)}$	i^{th} anticode word
d	Hamming distance of a code-deletion code
D	The number of random errors a code always can detect
$D_{AC}(m_d+1, k_d, \delta_d, \psi_d)$	Domain anticode with parameters m_d+1, k_d δ_d, ψ_d
DC(n, k, d)	Deletion code with parameters n, k, d
$\mathscr{E}(x)$	Entropy, average information per symbol
$\mathscr{E}'(x)$	Entropy rate
$\mathscr{E}(x/y)$	Conditional entropy
$F_{AC}(m_r, k_r, \delta_r, \psi_r)$	First mapped anticode
$[G]$	Generator matrix of a binary block code (deletion code)
$[G_E]$	Generator matrix in standard echelon form
$[G]_{AC}$	Generator matrix of an anticode
$[G]_{ACT}$	Column permuted generator matrix of an anticode
$[G]_{ACS}$	Row permuted generator matrix of an anticode
$[G_q]_{AC}$	Generator matrix of q-ary anticode
$[G_q]_{PAC} = [M_q]$	Generator matrix of q-ary parent anticode
$[G]_{BAC}$	Generator matrix of basic anticode
$[G]_{CAC}$	Generator matrix of an anticode in row canonical form
$[G]_{D_{AC}}$	Generator matrix of domain anticode
$[G]_{DC}$	Generator matrix of deletion code
$[G]_{F_{AC}}$	Generator matrix of first mapped anticode
$[G]_{ML}$	Generator matrix of a maximal length code

$[G]_{S_{AC}}$	Generator matrix of second domain anticode
$[G]_{PAC}$	Generator matrix of parent anticode
$[G]_{UAC}$	Generator matrix of unit anticode
$g(x)$	Generator polynomial of a cyclic code
$GF(q)$	Galios field of q elements
G/H	Quotient group
$< g(x) >$	Ideal generated by the polynomial $g(x)$
$[H]$	Parity check matrix
$H(x)$	Entropy function
$h(x)$	Parity check polynomial of a cyclic code
$(II)_b^a$	Inverted-inversion stacking
$(II)_b^o$	Sequential-inversion stacking
$(II)_o^a$	Interleaved-inversion stacking
$(IM)_b^a$	Interleaved-map stacking
$(IM)_b^o$	Simple stacking
$(IM)_o^a$	Interleaved-inversion stacking
I_{AC}	Image anticode
$I_{AC}(m_i, k_i, \delta_i, \psi_i)$	Image anticode with parameters $m_i, k_i, \delta_i, \psi_i$
$I(x;y)$	Transinformation
L.C.M.	Least common multiple
K	Number of information digits in a block code
	Dimension of an anticode
$[M_o]$	Generator matrix of the parent anticode including an all zero column
$M^{(i)}(x)$	Minimum polynomial of the element x^i
$(MM)_b^a$	Mapped-map stacking
$(MM)_b^o$	Simple stacking

$(MM)_o^a$	Simple mapped stacking
$(MI)_b^a$	Mapped-inversion stacking
$(MI)_b^o$	Sequential-inversion stacking
$(MI)_1^o$	Simple-inversion stacking
$(MI)_o^a$	Simple-mapped stacking
$[N]_{AC}$	Modular representation of an anticode
$[N]_{DC}$	Modular representation of a deletion code
$[N]_{PAC}$	Modular representation of a parent anticode
p	Channel crossover probability
PAC(m, k, δ)	Parent anticode with parameter m, k, δ
p(x)	Primitive polynomial
q	Power of a prime number
R	ring
$\{s\}$	Set s
$S_{AC}(m_s, k_s, \delta_s, \psi_s)$	Second mapped anticode
$S_{DAC}(m_c, k_c, \delta_c, \psi_c)$	Second domain anticode
t	Number of random errors a code always can correct
$UAC(m_o, k_o, \delta_o, \psi_o)$	Unit anticode with parameters m_o, k_o, δ_o, ψ_o
V_k	k-dimensional vector space
$W(\overline{x})$	Weight of the code word $\overline{x}$
$W_{AC}(x)$	Anticode weight enumerator polynomial
$W_{AC}(s; \alpha, \beta)$	Weight attribute of an anticode
$W_{DC}(x)$	Deletion code weight enumerator
$W_{AC}^c(x,y)$	Combined weight enumerator of an anticode and related deletion code
$W_{BAC}^c(x,y)$	Combined weight enumerator of a basic anticode and related deletion code

$[W]_{AC}$	Weight distribution matrix of an anticode
$[W]_{DC}$	Weight distribution matrix of a deletion code
$[W]_{PAC}$	Weight distribution matrix of a parent anticode
$[\alpha(x)]$	Equivalence class of the polynomial $\alpha(x)$

CHAPTER I

INTRODUCTION

This thesis is concerned with the question of how to achieve reliable communication of information over a relatively less reliable communication medium or channel. All communication channels are in some sense unreliable; theory shows, however, that information may be transmitted over an unreliable channel with a reliability which can be, subject to certain constraints, as high as desired. One practical way of achieving high reliability is to use an error-correcting code; methods of synthesising and analysing optimum and good error-correcting codes are the subject of the research which is described in this thesis.

The process of communication may be seen in the form of the following generalised system. Information is generated at a source and transmitted through a medium called the channel to a receiver which is called the sink. In this process it is, in some measure, disturbed by an unwanted phenomenon referred to as noise. The loss or misinterpretation of some of the information due to the effect of this noise is one of the major problems to be faced when transmitting information over a channel, since all channels are, to a greater or lesser extent, noisy.

The average information per symbol at the output of the source (source entropy) is denoted by $\mathcal{E}(x)$ and the source entropy rate (Rosie, 1973), which is the rate of information flowing from the source into the channel is denoted by $\mathcal{E}'(x)$. If the average information per symbol at the sink due to both the information transmitted and the effect of noise (sink entropy) is similarly denoted by $\mathcal{E}(y)$ and the source entropy rate denoted by $\mathcal{E}'(y)$, then it follows that the noise entropy or average information at the sink, when it is known that any message has been transmitted, is given by:

$$\mathcal{E}(n) = \mathcal{E}(y/x).$$

Laying the foundation of information theory, Shannon (1948) considered these entropies and showed that, in any communication system, the average information (entropy) flowing from source to sink, denoted by I(x;y) and called the transformation, is:

$$I(x;y) = \mathcal{E}(y) - \mathcal{E}(n)$$

and similarly that the rate of receiving information is,

$$R = \mathcal{E}'(y) - \mathcal{E}'(n)$$

or alternatively R can be:

$$R = \mathcal{E}'(x) - \mathcal{E}'(x/y)$$

where $\mathcal{E}(x/y)$ is the average information at source when it is known that a message has been received.

The exchange of entropies can be shown pictorially (F. M. Reza, 1961) as in Figure (1-1):

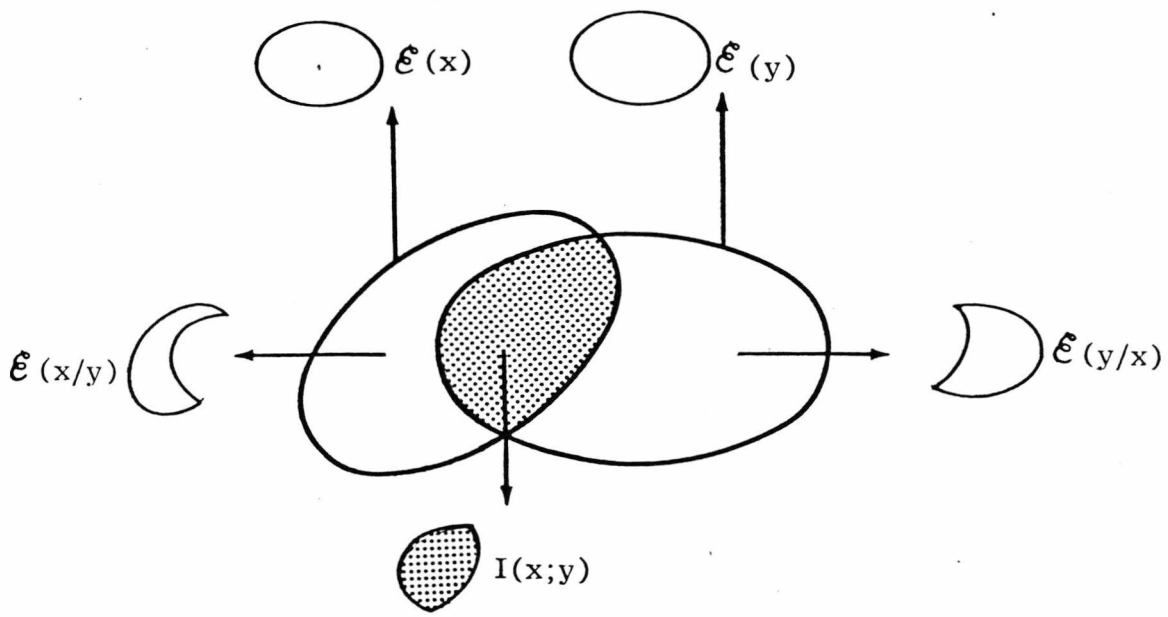


Figure (1-1) : The entropies exchange in a communication system.

Maximisation of R may be achieved by means of maximising, for instance, the source entropy $\varepsilon(x)$, (Fano 1949, Huffman 1952), or by matching the source and channel probabilities or probability distributions (Reza, 1961). This maximum value of R is called the channel capacity C, and it is the ultimate rate of transmission of information over a channel with defined characteristics.

Furthermore, Shannon showed that for a signal limited to a mean power S and a channel of WH_Z bandwidth with average noise power N, the maximum rate of information transmission is:

$$C = W\log_2(1 + \frac{S}{N}) \quad \text{bits/s.}$$

Shannon (1948), conceived the establishment of a 2WT dimensional space to show how a set of M equiprobable signals of duration T transmitted at a rate of 2WT samples (values) per second (Nyquist, 1924 and 1928) could be used as a source symbol set, and went on to show how with appropriate detection (decoding) of the vectors (minimum distance) that the maximum theoretical rate C is attainable with vanishingly small error rate if T, and therefore M, is large enough.

Therefore if, n-digit sequences of q symbols are used to transmit N different messages on a noisey channel, the N messages are mapped one-to-one into N different n-length sequences out of the q^n possible ones. Thus the images of the N messages form a subset of the q^n, n-digit sequences. If the mapping (encoding) is properly done, it provides the possibility of correctly reselecting (decoding) the original n-digit sequences, which have been transmitted and corrupted by noise, and hence of determining the related message (information digits). If the number N of messages is close to q^n, then difficulties arise in the process of the message reconstruction, which results in either a wrong message interpretation (decoding error), or in a very

complex decoding process. Therefore the number of messages N is normally rather smaller than q^n; that is, in practice the redundancy cannot usually be as small as that predicted by Shannon. The scheme for mapping the N messages into n-digit sequences (or alternatively, adding controlled redundancy) is called "coding", and the set of the images of the N messages is known as a "code". Shannon states that for any channel with the maximum rate of information transmission C there are codes with $R < C$ and length n which, with maximum likelihood decoding, have a vanishingly small probability of erroneous decoding (Van Lint 1973, Galager 1968) P_e, such that

$$P_e \leq e^{-nE(R)}$$

E(R) is a function of crossover probability of the channel.

The inequality reveals that the probability of erroneous decoding decreases rapidly with increasing n. The complexity of the decoder increases rapidly with n, however, which is a big disadvantage for practical systems. So, a coding system can be considered to be a good code if it is both long and also can be encoded and decoded easily and efficiently.

From the above considerations a general simplified communication can be shown schematically as in Figure (1-2).

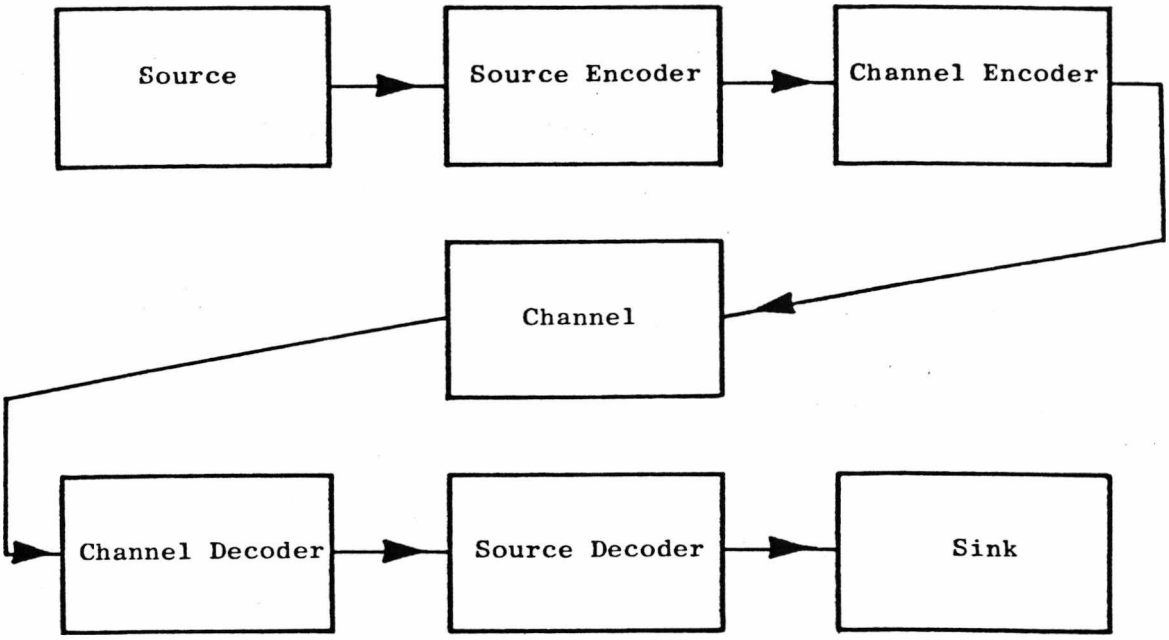


Figure (1-2) : A general simplified communication system.

The encoder is a system which improves the efficiency of the source and/or its matching with the channel in both the information flow (source encoding) and error correction (channel encoder) sense. These encoding operations are usually performed separately, but in certain cases may be combined with advantage (Ancheta, 1976). The decoder transforms the received noisy signal, by correcting (after detecting) errors and restoring information compressed or modified by the source encoder into a form acceptable to the sink.

The attempts so far which have been made to find a system (as in Figure (1-2)) which achieves in practice a capacity equal to Shannon's theoretical channel capacity, have led to the extensively investigated branch of digital communications known as coding theory. In spite of all these attempts, the problem of constructing codes which attain a low probability of error at a rate close to the now thirty years old Shannon capacity is still in general unsolved, particularly in the cases of very noisy, or time-varying, channels.

In 1952, Gilbert found a lower bound on the ratio of the minimum distance to the block length of the best code with a certain rate. However, all the best known classes of best constructive long codes are on or below this bound, with one exception, the Justesen codes (Justesen, 1972), which are rather impractical to implement. Thus the goal of coding research has been constructing codes which for any arbitrary rate at least meet the Gilbert bound. One of the most successful of all classes of codes came from the work of Elias (1954), who introduced the concept of iteration by combining Hamming (1950) codes. In 1966, Forney extended Elias' idea, introducing concatenated codes; and in 1972, Justesen developed the work done by Forney further to produce the codes mentioned above. These codes all have small output (decoded) probability of error for any arbitrary rate, or in other words for any code rate the ratio of minimum distance to block length is over-bounded away from zero.

However, many good, moderate length codes such as Reed-Muller codes (1954), Bose-Chaudhuri-Horquenghem codes (1959-1960), quadratic residue codes (Berlekamp 1968, Peterson & Weldon 1972) have also been discovered. Asymptotically, though, these codes fall below the Gilbert bound.

Thus several techniques and design methods for code construction have been originated and developed, and have resulted in a number of useful codes, which are considered to have good parameters, and efficient practical decoding techniques. Following Shannon's footsteps, a large number of researchers have produced an enormous amount of reports, papers and text-books which describe the development of the science of coding theory. A comprehensive coverage of algebraic coding theory is given by Berlekamp (1968), Sloane & MacWilliams (1978) - this book provides also an exhaustive bibliography of references in coding theory,

Peterson & Weldon (1972), Van Lint (1971), Wiggert (1978), Massey (1963), Forney (1966) and Gallager (1968).

The general subject of digital communications is covered by Lucky, Salz & Weldon (1968), Stiffler (1971), Wozencraft and Jacobs (1965), Rosie (1973) and others; all these books contain sections on coding theory.

From this brief review, it becomes apparent that coding theory is concerned with methods of choosing long codes with practical encoding methods and efficient decoding algorithms, though not necessarily fulfilling Shannon's promised rate and output error probability (see Section 3.1.3).

The main aim of coding theory is to try to approach Shannon's limit on capacity and error-free decoding; the practical aim of research in coding theory is to find codes with as high rate as possible for a given length and error-correcting power. These codes are called optimum codes, (see also Section 3.1.3). This can be achieved by fixing two code parameters, e.g. the block length and the rate, and attempting to maximise the third parameters, namely the error-correcting power. The ultimate values of these parameters have been investigated and given by various sources in the form of functional relationships, known as code bounds (see Section 4.1 and 4.2). In finding codes with specified parameters, our main concern in this thesis is to develop and extend one of the most fruitful techniques of constructing codes which have a set of good parameters; such codes are called near optimum or optimum codes. These techniques were first introduced by Farrell (1969) and are concerned with the concept of the anti-code. An anti-code is a set of N m-digit q-ary sequences, or words, which has opposite properties to an error-correcting code. Many codes may be derived from a given anti-code; the columns of code and anti-code books play an important part in the theory of these techniques.

In this thesis, we assume that even if the source symbols are affected by intersymbol influence, then the symbols delivered to the channel encoder by the source encoder are equiprobable and independent.

The second chapter in this thesis presents some of the elementary and simplified mathematical background related to error control coding schemes. In Chapter III, the achievement of reliable transmission of information over an unreliable channel is reviewed, and some related concepts are discussed. Chapter IV contains the analysis and construction techniques of some of the best known linear block codes and gives a brief outline of the functional relationship between the parameters of these codes. Through these functions (known as bounds) the performance of the constructed codes can be verfied. In Chapter V, the area of interest is narrowed to a detailed consideration of a family of codes related to the development of the main subject of this thesis. These codes, known as maximal length codes, are again refined to include the codes derived from them, as suggested by Griesmer (1960) and Solomon & Stiffler (1965). Furthermore, a new approach is given to the Griesmer bound and the generalization of this bound by Solomon and Stiffler.

Following Chapter V, in Chapter VI the analysis and construction of an infinite set of arrays complementary to error correcting codes, known as anticodes, is described. Consideration is confined to the class of linear binary anticodes. In Chapter VII, further properties of linear binary anticodes, and some related bounds, are given. Moreover, in this chapter, the linear binary construction results are presented. In Chapter VIII, the development, construction, and some properties of linear anticodes over GF(q), which is the generalization of the linear binary anticode, are given. However, the emphasis has been given to 3-ary linear anticodes. This chapter also includes the 3-ary linear binary construction results, and corresponding linear 3-ary codes. The

final Chapter, IX, is devoted to the further discussion of the new anticode construction methods, and possible use of related properties, and gives several recommendations and possible topics for further research based upon this work.

In addition, Appendix I gives a possible decoder and encoder for the codes derived from parent anticodes. Also, in Appendix II, the approximate performance of the linear binary codes derived from parent anticodes, with the corresponding computer program, are given. Appendix III includes the computer program for the refinement of the constructed anticodes.

CHAPTER II

MATHEMATICAL BACKGROUND AND CONCEPTS

2.1 General

The concepts to be presented in this section are some of the elementary and simplified properties of the algebraic systems and mathematical references which are related to error control coding schemes. The aim of this is to give some insight into the problems involving error control coding, through a knowledge of the analogies between error control codes and mathematical systems. The topics of this chapter can be found described in greater depth in text books such as Birkhoff & Burtee (1970), Fraleigh (1977), Burton (1967), Barnes (1963) and others.

2.2 Sets

A set D is a collection of objects or entities called the elements of the set. An empty set or null set is a set which has no elements, and it is symbolized by ϕ. A set T, each of whose elements are in the set D, is a subset of set D. This is shown by $T \subseteq D$ and read T is contained in D.

If $T \subseteq D$ and $T \neq D$ and $T \neq \phi$ then T is called a proper subset of the set D; however, if $T = D$ or $T = \phi$ then T is an improper subset of the set D. To indicate that T is a proper subset of D we will use the notation $T \subset D$.

2.3 Mapping

A correspondence α which associates each element of a set D with a unique element of a set I is called a mapping of D into I. This mapping of the set D into I is a subset of $D \times I$ such that no two distinct pairs has the same first component. The notation

$$\alpha: \ d \longrightarrow i$$

is used to indicate the correspondence of the element $i \in I$ with the element $d \in D$ due to mapping α. This may also be shown by $d\alpha = i$. The element i is called the image of the element d under α. The set D is called the domain and the set I is called the co-domain of the mapping α. The mapping α is symbolically expressed by writing

$$\alpha: \ D \longrightarrow I$$

or

$$D\alpha = \left\{ i = d\alpha \ \middle| \ i \in I, \ d \in D \right\}$$

If every element of the set I is the image of at least one element of D the correspondence is a mapping α of D onto I and the set I is called the range of α. If distinct elements of D are associated with distinct elements of I the mapping α is called a one-to-one mapping into. A one-to-one mapping α of D onto I is called a one-to-one correspondence between D and I, and it is symbolically expressed by writing

$$\alpha: \ D \longleftrightarrow I.$$

Two mappings α and β,

$$\alpha: \ D \longrightarrow I \quad \text{and} \quad \beta: \ I \longrightarrow R$$

can be combined to form a mapping $\alpha\beta$ called the product mapping α and β where

$$\alpha\beta : \ D \longrightarrow R.$$

Thus if $d \in D$ and $d\alpha = i$ the mapping $\alpha\beta$ associates the element d with the element $r \in R$ where $r = i\beta$. In other words we have that,

$$d(\alpha\beta) = d\alpha(\beta) = i\beta = r \ .$$

The product mapping $\alpha\beta$ is called identity mapping if the mapping $\alpha\beta$

associates every element of D with itself, that is

$$\alpha : D \longrightarrow I \quad \text{and} \quad \beta : I \longrightarrow D$$

and the product mapping

$$\alpha\beta = J : D \longrightarrow D$$

and also

$$J : d \longrightarrow d \quad \text{for all } d \in D.$$

If the mapping α is a one-to-one mapping of D onto I where

$$\alpha\beta = J : D \longrightarrow D \quad ,$$

then α is said to be the left inverse of β, and similarly β is said to be the right inverse of α, so that

$$\alpha = \beta^{-1} \quad \text{or} \quad \beta = \alpha^{-1}$$

2.4 Binary operation

A correspondence that associates each ordered pair (a,b) of the set $D \times D$ with a uniquely determined element of the set D is called a binary operation $*$. In other words, a binary operation $*$ is a mapping of D into D, as

$$* : D \times D \longrightarrow D$$

and for the sake of simplicity $(a,b)*$ is written as $a * b$. A binary operation $*$ on a set D is called commutative whenever

$$a * b = b * a$$

and it is called associative whenever

$$a * (b * c) = (a * b) * c$$

for all $a, b, c \in D$.

2.5 Residue Classes

The relation "congruent modulo m" which is symbolized by $\equiv \pmod m$ is defined on the elements of the set of all integers I by $a \equiv b \pmod m$ if and only if (a-b) is divisible by m, in other words a and b have the same remainder when divided by the positive integer m.

This relation is an equivalence relation, so it partitions the set of all integers I into m equivalence classes I/m, known as residue classes mod-m, such that, if [r] is an equivalence class, then

$$[r] = \left\{ a \mid a \in I, \quad a \equiv r \pmod m \right\} \quad .$$

As an example consider the residue classes of the integers mod 3,

$$[0] = \left\{ \ldots, -6, -3, 0, 3, 6, \ldots \right\}$$

$$[1] = \left\{ \ldots, -5, -2, 1, 4, 7, \ldots \right\}$$

$$[2] = \left\{ \ldots, -4, -1, 2, 5, 8, \ldots \right\}$$

In the next section we will see that the set of all residue classes mod-m of all integers,

$$I/m = \left\{ [0], [1], \ldots, [m-1] \right\}$$

form an algebraic system with respect to the binary operations + and . defined as follows:

$$[a] + [b] = [a + b] \quad \text{and} \quad [a] \,.\, [b] = [a \,.\, b]$$

which is known as a ring.

As an example consider the set of all residue classes of integers mod-3 (previous example); the operation tables are:

+	[0]	[1]	[2]
[0]	[0]	[1]	[2]
[1]	[1]	[2]	[0]
[2]	[2]	[0]	[1]

	[0]	[1]	[2]
[0]	[0]	[0]	[0]
[1]	[0]	[1]	[2]
[2]	[0]	[2]	[1]

2.6 Groups

An algebraic system is a set $M = \{a, b, \ldots\}$ with a number of operations and relations defined on it. A group G is an algebraic system with one binary operation * , provided this operation satisfies the following requirements:

(1) the associative law holds, namely

$$a * (b * c) = (a * b) * c;$$

(2) there exists an element $U \in G$, called the identity such that

$$a * U = U * a = a;$$

(3) for every $a \in G$ there exists an element $a^{-1} \in G$ called the inverse of a, such that,

$$a * a^{-1} = a^{-1} * a = U.$$

If the group also satisfies:

(4) the commutative law,

$$a * b = b * a \quad \text{for all } a, b \in G,$$

then the group G is called an Abelian group.

As an example, consider the set G of order n = 4:

$$G = \{a, b, c, d\} \quad .$$

This set forms a group with respect to the binary operation * defined by Table (2:1).

*	a	b	c	d
a	a	b	c	d
b	b	a	d	c
c	c	d	a	b
d	d	c	b	a

Table (2:1)

From Table (2:1) it is clear that G forms an Abelian group with respect to *. The identity element is "a", the inverse of every element is itself, and the associative law holds. In addition, due to the commutative property of the elements, the group G is an Abelian group.

2.6.1 *Properties of Groups*

Consider the set D of order N = 4,

$$D = \{0\,0\,0,\ 0\,1\,1,\ 1\,0\,1,\ 1\,1\,0\}$$

and the binary operation + which is a mod-2 component by component (component-wise) addition over the components of the elements of D, shown by Table (2:2):

*	0 0 0	0 1 1	1 0 1	1 1 0
0 0 0	0 0 0	0 1 1	1 0 1	1 1 0
0 1 1	0 1 1	0 0 0	1 1 0	1 0 1
1 0 1	1 0 1	1 1 0	0 0 0	0 1 1
1 1 0	1 1 0	1 0 1	0 1 1	0 0 0

Table (2:2)

Further consideration reveals that the set D with respect to the operation defined satisfies the properties of an Abelian group.

Now we establish a mapping α between set G and D such that

$$\alpha : G \longleftrightarrow D$$

$$a \longleftrightarrow 0\ 0\ 0$$

$$b \longleftrightarrow 0\ 1\ 1$$

$$c \longleftrightarrow 1\ 0\ 1$$

$$d \longleftrightarrow 1\ 1\ 0$$

It is readily verified that the mapping α is a one-to-one correspondence between the sets G and D, which preserves the operations of the groups G and D, namely if

$$g_1 \in G \longleftrightarrow d_1 \in D$$

$$g_2 \in G \longleftrightarrow d_2 \in D$$

then

$$g_1 * g_2 \longleftrightarrow d_1 + d_2$$

so we have that the set D (which is a linear binary block code) is a group (group code) with respect to binary operation mod-2 addition.

Every one-to-one correspondence of a group G onto a group D which preserves the operations and relations (if they exist) is called an isomorphism. The groups G and D are said to be isomorphic. If the mapping above is not one to one then it is called a homomorphism.

2.6.2. *Subgroups*

A non-empty subset H of a group G is called a subgroup of G if H is a group with respect to the operation defined on G. The group G and the set $\{U\}$, where U is the identity element of G, are subgroups of G, and are called the trivial or improper subgroups of G; other sub-groups are proper subgroups of G.

2.6.3 *Cyclic Group*

A group G is a cyclic group if for every element $b \in G$ the element b is in the form of

$$b = a^m$$

where $a \in G$ and m can be any integer. The element a is called the generator of G. The set G consisting of the six, sixth roots of unity,

$$G = \left\{ p,\ p^2,\ p^3,\ p^4,\ p^5,\ p^6 = 1 \right\}$$

form a multiplicative group, where p is the generator and $p^6 = 1$ is the identity element.

2.6.4 *Permutation (Permutation Group)*

Consider the set consisting of the n! permutations of the n symbols of the set $s = \left\{ a_1,\ a_2,\ \ldots\ a_n \right\}$. The set of all permutations of the elements of S in fact is n! different one-to-one mapping of the set S on-to itself. Consider the following n-tuple:

$$(a_{i_1},\ a_{i_2},\ a_{i_3}\ \ldots\ a_{i_n})$$

where i_k for K = 1, 2, 3, ..., n are 1, 2, 3, ..., n and moreover $i_k \neq im$ for $K \neq m$. This n-tuple or arrangement of elements of the set S determines the following one-to-one correspondence:

$$\alpha_i : \begin{pmatrix} a_1 & a_2 & a_3 & \ldots & a_n \\ \downarrow & \downarrow & \downarrow & & \downarrow \\ a_{i_3} & a_{i_2} & a_{i_n} & \ldots & a_{i_5} \end{pmatrix}$$

of the set S onto itself. This α_i is the i^{th} permutation of the set S where i = 1, 2, ... n! . We will usually denote α_i by the n-tuple determining it, i.e. $(a_{i_1}\ \ a_{i_2}\ \ a_{i_3}\ \ \ldots\ a_{i_n})$.

Consider the set,

$$S_n = \left\{ \alpha_1,\ \alpha_2,\ \ldots,\ \alpha_i,\ \ldots\ \alpha_{n!} \right\}$$

consisting of all n! permutations of the set S. Let * be the product of the one-to-one mapping of S onto S, then S_n with this binary operation forms a group (non-Abelian) of order n!. The identity element of the group, is α_1 where,

$$\alpha_1 : \begin{pmatrix} a_1 & a_2 & a_3 & \cdots & a_n \\ & & & & \\ a_1 & a_2 & a_3 & \cdots & a_n \end{pmatrix}$$

and the inverse of every element of S_n such as α_i is an α_j where the rows are interchanged.

As an example, let the set

$$S = \{1, 2, 3, 4\}$$

and let α and β be

$$\alpha = \begin{pmatrix} 1 & 2 & 3 & 4 \\ & & & \\ 2 & 3 & 4 & 1 \end{pmatrix} \quad \text{and} \quad \beta = \begin{pmatrix} 1 & 2 & 3 & 4 \\ & & & \\ 3 & 4 & 1 & 2 \end{pmatrix}$$

since interchange of the columns of β is immaterial then β can be written as:

$$\beta = \begin{pmatrix} 2 & 3 & 4 & 1 \\ & & & \\ 4 & 1 & 2 & 3 \end{pmatrix}$$

In this form the first row of β is the second row of α, therefore

$$\gamma = \alpha * \beta = \begin{pmatrix} 1 & 2 & 3 & 4 \\ & & & \\ 2 & 3 & 4 & 1 \end{pmatrix} * \begin{pmatrix} 2 & 3 & 4 & 1 \\ & & & \\ 4 & 1 & 2 & 3 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ & & & \\ 4 & 1 & 2 & 3 \end{pmatrix}$$

the identity element is

$$U = \begin{pmatrix} 1 & 2 & 3 & 4 \\ & & & \\ 1 & 2 & 3 & 4 \end{pmatrix}$$

and the inverses α and β are:

$$\alpha^{-1} = \begin{pmatrix} 2 & 3 & 4 & 1 \\ & & & \\ 1 & 2 & 3 & 4 \end{pmatrix} \quad \text{and} \quad \beta^{-1} = \begin{pmatrix} 4 & 1 & 2 & 3 \\ & & & \\ 1 & 2 & 3 & 4 \end{pmatrix} .$$

If a permutation changes only two elements of the set, the permutation is called a transposition and it can be shown (Ayres, 1965) that any permutation is a product of transpositions. Also, every group of order* n is isomorphic to a permutation group of n symbols (MacCoy 1977, Ayres 1965).

2.6.5 *Cosets (Invariant Subgroups)*

A subgroup H of a group G can partition the elements of the group G (parent group) into mutually disjoint subsets, known as cosets. In the subject of error correcting codes the set of cosets is called the standard array.

Consider the subgroup H of the finite group G with binary operation * , for which the subgroup H and an element $a \in G$ form a subset Ha known as a right coset, where

$$Ha = \left\{ h * a \;\middle|\; h \in H \right\} .$$

Similarly the left coset aH is the set

$$aH = \left\{ a * h \;\middle|\; h \in H \right\} .$$

The element a is called the coset leader or coset representative of aH. It can be shown that the order of each subgroup of a finite group divides the order of the parent group†, and furthermore if aH = Ha, then H is called a normal or invariant subgroup of the parent group (Burton 1967, Barens 1963).

---

* **The order of a group is the number of elements in the group.**

† **This is Lagrange's theorem (Birkhoff & Burtee, 1970), the converse of which is not true; namely, a group of order n need not have a sub-group of order K when K divides n.**

2.6.6 *Quotient Group*

The set of all distinct cosets of a finite Abelian group H with respect to the operation " o " defined on the cosets as

$$(Ha) \circ (Hb) = \left\{ (h_1 a) * (h_1 b)/h_1,\ h_2 \in H \right\}$$

or $$(Ha) \circ (Hb) = H(a * b)$$

form a group called a quotient group which is symbolized by G/H and is given by

$$G/H = \left\{ Ha,\ Hb,\ \ldots\ldots \right\}$$

2.6.7 *Standard Array*

The concept of group decomposition into cosets is useful in error correcting codes when studying the decoding and structure of linear codes (Slepian 1956, Peterson and Weldon 1972).

Consider the set V of all the binary n-tuple vectors; this set forms an Abelian group of order 2^n with respect to binary (mod-2) addition operation + . The n-tuple consisting of only zeros is the identity element of V. If v is a subgroup of order 2^k of the group V, then the number of distinct cosets is $2^n/2^k = 2^{n-k}$.

In order to form the standard array of V w.r.t. v, the elements of v are placed in a row with the identity element (u), in the leftmost position of this row. The second row is started with an element e_1 of V which has not appeared in the first row. This element, in error correcting terms, is usually chosen to be one of the most likely elements of V to be received if the identity element is transmitted and errors occur. In other words, this n-tuple is a low weight* error pattern e_1 (see Peterson and Weldon 1972, Slepian 1956) and the rest of the second row is formed by adding this element e_1 to each element of the first row.

---

* The weight of an n-tuple is the number of non-zero digits it contains; e.g. w(1101) = 3.

The third and consecutive rows are formed in the same way as the second row, see Table (2:3).

As an example, consider the standard array formed by decomposition of the 6-tuple binary vectors into 8 cosets, as in Table (2:4). Since $n = 6$ then $|V| = 2^6$ and $|v| = 2^3$, where $|V|$ is the order of the group V.

(u)	(v_1)	(v_2)	....	(v_{2^k-1})
(e_1)	$(v_1 + e_1)$	$(v_2 + e_1)$	....	$(v_{2^k-1} + e_1)$
(e_2)	$(v_1 + e_2)$	$(v_2 + e_2)$	....	$(v_{2^k-1} + e_2)$
- - - - -	- - - - -	- - - - -	- - - - -	- - - - -
$(e_{2^{n-k}-1})$	$(v_1 + e_{2^{n-k}-1})$	$(v_2 + e_{2^{n-k}-1})$	....	$(v_{2^k-1} + e_{2^{n-k}-1})$

Table (2:3)

000000	001011	010101	011110	100110	101101	110011	111000
000001	001010	010100	011111	100111	101100	110010	111001
000010	001001	010111	011100	100100	101111	111001	111010
000100	001111	010001	011010	100010	101001	110111	111100
001000	000101	011101	010110	101110	100101	111011	110000
010000	011011	000101	001110	110110	111101	100011	101000
100000	101011	110101	111110	000110	001101	010011	011000
001100	000111	011001	010010	101010	100001	111111	110100

Table (2:4)

2.7 Rings

A ring R is an algebraic system with two binary operations, namely addition "+" and multiplication "∘", provided these operations satisfy the following requirements:

(1) R is an Abelian group with respect to the addition operation;

(2) the associative law of multiplication holds, namely $a \circ (b \circ c) = (a \circ b) \circ c$ for all $a, b, c \in R$;

(3) the distributive law (relation between two operations) holds, namely

$$a \circ (b + c) = a \circ b + a \circ c$$

$$(b + c) \circ a = b \circ a + c \circ a .$$

If, in addition, the ring R satisfies

(4) the commutative law of multiplication, namely

$$a \circ b = b \circ a$$

for all $a, b \in R$,

the ring is called a commutative ring.

As an example, consider the group G of Section 2.6 with the binary operation " + " of Table (2:5) where

$$G = \{a, b, c, d\} .$$

This set forms a ring R with respect to the binary operation addition + and multiplication $\circ$ defined by Table (2:6).

+	a	b	c	d
a	a	b	c	d
b	b	a	d	c
c	c	d	a	b
d	d	c	b	a

Table (2:5)

$\circ$	a	b	c	d
a	a	a	a	a
b	a	b	c	d
c	a	c	d	b
d	a	d	b	c

Table (2:6)

More consideration reveals that the existence of the symmetric properties on both tables with respect to the main diagonals from upper left to lower right, means the ring R is a commutative ring and therefore for every two elements $x, y \in R$

$$x \circ y = y \circ x .$$

Furthermore, from the second row of the table of the multiplication, it is clear that b is the multiplicative identity, called the unity element of the ring. This ring is called "a ring with unity". In addition, for every two non-zero elements $x, y \in R$, the multiplication table shows $x \circ y \neq 0$. Such a ring is said to be a ring having no divisor of zero. A commutative ring with unity which has no divisor of zero is called an integral domain.

2.7.1 *Subrings*

A subset S of a ring R which is itself a ring is called a subring. It is clear that S is a subgroup of the additive group R. The subring $\{Z\}$, where Z is the zero element of the ring, and R, are the improper subrings of R; the others are the proper subrings of R.

2.7.2 *Ideals*

An important class of subrings used in error-correcting codes, which are similar to those of the invariant subgroups, are known as ideals. A subring I of a ring R is called a left ideal of R if for every element $a \in R$ and $b \in I$, $a \circ b \in I$; and it is called a right ideal whenever $b \circ a \in I$ for every $a \in R$ and every $b \in I$. If the ideal is both a left and a right ideal, it is called a two-sided ideal or simply an ideal. In addition to those properties, if the elements of an ideal can be generated by some fixed elements of the ideal, it is called a principal ideal.

An ideal I of a ring R is an invariant subgroup of the additive

group of ring R, so this subgroup can partition R into a set R/I of distinct cosets, such that

$$R/I = \{a + I : a \in R\} .$$

The elements of this set form a ring with respect to the following well defined operation on the cosets,

$$(a + I) + (b + I) = (a + b)I$$

$$(a + I) \circ (b + I) = (a \circ b) + I .$$

The ring of the elements of R/I is called the quotient ring, or, in other words, the residue classes modulo the ideal (I).

2.7.3 *Division Ring (Skew Field-Quasi Field), Field*

If the set of the non-zero elements of a ring form a multiplicative group with respect to the binary multiplication operation, it is called a division ring or skew field. A ring R is a field if the set of non-zero elements in the R form an Abelian multiplicative group. A subset of a field F which is itself a field is called a subfield. The subfields $\{Z\} = \{0\}$ and F are improper and the others are proper subfields of the field F. A field with a finite number of elements q, where q is a prime integer, or a power of a prime integer, is called a Galois field of q elements, where $q = p^m$ with p a prime integer and m any non-zero positive integer. The Galois field of q elements is denoted as

$$GF(q) = GF(p^m) .$$

2.8 Vector Spaces

A vector space V(F) over a field F (called the set of components) is a set of elements called vectors, such that these elements form an Abelian group with respect to the binary addition operation defined on these elements and satisfies the requirements:

(1) scalar multiplication:

for any pair $a, b \in F$ and $v \in V(F)$ the product

$a\ v \in V(F)$ is defined for all $a \in F$ and $v \in V(F)$;

(2) distributive law:

$a(u + v) = au + av$

also $(a + b)v = av + bv$

for all $a, b \in F$ and $u, v \in V(F)$;

(3) associative law:

$(ab)v = a(bV)$

for all $a, b \in F$ and $V \in V(F)$.

In addition, if I is the unity element of the field F then

$$Iv = v \ .$$

As an example consider the set of all different ordered n-tuples of elements of the field F, with the addition of two n-tuples (V) and (U), defined as below.

If

$$(V) = (v_1, v_2, v_3, \ldots, v_{n-1}, v_n)$$

$$(U) = (u_1, u_2, u_3, \ldots u_{n-1}, u_n)$$

where v_i and $u_i \in F$ for all $i = 1, \ldots n$, then

$$(V) + (U) = (v_1 + u_1, v_2 + u_2, \ldots, v_n + u_n)$$

bearing in mind that whenever the field F is GF(p) the addition of components is carried out (mod-p). Also if c is any element of F then,

$$c(V) = (cv_1, cv_2, cv_3, \ldots cv_n)$$

is the scalar product of $c \in F$ and $(v) \in V(F)$. Thus the set of all n-tuples over a field F forms a vector space. The inner product of two vectors $(V) = (v_1, v_2, \ldots, v_n)$ and $(U) = (u_1, u_2, \ldots, u_n)$ is defined by,

$$(V).(U) = v_1u_1 + v_2u_2 + \ldots + v_nu_n.$$

Two vectors V and U are said to be orthogonal if the inner product of u and v is equal to zero.

2.8.1 *Subspace of a Vector Space*

A non-empty subset of the elements of a vector space V over a field F is called a subspace, provided that the subset is itself a vector space over the field F. A set of vectors $\{v_1, v_2, v_3, \ldots v_k\}$ is linearly dependent if and only if there are scalars $c_1, c_2, \ldots c_k$, not all zero, such that

$$\sum_{i=1}^{k} c_i V_i = 0$$

A set of vectors is linearly independent if it is not linearly dependent; furthermore if any vector in a vector space is a linear combination of the element of a set of k linearly independent vectors, the vector space is the row space of the k linearly independent vectors; or in other words the k vectors span the vector space and are called a basis of the vector space, and the vector space is called a k-dimensional vector space.

2.9 Matrices

A matrix of order m × n over a field F is a rectangular array of m rows and n columns with as elements the ordered elements of the field F. The transpose of any matrix [A] of order m × n is a matrix [B] of order n × m such that the matrix [B] can be obtained by interchanging rows and columns of the matrix [A], and it is shown as

$$\left[B\right] = \left[A^T\right]$$

The set of all linear combination of the rows of a k × n matrix over a field F is the row space of this matrix. If k rows are linearly

independent the row space is a k-dimensional subspace of the n-dimensional vector space, and it is said that the row rank of the matrix is k. Similarly, the set of all linear combinations of the columns of a matrix is the column space, and the dimension and column rank have the same definition. The process of interchanging of any row, or multiplying any row by an element of the field F, or addition of any multiple of one row of a matrix over a field F, to another row, is called the elementary row transformation (operation). The similarly defined operation on the columns of a matrix is called the elementary column transformation. Two matrices are row or column equivalent if one can be obtained from another respectively by a sequence of row or column linear transformations. Two matrices are equivalent if one can be obtained from another by a sequence of row and column transformations.

Using elementary row operations, a matrix can be put in the simple canonical form called standard echelon form which has the following properties:

(1) any non-zero row has a ONE as the leading (leftmost) term;

(2) any column containing such a leading term has all other elements equal to zero;

(3) the leading term of any row is to the right of the leading term in every preceding row, and all zero element rows are below the all non-zero rows.

The dimension of the row space of the rows of a matrix A in standard echelon form is the same as the number of the non-zero rows of the matrix $[A]$.

A square matrix over a field F which has the unit element of the field F in the main diagonal from top left to the right bottom, and zeros elsewhere is called the identity matrix $[I]$. A square matrix is said to be non-singular if it is row equivalent to the identity matrix $[I]$.

Consider the identity matrix $[I]$ and a linear transformation being carried out on its rows. Multiplying this matrix on the right with the matrix $[A]$ is the same as performing the linear transformation on the rows of the matrix $[A]$. This transformation matrix is called an elementary row transformation matrix $[S]$, and is a non-singular matrix. A similar definition applies for an elementary column transformation (permutation) matrix $[T]$ (Peterson & Fontaine, 1959). The matrix $[T]$ operates on the right of the matrix $[A]$. It can be shown (Ayres 1965), that if two matrices $[A]$ and $[B]$ are equivalent, then there are non-singular matrices (elementary row matrix $[S]$, and elementary column matrix $[T]$) such that

$$[S] \cdot [A] \cdot [T] = [B]$$

2.10 Galois Field, Polynomial Residue Classes

The ring of residue classes mod-P of integers of Section 2.5 form a finite field of order P if and only if P is a prime number. This field is called the Galois field and is denoted by GF(P) (see Section 2.7.3).

Consider $F[x]$, the set of all polynomials over the field GF(P) of the residue classes of integers mod-P. $F[x]$ forms a ring with respect to the addition and multiplication defined below.

If $\quad \alpha(x) = a_o + a_1 x + a_2 x^2 + \ldots\ldots$

and $\quad \beta(x) = b_o + b_1 x + b_2 x^2 + \ldots\ldots$

where $a_i, b_i \in GF(P)$

and $\alpha(x), \beta(x) \in F[x]$

Then $\quad \alpha(x) + \beta(x) = \sum_i (a_i + b_i) x^i$

$$\alpha(x) \cdot \beta(x) = \sum_i \left(\sum_{j=o}^{i} a_i + b_{i-j} \right) x^i$$

(the addition of $a_i, b_i \in GF(p)$ is performed mod-P).

The ring of polynomials $F[x]$ can be partitioned by any polynomial $P(x)$ of degree m into a commutative ring with unity, called the residue class of polynomials modulo, the polynomial $P(x)$, denoted by $F[x]/P(x)$ such that

$$F[x]/P(x) = \left\{ \left[a_o + a_1 x + ,,,, + a_{m-1} x^{m-1} \right] \;\middle|\; a_i \in GF(P) \right\}$$

where

$$\left[a_o + a_1 x + ... + a_{m-1} x^{m-1} \right] = \left[\alpha(x) \right] = \left\{ \alpha(x) + P(x)\, \gamma(x) \;\middle|\; \gamma(x) \in F[x] \right\} .$$

The addition and multiplication on the residue classes of polynomials is defined as

$$[\alpha(x)] + [\beta(x)] = [\alpha(x) + \beta(x)]$$

$$[\alpha(x)] \;.\; [\beta(x)] = [\alpha(x) \;.\; \beta(x)]$$

The addition operation over the coefficient a_i and b_i is carried out mod-P, and the multiplication is carried out mod-$P(x)$. Furthermore, scalar multiplication of each element a_i of GF(P) and elements $[\alpha(x)]$ of $F[x]/P(x)$ is as :

$$a_i [\alpha(x)] = [a_i\, \alpha(x)]$$

The distribution property of multiplication is as:

$$[\alpha(x)] \left([\beta(x)] + [\gamma(x)] \right) = [\alpha(x)]\; [\beta(x)] + [\alpha(x)]\; [\beta(x)] \;.$$

The ring of polynomial residue classes mod-P forms an m-dimensional vector space of GF(P), such that the basis residue classes are

$$[1] \;,\; [x] \;,\; [x^2] \;,\; ..... \;,\; [x^{m-2}] \;,\; [x^{m-1}]$$

and any vector or residue class of $F[x]/P(x)$ can be formed by linear combination of the basis residue classes, that is:

$$a_o[1] + a_1[x] + a_2[x^2] + ... + a_{m-1}[x^{m-1}] =$$

$$[a_o] + [a_1 x] + [a_2 x^2] + \ldots + [a_{m-1} x^{m-1}] = [a_o + a_1 x + a_2 x^2 + \ldots + a_{m-1} x^{m-1}]$$

where $a_i \in GF(P)$.

If any of the residue classes $[a_i]$ or $[x^i]$ is regarded as

$$[a_i] = a_i \quad \text{and} \quad [x^i] = x^i$$

then the above equation can be written as a polynomial

$$a_o + a_1 x + a_2 x^2 + \ldots + a_{m-1} x^{m-1} \quad .$$

Thus every residue class forms an n-tuple,

$$a_o, a_1, a_2, \ldots, a_{m-1} \quad .$$

Any ideal generated by a monic polynomial (polynomial which has the unity element of GF(P) as the leading coefficient) g(x) in the ring of residue classes of polynomials mod-P(x) is a subspace of the vector space V_m over GF(P) such that g(x) divides p(x). If the degree of g(x) is c, then it has been shown (Peterson & Weldon, 1972) that the ideal generated by g(x) forms an (n, k) cyclic code of order P^{m-c}. Also p(x)/g(x) = h(x), where the degree of h(x) is equal to m - c = k. The polynomial h(x) generates an ideal of order P^{m-k} which is a cyclic group called the equivalent dual code of the cyclic code (MacWilliam & Sloane, 1978) generated by g(x). If P(x) is an irreducible polynomial over GF(P) then the set of residue classes mod-P(x) forms the field GF(p^m) of order $q = p^m$. The multiplicative group of this field forms a cyclic group of order (q-1). The element $\alpha \in GF(q)$, all the powers of which give the non-zero element of GF(q), is called a primitive element of the field, and

$$\alpha, \alpha^2, \alpha^3, \ldots \alpha^{q-2}, \alpha^{q-1} = 1 \in GF(q).$$

The primitive element α is a root of the irreducible polynomial P(x) which is called a primitive polynomial over GF(P). Furthermore the elements of GF(q) are the roots of the polynomial $X^{q-1}-1$, thus $x^{q-1}-1$ can be factorized into the irreducible polynomials over GF(q).

An irreducible polynomial m(x) over the ground field F of an extension field is called a minimum polynomial of any element β of the extension field

if $m(\beta) = 0$. It can be shown that all the roots of the minimum polynomial of any element β have the same order (see Peterson & Weldon, 1972). The polynomial $x^{q-1}-1$ can be factorized into minimum polynomials of the elements of $GF(q) = GF(p^m)$ such that

$$x^{q-1}-1 = \prod_{s} M^{(s)}(x)$$

where s runs through all the coset representatives of the cyclotomic cosets of integers mod p^m-1 shown as

$$C_s = \left\{ s, Ps, P^2s, \ldots . , P^{m-1}s \right\}$$

Furthermore, the roots of the polynomial x^n-1, which are called the n^{th} roots of unity, lie in the field $GF(q^m)$ where m is the least integer which n divides q^m-1. Since the generator polynomial of the ideal mod (x^n-1) divides the polynomial x^n-1 such that,

$$(x^n-1)/g(x) = h(x)$$

or

$$x^n-1 = g(x) \; . \; h(x)$$

the n roots of the polynomial x^n-1 which are the roots of the generator polynomial g(x), in error-correcting code terminology are called the zeros of the code (McWilliams & Sloan, 1978) and the other roots which are not zeros of g(x) are the zeros of h(x) and are called the non-zeros of the code.

CHAPTER III

FUNDAMENTALS OF ERROR CONTROL CODING

3.1 The Linear Block Code : Analysis and Construction

The reliable communication of information over an unreliable channel is possible, by protection of the actual transmission messages through giving up a portion of the transmission rate to the controlled insertion of redundancy. The process of adding redundancy to the set of messages $\{M\}$ with a finite number of elements $|M|$, in fact is a mapping α of the elements of $\{M\}$ one-to-one into a set $\{C\}$ with a finite number of elements $|C|$.

Now, a linear code of length n and dimension K is a sub-space V_K of a vector space of all n-tuples V_n over a finite field GF(q) of $|q|$ elements, where q is a power of a prime number. If the message vectors are considered to be the set of the vertices of a K-dimensional unit cube, the mapping α mapps the vertices of this cube into vertices of an n-dimensional cube (see, e.g. Figure (3-1) and Figure (3-2)). The function of the mapping α must be such that it provides the maximum separation between the images of the elements in $\{M\}$, which are the code words of the code. This guarantees the protection needed against the effects of channel disturbance. This mapping can be considered as a linear transformation of the vector space V_K into a vector space V_n over the finite field of GF(q). If the linear transformation is non-singular, then the K images of the basis vectors of V_K are linearly independent. Thus the rank of the linear transformation is equal to K, so K images of the basis vectors of the vector space V_K can span a row space V_C which is the subspace of the vector space of all n-tuples. The linear transformation may be written as:

$$\alpha : \begin{cases} \varepsilon_{1,K} \longrightarrow \varepsilon_{1,n} \\ \varepsilon_{2,K} \longrightarrow \varepsilon_{2,n} \\ \varepsilon_{3,K} \longrightarrow \varepsilon_{3,n} \\ \text{----------} \\ \varepsilon_{K,k} \longrightarrow \varepsilon_{K,n} \end{cases}$$

$\varepsilon_{i,K}$ and $\varepsilon_{i,n}$ are the i^{th} basis vectors of V_K and V_n.

As an example consider V_3 as the message vector space of all 3-tuple vectors, and the linear transformation of V_3 into V_4 (the image space of the space of all 4-tuple vectors), both over GF(2). The images of the basis of V_3 can be as follows:

$$\alpha : \begin{cases} \varepsilon_{1,3} = (1\ 0\ 0) \longrightarrow (1\ 0\ 0\ 1) = \varepsilon_{1,4} \\ \varepsilon_{2,3} = (0\ 1\ 0) \longrightarrow (0\ 1\ 0\ 1) = \varepsilon_{2,4} \\ \varepsilon_{3,3} = (0\ 0\ 1) \longrightarrow (0\ 0\ 1\ 1) = \varepsilon_{3,4} \end{cases}$$

Spanning both the domain (information vectors) and the range of mapping results in:

$$\begin{aligned} &(0\ 0\ 0) \longrightarrow (0\ 0\ 0\ 0) \\ &(0\ 0\ 1) \xrightarrow{\varepsilon_{1,K}\alpha} (0\ 0\ 1\ 1) \\ &(0\ 1\ 0) \xrightarrow{\varepsilon_{2,K}\alpha} (0\ 1\ 0\ 1) \\ &(0\ 1\ 1) \longrightarrow (0\ 1\ 1\ 0) \\ &(1\ 0\ 0) \xrightarrow{\varepsilon_{3,K}\alpha} (1\ 0\ 0\ 1) \\ &(1\ 0\ 1) \longrightarrow (1\ 0\ 1\ 0) \\ &(1\ 1\ 0) \longrightarrow (1\ 1\ 0\ 0) \\ &(1\ 1\ 1) \longrightarrow (1\ 1\ 1\ 1) \end{aligned}$$

This linear transformation, which associates each K-component message with an n-component vector or code word, is referred to as an (n,K) code, and geometrically is depicted by Figure (3-2).

3.1.1 *Generator Matrix of a Linear Block Code*

The linear transformation in 3.1 is non-singular, so then the

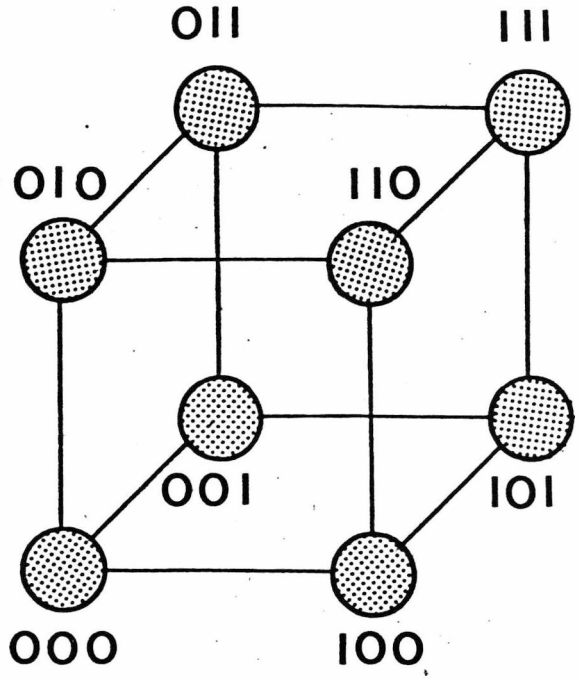


Figure (3-1) : A 3-dimensional cube

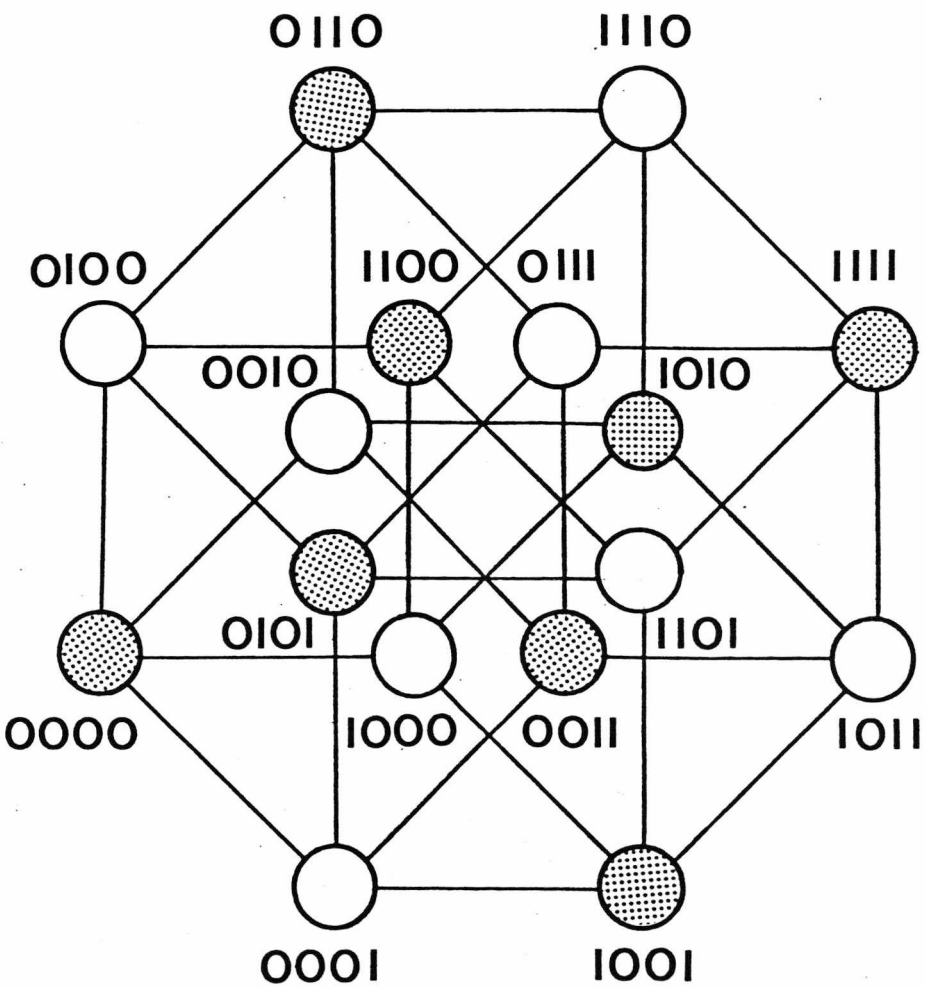


Figure (3-2) : A 4-dimensional cube

K images of the K basis vectors of V_K are linearly independent (Slepian, 1956). The row space of these K linearly independent vectors form the code book of a linear block code V_C ($=\{C\}$) which is a linear subspace of the vector space of all n-tuples V_n. The K linearly independent vectors are usually arranged in a K × n matrix $[G]$ called generator matrix of the code. The generator matrix of a linear code of dimension K and length n can be shown as:

$$[G] = \begin{bmatrix} \varepsilon_{1,n} \\ \varepsilon_{2,n} \\ \varepsilon_{3,n} \\ -- \\ \varepsilon_{K,n} \end{bmatrix} = \begin{bmatrix} g_{1,1} & g_{1,2} & g_{1,3} & - - - - & g_{1,n-1} & g_{1,n} \\ g_{2,1} & g_{2,2} & g_{2,3} & - - - - & g_{2,n-1} & g_{2,n} \\ g_{3,1} & g_{3,2} & g_{3,3} & - - - - & g_{3,n-1} & g_{3,n} \\ -- & -- & -- & & -- & -- \\ g_{K,1} & g_{K,2} & g_{K,3} & - - - - & g_{K,n-1} & g_{K,n} \end{bmatrix}$$

The K rows of the matrix $[G]$ can be any K linearly independent vectors of V_K and V_n. It is possible to choose or transform the matrix $[G]$ into a form such that the first K columns of the matrix $[G]$ form an identity matrix $[I_K]$, where the linear combination of the rows of $[I_K]$ gives the set of all q^K possible information messages. This is called the standard echelon form, shown as:

$$[G_e] = [I_K \; g] = \begin{bmatrix} 1 & & & 0 & g_{1,K+1} & g_{1,K+2} & - - - - & g_{1,n} \\ & & & & g_{2,K+1} & g_{2,K+2} & - - - - & g_{2,n} \\ & & 1 & & & & & \\ & & & & & & & \\ & & 0 & & & & & \\ & & & & & & & \\ & & & 1 & g_{K,K+1} & g_{K,K+2} & - - - - & g_{K,n} \end{bmatrix}$$

The corresponding code generated by this matrix is called a systematic code (Gallager 1968, Slepian 1956). These are

$$N = q^K$$

code words in the code book of a linear code. Assuming that they have an equal a priori probability of being transmitted, then the actual

average information transmitted (per block) is,

$$R_o = \log_2 q^K$$

and the maximum possible information that could be transmitted without redundancy (i.e. without coding) is:

$$R_1 = \log_2 q^n$$

The ratio of these two values of information (entropies) is called the relative entropy or rate of the code (see Rosie 1973), which is therefore:

$$R = \frac{R_o}{R_1} = \frac{\log_2 q^K}{\log_2 q^n} = \frac{K}{n}$$

The redundancy of the code is defined by:

$$C_o = 1 - R = 1 - \frac{K}{n} \quad .$$

3.1.2 *The Hamming Distance and the Corresponding Error Correction-Detection Ability of Linear Block Codes.*

One of the criterion used (not always) to define the capability of a linear block code to correct or detect errors is the concept of distance (Hamming, 1956), which provides the ability to decide to which uncorrupted code word the corrupted one is related. The Hamming weight $W(\bar{x}_i)$, or simply the weight, of any of the q^K distinct code words of a linear block (n,K) code is the number of non-zero elements of the code word. The Hamming distance, or simply distance, d_{ij} between any two code words $\bar{x}_i$ and $\bar{x}_j$ is the weight of the vector $W(\bar{x}_r)$ resulting from digit-by-digit subtraction mod-q of $\bar{x}_i$ and $\bar{x}_j$, shown as

$$d_{ij} = W(x_r) = W(\bar{x}_i \theta_q \bar{x}_j) \quad .$$

In other words, d_{ij} is the number of places in which the code words $\bar{x}_i$ and $\bar{x}_j$ differ.

Consequently, the minimum distance of an(n,K) code d is the minimum value of all d_{ij} where $i \neq j$ (Farrell, 1977 and 1969), denoted:

$$d = \min(d_{ij}) \qquad i \neq j$$

However, for a linear code, the linear combination of any two code words is another code word (Slepian, 1956), thus the minimum distance of the code is the smallest non-zero weight of a code word in the code. Furthermore, in the terms of a geometrical model, the smallest path length (the number of edges) between all pairs of code words is the minimum distance of the code.

If d is the minimum distance of a code, it has been shown that the code is capable of correcting errors of weight

$$t \leq \left[\frac{d-1}{2}\right]$$

where the square bracket means the largest integer not more than (d-1)/2; or the code is capable of detecting

$$D = d - 1$$

errors. Also the code is able to correct up to C errors and detect up to D errors such that

$$d = C + D + 1$$

where $D \geq C$ (Slepian, 1956).

Considering the geometrical model again (Section 3.1), the (n,K) code is t-error-correcting if the code words of the code stand on the vertices of the n-dimensional unit cube, separated by at least (2t + 1) edges. In other words, the code can be "packed" in such a way that every code word can be positioned at the centre of a sphere of radius t where all the q^K spheres are disjoint. As an example, consider the (n, K, d) $\equiv$ (8, 2, 5) code. Each code word is at the centre of a sphere of radius

$$t = \left[\frac{d-1}{2}\right] = \left[\frac{5-1}{2}\right] = 2$$

such that each sphere contains

$$1 + \sum_{i=1}^{t} \binom{n}{i} = 1 + \sum_{i=1}^{2} \binom{8}{i} = 37$$

vertices of the 8-dimensional unit cube.

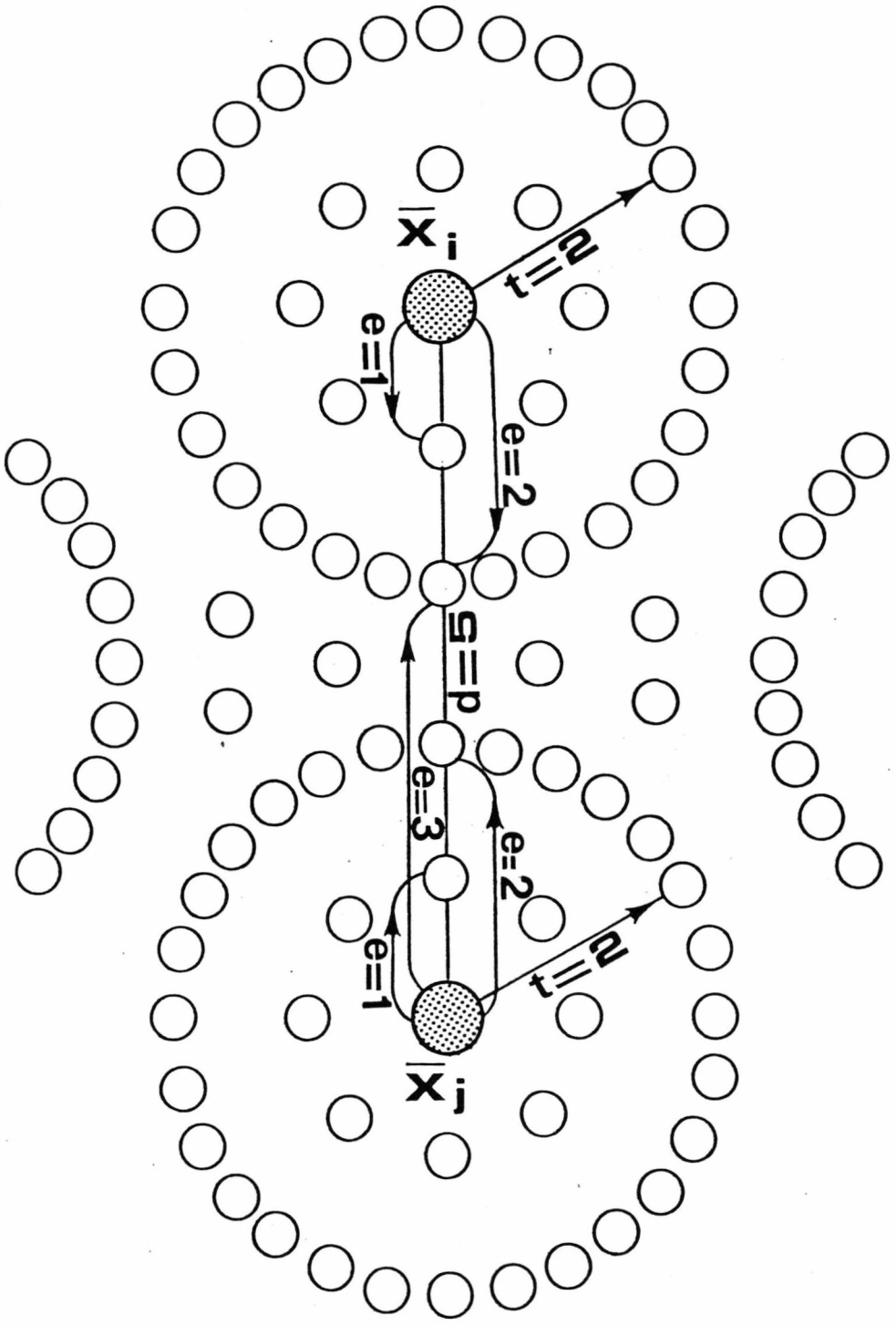


Figure (3-3) : Two code words of the (8, 2, 5) code.

Any two or less errors on any code word do not remove the corrupted code word out of the sphere to which it "belongs", so it may be recovered unambiguously.

If the code is just error correcting, then every error weight $> t = 2$ removes the code word out of its sphere, and any attempt to recover it results in post decoding error (Berlekamp, 1968). If the code is an e-error detecting code, then every error weight $D \leq d - 1 = 4$ cannot alter any code word into another, i.e. the corrupted words are all non code words, so the errors can be detected. If the code is $D = 3$ error detecting and $C = 1$ error correcting, such that $d = C + D + 1 = 1 + 3 + 1 = 5$, then any single error can be corrected unambiguously, but double errors and triple errors can only be detected. Any attempt to correct a double error results in post decoding errors in this case, as is also the case for the triple errors.

The Hamming distance criterion is not always the best one to use, however. The Lee criterion (Lee distance) is more suitable in some cases than Hamming distance (Berlekamp, 1968). The two criteria coincide in the binary case ($q = 2$).

Furthermore, it can be shown (Slepian, 1956) that though codes with the same (n,K) parameters may have different minimum distances, it does not necessarily follow that the code with the larger minimum distance will have superior performance. Most of the linear block codes with Hamming distance d can correct not only the error patterns of weight

$$t \leq \left[\frac{d-1}{2}\right]$$

but also many error patterns of weight bigger than t (Turner-Olanyan, 1976). Even if an (n,K) linear block code only has $d = 1$, using the weight distribution (Hobbs, 1965) it is shown in Appendix II that nearly all of the single errors and a few of the bigger error patterns are correctable. Thus, a better figure of merit for a linear block code,

considering different codes with the same block length and the same number of information symbols, would be the correctability performance of the code or in other words the probability of error after decoding (post decoding error rate). The code which has the smallest of these probabilities is called an optimum code (Fontaine & Peterson 1959, Bose & Kubler, 1958) with respect to the others.

3.1.3 *The Parity Check Matrix of a Linear Block Code*

The parity check matrix $[H]$ of a linear block code generated by the generator matrix $[G]$ is the coefficient matrix of a system of n-K linear homogeneous equations called the parity check equations of the code (Van Lint 1970, Slepian 1960), such that the set of q^K code words generated by the matrix $[G]$ are the set of solutions of the n-K equations. The matrix $[H]$ in standard echelon form is shown as:

$$[H] = [-g^T \quad I_{n-K}] = \begin{bmatrix} -g_{1,K+1} & -g_{2,K+1} & \cdots & -g_{K,K+1} & 1 & & \\ -g_{1,K+2} & -g_{2,K+2} & \cdots & -g_{K,K+2} & & 1 & \\ & & & & & & \ddots \\ -g_{1,n} & -g_{2,n} & \cdots & -g_{K,n} & & & 1 \end{bmatrix}$$

The above property and construction of the $[H]$ matrix arises from the rules governed by the n-K right-hand columns of the generator matrix $[G]$, and the way that the information digits of a message combine the rows of $[G]$ in standard echelon form (see Wiggert 1978, Lucky, and Salz & Weldon 1968). Thus for any code work $\bar{x} \in C$ the product

$$\bar{x}.[H^T] = \bar{x}\begin{bmatrix} g^T \\ - - - \\ -I_{n-K} \end{bmatrix} = [O] \quad .$$

The result of this multiplication is an important property of linear codes. That is, for every code work $\bar{x} \in C$, (C is the code book of the

linear block code) of weight $W(\overline{x})$, the linear combination of those columns of $[H]$ corresponding respectively to the positions containing non-zero elements in $\overline{x}$ results in an all-zero (n-K)-tuple vector. Conversely, for every linear combination of $W(\overline{x})$ columns of $[H]$ matrix which results in an all zero (n-K) tuple vector there is a code word in C (Peterson & Weldon, 1972).

Since in a linear block code the minimum Hamming distance of the code is the weight of the minimum weight non-zero code word (Lucky and Salz & Weldon, 1968), then no linear combination of less than d columns of the $[H]$ matrix of a linear block code which results in an all-zero (n-K) tuple vector; in other words, every d-1 or less columns of the parity check matrix $[H]$ of a linear block code are linearly independent. Furthermore, for a linear code which is capable of correcting up to t errors it has been stated that

$$t = \left[\frac{d-1}{2}\right]$$

and

$$d = 2t + 1 \quad .$$

Then it can be said that a code is a t error correcting code if and only if every 2t column of $[H]$ is linearly independent (Sack, 1958).

As a consequence of the above discussion, for any code word $\overline{x} \in C$, $\overline{x} \cdot [H^T] = O$ and for any n tuple $\overline{y} \in C$, which is not a code word,

$$\overline{y} \cdot [H^T] \neq [O] = [S] \quad .$$

If $\overline{y}$ is a code word $\overline{x}$ corrupted by the n-tuple noise vector $\overline{e}$, then

$$\overline{y} = \overline{x} \oplus_q \overline{e}$$

so

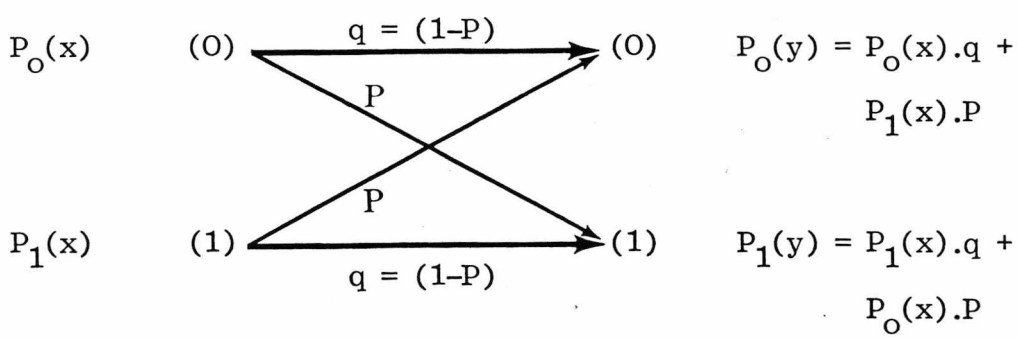
$$\bar{y} \cdot [H^T] = (\bar{x} \theta_q \bar{e}) \cdot [H^T] = \bar{x} \cdot [H^T] \theta_q \bar{e} \cdot [H^T]$$

$$= [O] \theta_q \bar{e} \cdot [H^T] = \bar{e} \cdot [H^T] = [S]$$

The (n-K) tuple vector $[S]$ is called the syndrome related to the error vector $\bar{e}$. It is simply verified that, for a t-error-correcting linear block code all the syndromes of the error vectors of weight $\leq$ t must be unique; or in other words, every linear combination of t or less columns of $[H]$ must be a unique non-zero (n-K) tuple vector in the vector space V_{n-K} (Hashim, 1974). Thus, in general, if α_i be the number of unique syndromes of the different error vectors of weight i, then the error correctability of a linear block code over a binary symmetric channel (Berlekamp, 1968) can be shown to be:

$$P_C = \sum_{i=o}^{n} \alpha_i \, P^i (1 - P)^{n-i} \qquad \left(\text{where, } \sum_{i=o}^{n} \alpha_i \leq q^{n-K}\right)$$

(Hobbs 1965, Fountain 1959) where P_C and P are the probability of correct recovery (decoding) of the corrupted code word by noise, and the channel cross-over probability, respectively. The following diagram illustrates the various source, channel and sink probabilities, in the case of a binary symmetric channel (BSC).



where $q = P(y/x) = P(O/O) = P(1/1)$

$P = P(y/x) = P(1/O) = P(O/1)$.

The set of (n-K) linear independent rows of the parity check matrix $[H]$ of a linear block code, which is orthogonal to every code word of the code, spans a sub-space V_{n-K} of dimension (n-K) of the vector space of all n-tuple vectors. This subspace forms an (n, n-K) linear block code called the dual code of the original code. If in the original code n = 2K the code and its dual code are identical and the code is called a self-dual code.

3.1.4 *The Modular Representation of a Linear Block Code, and Equivalent Codes.*

A linear block (n, K, d) code over GF(q) is the row space of the K linear independent vectors (rows) of its generator matrix $[G]$. The matrix $[G]$ has n columns out of $q^K - 1$ different types of non-zero columns. If matrix $[M]$ is the standard variable q-ary truth table, then $[M^T]$ has as its columns all $q^K - 1$ possible types of column in increasing number order from far left to the right.

If the i^{th} column of $[M^T]$ be column type i then a $1 \times (q^K-1)$ matrix

$$N = [n_1, n_2, \ldots, n_i, \ldots n_{q^K-1}]$$

in which every element n_i is the number of the column type i of the matrix $[G]$ is called the modular representation of the linear block code generated by $[G]$. Any set of K linearly independent vectors of the sub-space (code) generated by $[G]$ spans the same code. Thus any elementary row transformation on $[G]$, or multiplication on the left by an elementary (non-singular) $K \times K$ matrix $[R]$ results in a different generator matrix but an equivalent (in Hamming distance sense) code (Fontaine 1959). So,

$$[R][G] = [G_r]$$

and

$$[G_r] \simeq [G] .$$

An elementary column transformation on $[G]$, or multiplying $[G]$ on the

right by an n × n permutation matrix $[S]$, changes the code in a trivial manner,

$$[G] \cdot [S] = [G_S]$$

and

$$[G] \simeq [G_S] \quad .$$

These two codes are called column equivalent codes. Two codes are called equivalent if the generator matrix of one can be obtained from another by both column and rwo elementary transformation.

$$[R] \cdot [G] \cdot [S] = [G_{rs}]$$

and

$$[G] \simeq [G_{rs}] \quad .$$

3.1.5 *Coset Decomposition of a Linear Block Code*

The equivalence relation "has the same syndrome" partitions the set of q^n vectors of the vector space V_n into the q^{n-K} equivalence classes known as cosets. The first coset is the linear block (n,K) code itself, which is the subspace V_K of the vector space V_n of all n-tuple vectors (Slepian 1960, Lucky & Salz & Weldon, 1968). As mentioned earlier (see Chapter II, Section 2.6.7) the coset leaders are chosen to be the minimum weight (the most likely errors) vectors of V_n.

Since there is a one-to-one relation (correspondence) between a syndrome and a coset leader (McWilliams & Sloane 1977) such that,

$$[S] \longleftrightarrow [e]$$

and also

$$[S] = [e] \quad [H^T]$$

then a complete list of the set of coset leaders and corresponding syndromes leads to the systematic search decoding method for linear block codes which is in fact a maximum liklihood decoding and is optimum.

Whenever q^{n-K} has a reasonable value ($< \sim 1000$), this method (compute the syndrome, consult the list for the error pattern, and subtract the error pattern from the received n-tuple) is a practical decoding method.

For a t-error-correcting linear block (n, K, d) code, the number of coset leaders of weight $\leq t$ over a binary symmetric channel is given by

$$\sum_{i=0}^{t} \binom{n}{i}$$

and the number of coset leaders of weight $> t$ is equal to

$$2^{n-K} - \sum_{i=0}^{t} \binom{n}{i} .$$

Therefore the word error probability regardless of the number of errors in the code words is lower bounded by:

$$P_e \geq 1 - \left[\sum_{i=0}^{t} \binom{n}{i} P^i (1-P)^i \right] - \left\{ \left[2^{n-K} - \sum_{i=0}^{t} \binom{n}{i} \right] P^{t+1} (1-P)^{t+1} \right\} .$$

CHAPTER IV

BEST KNOWN LINEAR BLOCK CODES: CONSTRUCTION TECHNIQUES, AND SOME RELATED CONCEPTS.

4.1 General

The search for efficient techniques for constructing codes which with a rate equal to channel capacity have a vanishingly small error probability with increasing block length, has not been fruitful yet. The codes of Elias and similar followers, which have a mathematical construction (algebraic codes) for a fixed rate, provide a lower bounded value of distance to code length. They have a vanishingly small error probability (Cooper 1977), but their rates are far removed from the ideal channel capacity and also from the Varshamov-Gilbert bound. But good moderate length efficient codes which have a mathematical structure and form the class of algebraic codes have been constructed. These codes, due to their mathematical properties, have encoding and decoding equipment of moderate complexity.

In this section a brief review of the best known of these linear block codes and some related concepts are given.

4.2 Minimum Distance Bounds for Linear Block Codes

In spite of the shortcomings of the Hamming distance criterion (Section 3.1.3), the minimum Hamming distance of a linear block code is an important parameter for evaluation of the performance of the code. Thus it is clearly of interest to know the ultimate value of the minimum distance for a certain block length and number of information digits. These values have been given by various sources in the form of a functional relationship between the code parameters n, k and d. These relationships are in the form of bounds which relate any one of the given parameters in terms of the remaining two.

4.2.1 *The Hamming Bound*

The Hamming sphere-packing bound (Hamming 1950) is an upper bound on the maximum number of code words N of an (n, d) code. The code words of a t-error-correcting code can be packed in such a way that every code word is positioned at the center of a sphere of radius

$$t = \left[\frac{d-1}{2}\right]$$

in an n-dimensional space. Since there are N distinct code words, then the N disjoint spheres contain:

$$N \cdot \sum_{i=o}^{t} \binom{n}{i}(q-1)^i$$

vertices of the n-dimensional cube. The total number of vertices of the n-dimensional code is q^n, so that,

$$N \cdot \sum_{i=o}^{t} \binom{n}{i}(q-1)^i \leqslant q^n \qquad (4\text{-}1)$$

For a linear binary block code q = 2 and $N = 2^k$ then,

$$2^{n-k} \geqslant \sum_{i=o}^{t} \binom{n}{i} \qquad (4\text{-}2)$$

Taking the logarithm of both sides

$$\frac{k}{n} \leqslant 1 - \frac{1}{n}\log_2\left[\sum_{i=o}^{t}\right]\binom{n}{i}$$

The asymptotic form of this bound (Peterson & Weldon 1972, Appendix A, see also the simple proof given by Park 1969), is

$$\frac{k}{n} = 1 - H\left(\frac{t_h}{n}\right)$$

where H(x) is the entropy function and t_h is the largest integer t for

which the above inequality holds. The Hamming sphere-packing bound has been refined by Wax (1959) and Johnson (1962). The Johnson upper bound is tighter than the Wax upper bound except for some values of the parameters of (n, d). Also, for some values of k and d, the combined form of the bound with the Greismer bound gives a stronger lower bound than the Greismer and Hamming bounds individually.

4.2.2 *Perfect Codes*

An (n, k, d) code is a perfect code if there is an integer t such that the coset leaders of this code include all the coset leaders of weight $\leqslant t$ and none of the coset leaders of weight $\geqslant t + 1$. These codes satisfy the Hamming sphere packing bound exactly, or in other words, the (n, k, d = 2t + 1) perfect code can partition the q^n vertices of an n-dimensional unit cube perfectly into q^k disjoint spheres of radius t (Vasiliev 1962). Therefore, in a perfect code

$$N \cdot \sum_{i=o}^{t} \binom{n}{i}(q-1)^i = q^n$$

and the word error probability of Section 3.1.6 becomes an equality, as

$$P_e = 1 - \sum_{i=o}^{t} \binom{n}{i} P^i (1-P)^{n-i}$$

These optimum codes were first introduced by Hamming (1950) as the linear binary single error correcting perfect codes (see Section 4.2.4). Golay (1949 & 1954) discovered a linear binary triple-error-correcting code and a linear double-error-correcting code over GF(3). This introduced the possibility of generalizing the Hamming codes (see Section 4.3.3). Vasiliev (1962) constructed the class of binary single error correcting perfect codes which includes both linear and non-linear codes. This was in contrast to the idea of the non-existence of more perfect codes, conjectured by Shapiro & Slotnick (1959). The method of

constructing linear and non-linear single error correcting binary perfect codes by Vasiliev was generalized by Schonheim (1968) over $GF(p^m)$ with p a prime integer.

Van Lint (1970), and Tietavainen (1973), in two complementary papers, proved that the known perfect codes are the only existent perfect codes over $GF(p^m)$.

4.2.3 *Repetition Codes*

The class of repetition codes over GF(p) is a trivial example of the 1/n rate perfect code, with parameters (n, 1, n) and with odd block length. These codes can correct all the errors of weight $t \leqslant \frac{n-1}{2}$ and no errors of weight $t > \frac{n-1}{2}$. The class of repetition codes have the highest error correctability.

4.2.4 *Hamming Codes*

The Hamming codes were first introduced by Hamming (1950).

The linear binary Hamming codes over GF(2) have parameters

$$n = 2^m - 1, \quad K = n - m, \quad d = 3,$$

where m is any integer $\geqslant 2$, which represents the number of parity checks of the code.

The $2^m - 1$ columns of the parity check matrix of the Hamming code from the all $2^m - 1$ distinct binary m-tuple vectors, each being a non-zero binary number in the range 1 up to $2^m - 1$. Therefore no two columns are the same, so the minimum distance of the code is d = 3. There are $2^m - 1$ columns each of which is a distinct syndrome for a distinct single error, so the code is perfect.

Hamming codes have the desirable property that

$$R = k/n \longrightarrow 1 \quad \text{when} \quad n \longrightarrow \infty,$$

but on the other hand $d/n \longrightarrow 0$ with $n \longrightarrow \infty$, therefore the error correctability of these codes is relatively poor. Golay (1949, 1958) was the

first to generalize the Hamming codes over GF(p) and $GF(p^2) = GF(q)$, with parameters,

$$\left[n = (q^m - 1)/(q - 1),\ n - m,\ d = 3 \right].$$

In 1959 Cock generalized the Hamming codes over GF(q) (q is a power of a prime number) with the same parameters

$$\left[n = (q^m - 1)/(q - 1),\ n - m,\ d = 3 \right].$$

The code obtained by adding an overall parity check to the Hamming binary single error correcting code has parameters ($n = 2^m$, $n - m$, $d = 4$), and is called the extended single-error-correcting, double-error-detecting Hamming code. This code has been extensively used by Elias (1954) to construct an infinite family of good codes (see Section 4.6.1). Hamming codes are equivalent to the cyclic codes (Parang (1957), Abramson (1959 & 1961), and others).. Hamming codes form a subclass of primitive BCH codes; for the binary case, the parity check matrix is

$$H = \left[\alpha^{2^m-2}, \ldots, \alpha^2, \alpha, 1 \right],$$

where α is a primitive element of $GF(2^m)$, (see also Peterson & Weldon 1972, p. 221).

4.2.5 *Golay Codes*

The only non-trivial binary linear perfect code other than the Hamming single-error-correcting codes, has been found by Golay (1949). This code has parameters (23, 12, 7), and since it is perfect, all the 23-tuple vectors of the 23-dimensional space can be packed by the code words of this code into 2^{12} sphere of radius 3 with no overlap, i.e.

$$2^{12}\left[1 + \binom{23}{1} + \binom{23}{2} + \binom{23}{3}\right] = 2^{23}.$$

The parity check matrix [H] of this code forms an 11×23 matrix, therefore the

$\sum_{i=o}^{3} \binom{23}{i}$ linear combinations of the columns of the [H] matrix gives the 2^{11} distinct syndromes of all single, double, and triple errors, i.e.

$$\sum_{i=o}^{3} \binom{23}{i} = 2^{11} \quad .$$

The algebraic (cyclic) construction of this code is given as follows (McEliece 1977):

The ideal generated either by the polynomial $g_1(x)$ or $g_2(x)$, where

$$g_1(x) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1$$

$$g_2(x) = x^{11} + x^{10} + x^6 + x^5 + x^4 + x^2 + 1$$

in the ring of residue classes of polynomials modulo the polynomial $x^{23} - 1$ is the cyclic form (see Section 4.5.1) of the (23, 12, 7) Golay code. The roots of the generator polynomials $g_1(x)$ and $g_2(x)$ are the roots of the code polynomials (words, see Section 2.10) and also the roots of the polynomial $x^{23} - 1$, i.e.

$$x^{23} - 1 = (x - 1)\, g_1(x)\, g_2(x)$$

The 23 roots of $x^{23} - 1$, known as 23 roots of unity, lie in the elements of the $GF(2^{11})$.

The code obtained by adding an overall parity check to the (23, 12, 7) Golay code has parameters (24, 12, 8) and is the extended Golay code (see also Leech 1967 & 1971 and Pless 1958). Golay also found the (11, 6, 5) double-error-correcting ternary perfect code.

Furthermore, the generator matrix of the (23, 12, 7) Golay code can be put (Leech 1964) into the matrix form:

$$\underset{\text{Golay}}{[G]} = \left[\begin{array}{c|c} I & C \\ \hline & I \end{array}\right]$$

here C has the property that each row is a cyclic shift of the previous row by one place. The matrix C is called a Circulant matrix and the code generated by a generator matrix of this type is a *Circulant* code. The work by Leech followed by that of Karlin (1969, 1970) and Pless (1969, 1972) resulted in a large number of binary codes generated by circulants, including codes over GF(3).

4.2.6 *Quasi-perfect Codes*

An (n, k, d) code is a quasi-perfect code. If there is an integer t, such that the coset leaders of this code include all the cosets of weight $\leqslant t$, some of weight $t + 1$ and none of weight $> t + 1$, then the code is quasi-perfect. The inequality of Section 3.1.6, which is the word error probability, is satisfied by quasi-perfect codes. Therefore,

$$P_e = 1 - \left[\sum_{i=o}^{t} \binom{n}{i} p^i (1 - p)^{n-i} \right] - \left\{ \left[2^{n-k} - \sum_{i=o}^{t} \binom{n}{i} \right] p^{t+1} (1 - p)^{t+1} \right\}$$

Considering the properties of the [H] matrix of a linear block code Wagner (1966a, 1966b, 1967), obtained several quasi-perfect codes, also the double-error-correcting B.C.H. codes have been shown to be quasi-perfect (Gorenstein & Peterson & Zieler 1960, see also McWilliams & Sloane 1978, p. 279).

4.2.7 *The Plotkin Bound*

The Plotkin bound (Plotkin 1960) is an upper bound on the maximum number of code words N of an (n, d) code. The derivation of this bound is based upon the fact that the average weight,

$$n\, q^{k-1}(q - 1)/(q^k - 1)$$

of a code word of the code over GF(q) is at least equal to the minimum distance d of the code, thus

$$d \leqslant nq^{k-1}(q - 1)/(q^k - 1)$$

where $qd > (q - 1)n$.

For a linear (n, d) code over GF(q), $N = q^k$ therefore the above inequality can be written as:

$$N = q^k \leqslant qd \;/\; qd + n(1 - q)$$

and for a binary code

$$N = 2^k \leqslant \frac{2d}{2d - n} \qquad 2d > n \ .$$

Also for a non-linear (Plotkin 1960, see also Farrell 1969), code the number of code words

$$\begin{cases} N_{even} = 2m \leqslant \dfrac{2d}{2d - n} \\ N_{odd} = 2m - 1 \leqslant \dfrac{n}{2d - n} \end{cases}$$

where $2d > n$

and m is an integer.

Furthermore, Elias (1955) extended the Plotkin bound (see Peterson and Weldon, p.78), and showed that the number of parity checks required for an (n, d) linear block code is given by

$$n - k \geqslant (qd - 1) \;/\; (q - 1) \; - 1 - \log_q d$$

where $n \geqslant (qd - 1) \;/\; (q - 1)$

and for a linear binary code

$$n - k \geqslant 2d - 2 - \log_2 d$$

where $n \geqslant 2d$.

The Plotkin bound is weaker than Hamming sphere-packing bound at higher rates and tighter at lower rates (Berlekamp 1968).

4.2.8 *The Elias Bound*

The Elias upper bound (Elias 1960) is a tighter bound on the maximum minimum distance of an (n, k) block code than Hamming bound and Plotkin bound over the range of medium rate (Berlekamp 1968).

The derivation of this bound is based upon both the Hamming sphere-packing and the Plotkin average distance bounds, and is given by

$$d \leq 2t(1 - \frac{t}{N})(\frac{N}{N-1}) ,$$

where N is the smallest integer satisfying the inequality

$$N \geq 2^k \sum_{i=o}^{t} \binom{n}{i} / 2^n$$

and t is any integer such that,

$$2^{n-k} < \sum_{i=o}^{t} \binom{n}{i} .$$

This bound is uniformly tighter than Hamming and Plotkin bounds. The difference is considerable in the region of medium rate, and at high and low rates the difference becomes smaller. However, tighter upper bounds have been derived by Levenstein (1975), and also by McEliece, Rumsey and others (1976), and is given by

$$R \leq H\left[1/2 - \sqrt{d/n(1 - d/n)}\right],$$

where R is the rate of the code and H is the entropy function. This is at present the best (closest to the best lower bound) upper bound.

The discussed upper bounds are not constructive bounds, and therefore are not necessarily attainable over the whole range of code parameters. In other words, an optimum code does not necessarily meet that upper bound which is the least value of all upper bounds for the given parameters (n, d). However, the lower bounds (see next Section) are constructive bounds in the sense that it is possible to construct a code that meets a lower bound (Pierce 1967).

The bounds reviewed above are plotted in Figure 4-1.

4.2.9 *The Varshamov-Gilbert Bound*

Applying a systematic procedure to the construction of a linear block code, Varshamov (1957) and Sacks (1958) derived a lower bound on the number of code words of an (n, d) code. This bound is a refinement of a bound found by Gilbert(1952).

The derivation of this bound is based upon a property of the columns of the parity check matrix [H] of a linear block (n, k, d) code over GF(q), which is that no set of d - 1 or fewer columns of the matrix [H] is linearly dependent. For a linear (n, k, d) block code this is possible (see Peterson & Weldon, 1972) if

$$\binom{n}{1}(q-1) + \binom{n}{2}(q-1)^2 + \ldots + \binom{n}{d-2}(q-1)^{d-2} \geqslant q^c - 1$$

or

$$\sum_{i=0}^{d-2} \binom{n}{i}(q-1)^i \geqslant q^{n-k}$$

where $q^c - 1$ is all possible non-zero c-tuple vectors included in the columns of the parity check matrix [H].

The asymptotic form of this bound is given (see Section 4.2.1) by

$$1 - \frac{k}{n} \leqslant H\left(\frac{d-n}{n}\right) \quad .$$

A possible improvement on the Varshamov lower bound has been proposed by Hashim (1974).

Thus the Varshamov-Gilbert bound, which is a constructive bound states:

For any fixed integer R where $1 \geqslant R \geqslant 0$ there exists a binary (n, k, d) code with $R \geqslant k/n$ and $d/n \geqslant H^{-1}(1 - R)$.

However, all the best constructive long codes are far from this bound. Therefore, the Varshamov-Gilbert bound leads to the definition:

"A family of codes over GF(q) is said to be good if it contains

an infinite sequence of codes ϕ_1, ϕ_2, ... where ϕ_i is an (n_i, k_i, d_i) code over GF(q) and both $R_i = k_i/n_i$ and d_i/n_i has a positive limit as $i \longrightarrow \infty$ " .

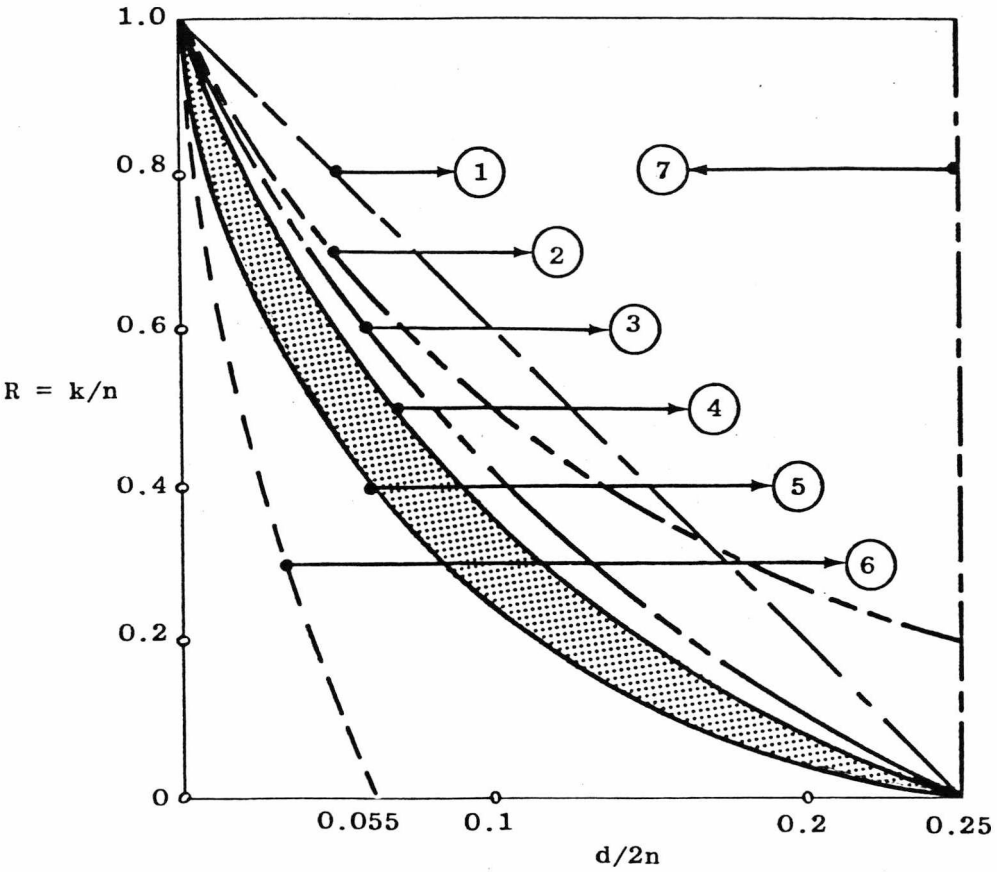


Figure (4-1) : Bounds on minimum distance for the best binary block codes

(1) Plotkin's upper bound

(2) Hamming's upper bound

(3) Elias's upper bound

(4) McEliece's et al upper bound

(5) Varshamov-Gilbert's lower bound

(6) Justesen's lower bound (see Section 4.6.3)

(7) Plotkin's average distance bound

4.3 Cyclic Codes

The most extensively studied class of linear block codes are the cyclic codes.

Cyclic codes have considerable mathematical structure which simplifies their implementation and analysis. However, this family of linear block codes does not have a ratio of minimum distance to block length bounded away from zero, i.e.

$$d/n \longrightarrow 0 \quad \text{as} \quad n \longrightarrow \infty \quad .$$

Also, it has been shown by Berlekamp (1972) that for BCH codes, which is one of the most important classes of linear cyclic codes

$$d \sim \frac{2n\log_e R^{-1}}{\log_{10} n}$$

which is a big disadvantage for these codes. But, still it is not known whether all long cyclic codes are bad (see Berman (1967), McEliece (1970), Chen (1969)).

Cyclic codes were first discovered by Prange (1957, 1958) and investigated by Peterson & Brown (1961), and Lucky, Salz & Weldon (1968), Berlekamp (1968), and others.

4.3.1 *Cyclic Code Analysis and Construction*

A linear subspace C of the n-tuple vector space V_n is called a cyclic code if any cyclic shift of any code word is a code word.

The isomorphism property of the n-dimensional vector space V_n over GF(q) and the ring of residue classes of polynomials modulo $x^n - 1$ (see Section 2.8) gives the possibility of the following definition:

An (n, k, d) linear cyclic code C is an ideal (principal ideal) $< g(x) >$ in the ring of residue classes of polynomials $F[x]/p(x)$ modulo the polynomial $p(x) = x^n - 1$ over GF(q). Thus the ideal C (cyclic code) is generated by the polynomial g(x) which is a monic

polynomial of degree $n - k = c$ and divides $p(x) = x^n - 1$, i.e.

$$(x^n - 1)/g(x) = h(x) \ ,$$

where $n - c = k$ is the degree of $h(x)$.

Therefore, if $c(x) = c_o + c_1 x + c_2 x^2 + \ldots + c_{n-1} x^{n-1}$ is a code polynomial in the ideal $< g(x) >$, then for every polynomial $f(x) \in F[x]$, the polynomial $f(x)\, c(x) \in < g(x) > \longleftrightarrow C \bmod (x^n - 1)$ and also every $x\, c(x) \in < g(x) > \longleftrightarrow C \bmod (x^n - 1)$ is in C. Hence if

$$c(x) = c_o + c_1 x + c_2 x^2 + \ldots + c_{n-1} x^{n-1} \in < g(x) >$$

$$\longleftrightarrow (c_o, c_1, c_2, \ldots, c_{n-1}) \in C$$

then $x\, c(x) = c_o x + c_1 x^2 + c_2 x^3 + \ldots + c_{n-1} x^n = c_{n-1} + c_o x + \ldots$

$$+ c_{n-2} x^{n-2} \longleftrightarrow c_{n-1}, c_o, c_1, \ldots, c_{n-2} \in C$$

which is a right cyclic shift of the coefficients of $c(x)$ or a cyclic shift of the related code word by one place.

Therefore the $k = n - c$ different polynomials $g(x)$, $xg(x)$, $x^2 g(x)$, ..., $x^{n-c-1} g(x)$ are linearly independent, so the vectors related to these polynomials form the rows of the generator matrix $[G]$ of the cyclic code $C \longleftrightarrow < g(x) >$, so if $g(x) = g_o + g_1 x + \ldots + g_c x^c$

$$[G] = \begin{bmatrix} g(x) \\ x\, g(x) \\ \vdots \\ x^{n-c-1} g(x) \end{bmatrix} \longleftrightarrow \begin{bmatrix} g_o, g_1, \ldots, g_c & & & 0 \\ & g_o, g_1, \ldots, g_c & & \\ & & \ddots & \\ 0 & & & g_o, g_1, \ldots, g_c \end{bmatrix}$$

Thus it is evident that a polynomial is a code polynomial if it is divisible by $g(x)$.

Since $(x^n - 1) \,/\, g(x) = h(x)$ then

$$x^n - 1 = g(x) \cdot h(x) = 0 \bmod x^n - 1.$$

Therefore if c(x) is a code polynomial then

$$c(x)\ h(x) = f(x)\ g(x)\ h(x) = 0 \mod (x^n-1)\ .$$

From this property, it is evident that the polynomial h(x) generates an ideal which is the null space of the ideal generated by g(x). But, the polynomial multiplication and inner product of the related vectors, is different (if two polynomials a(x) and b(x) are orthogonal, i.e. $a(x)\ .\ b(x) \equiv 0 \mod x^n-1$, then the related n-tuple vector $\vec{a}$ is orthogonal to n-tuple vector $\vec{b}$ in reverse order (see also Section 2.10, Van Lint 1973 and Peterson & Weldon (1972)). Thus the parity check matrix for the cyclic code generated by g(x) where

$$h(x) = (x^n - 1) \,/\, g(x) = h_k\, x^k + \ldots + h_1 x + h_o \text{ is}$$

$$[H] = \begin{bmatrix} h(x) \\ xh(x) \\ \text{------} \\ x^{n-k-1}h(x) \end{bmatrix} \longleftrightarrow \begin{bmatrix} 0 \quad h_k, --, h_2, h_1, h_o \\ h_k, --, h_2, h_1, h_o \\ \\ h_k, --, h_2, h_1, h_o \quad 0 \end{bmatrix}$$

(Note: the coefficients g_c and h_x are the unity element of GF(q)). The generator polynomial g(x) of a linear cyclic code over GF(q) of length n divides the polynomial x^n-1, therefore the n - k = c roots of g(x) lie among the n different roots of x^n-1. These n roots are also the elements of the extension field $GF(q^m)$ of the ground field GF(q). They form a cyclic group of order n called the n^{th} roots of unity. m is called the multiplicative order of q and is the smallest integer such that n divides $q^m - 1$. Thus the polynomials g(x) and $x^n - 1$ are completely factorized over $GF(q^m)$, such that

$$(x^n - 1) = \prod_{i=0}^{n-1} (x - \alpha^i) = \prod_{c_s} M^{(s)}(x)$$

(see Section 2.10) where α is a primitive n^{th} root of unity, and s runs

through the set of cyclotomic coset representatives mod-n. Similarly, g(x) can be factorized as

$$g(x) = \prod_{i \in k} (x - \alpha^i) \quad \text{where } K \subset \{ 1, 2, \ldots n-1 \}$$

and $\alpha^i \in GF(q^m)$; this means that g(x) is the product of the minimum polynomials of α^i for all $i \in k$.

4.3.2 *Shortened Cyclic Codes*

The factorization of x^n-1 over $GF(q^m)$ into minimum polynomials $M^{(s)}(x)$ does not always give many factors, thus not many generator matrices can be formed. Therefore, for some values of (n, k) a shortened version of an (n+i, k+i) linear cyclic code can be used to obtain an (n, k) code. The first i columns and rows of the generator matrix of the (n+i, k+i) code are deleted, which is the same as detecting the first i columns of the $[H]$ matrix of cyclic codes.

The shortened form of a cyclic code is not a cyclic code but can be encoded and decoded as the original cyclic code (see Lucky & Salz & Weldon). Also, it has the same minimum distance as the original cyclic code (the same coset leaders).

The cyclic codes which are ideals in the ring of polynomials modulo a polynomial f(x) other than $x^n - 1$ are called Pseudo Cyclic codes. Shortened cyclic codes are identical to the pseudo cyclic codes, and also every pseudo cyclic code having minimum distance greater than two is a shortened cyclic code (theorems 8.9 & 8.10 Peterson & Weldon 1972). Kasami (1969) has shown that arbitrarily long pseudo cyclic codes exist which have parameters that meet the Varshamov-Gilbert bound.

Cyclic codes are invariant under the one place cyclic permutation group (see Peterson & Weldon, Section 8.11). An $(n, k) = (mn_o, mk_o)$ linear code which is invariant under the n_o places cyclic permutation group is called a quasi cyclic code (Townsend &

Weldon 1967); that is, the cyclic shift of every code word by n_o places is again another code word. The quasi cyclic linear (n, n/2) codes found by Chen (1969)$_a$ have the maximum value of the minimum distance for (n, n/2) linear codes. Quasi cyclic codes also have been studied by Karlin (1969) and Hoffner & Redy (1970), Chen & Peterson & Weldon (1969)$_b$. The latter have shown the existence of long but not arbitrarily long quasi cyclic codes that meet the Varshamov-Gilbert bound.

4.3.3 *Extended Cyclic Codes*

A cyclic (n, k, d) code generated by the generator polynomial g(x) which is not divisible by (x-1) can be extended by adding a new row and column to the $[H]$ matrix. The generator and parity check matrix of the extended cyclic code respectively are:-

$$[G_{ext}] = \left[\begin{array}{c|c} G_{aug} & \begin{matrix} g^{(1)}_{aug} \\ \vdots \\ g^{(1)}_{aug} \end{matrix} \end{array}\right] \quad \& \quad [H_{ext}] = \left[\begin{array}{c|c} H_{aug} & \begin{matrix} 0 \\ 0 \\ \vdots \\ 0 \end{matrix} \\ \hline 1\ 1\ ----\ 1 & 1 \end{array}\right]$$

where g(x) and G_{aug} are the generator matrix of the original cyclic code when augmented (see Berlekamp 1970), $[G_{ext}]$ and $[H_{ext}]$ are respectively the generator matrix and parity check matrix of the extended cyclic code, and H_{aug} is the parity check matrix of the original code when augmented. The extended cyclic code has parameters (n+1, k, d+1).

4.4 BCH Codes

The most important class of cyclic codes, known as BCH codes, was discovered independently by Bose & Ray-Chaudhuri (1960a, 1960b), and Hoquenghem (1959). This class of codes has been studied extensively because of their relatively simple and powerful decoding algorithm. Berlekamp (1968) gives an excellent explanation of these codes. Also,

these codes have been studied by Gorenstein, et al (1960)(Generalisation of BCH codes), Mann (1962), Goldman et al (1968), Kasami and Tokura (1969), Chien (1972), Wolf (1969-70) and many other coding theorists.

A cyclic code of length n over GF(q) generated by the monic polynomial g(x), is a BCH code if for some integer b, g(x) is the lowest degree polynomial such that

$$g(x) = \text{Lcm}\left\{ M^{(b)}(x),\ M^{(b+1)}(x),\ \ldots.,\ M^{(b+d_o-2)}(x) \right\}$$

Therefore g(x) has a string of $d_o - 1$ consecutive powers of α as the zeros, and $M^i(x)$ are the minimum polynomials of the elements $\alpha^i \in GF(q^m)$. These zeros are also the zeros of the code, i.e. if c(x) is a code word:

$$c(\alpha^b) = c(\alpha^{b+1}) = \ldots., c(\alpha^{b+d_o-2}) = 0$$

Thus, the parity check matrix of the code can be formed as:

$$[H] = \begin{bmatrix} 1 & \alpha^b & \alpha^{2b} & \ldots\ldots & \alpha^{(n-1)b} \\ 1 & \alpha^{b+1} & \alpha^{2(b+1)} & \ldots\ldots & \alpha^{(n-1)b+1} \\ & & & & \\ 1 & \alpha^{b+d_o-2} & \alpha^{2(b+d_o-2)} & \ldots\ldots & \alpha^{(n-1)(b+d_o-2)} \end{bmatrix}$$

The minimum distance d of this code is bounded by

$$d \geq d_o$$

(BCH bound, theorem 8, MacWilliams and Sloane 1977), where d_o is called the designed distance of the code. Every element of the matrix [H] is an m-tuple of GF(q), so there are $m(d_o-1)$ rows in [H] and the dimension of the code is such that $k \geq n - m(d_o - 1)$.

For the binary case, if b = 1 and $n = q^m-1$ (primitive BCH code) then:

$$M^{(i)}(x) = M^{2i}(x) = \ldots\ M^{[(q^m-1)i]}(x),$$

thus $$g(x) = \mathrm{Lcm}\left\{M^{(1)}(x),\ M^{(3)}(x),\ \ldots\ ,\ M^{(d_o-1)}(x)\right\}$$

and the check matrix $[H]$ simplifies to:

$$[H] = \begin{bmatrix} 1 & \alpha & \alpha^2 & \ldots\ldots & \alpha^{n-1} \\ 1 & \alpha^3 & \alpha^6 & \ldots\ldots & \alpha^{3(n-1)} \\ & & & & \\ 1 & \alpha^{d_o-1} & \alpha^{2(d_o-1)} & \ldots\ldots & \alpha^{(d_o-1)(n-1)} \end{bmatrix}$$

since $d_o = 2t+1$, then $[H]$ has mt rows and the dimension of the code is $k \geqslant n - mt$.

It was suggested by Kasami & Lin & Peterson (1966) that the minimum distance of a primitive BCH code was equal to the designed distance of the code, but later, Kasami & Tokura (1969) showed the existence of primitive BCH codes with $d > d_o$ (see also Berlekamp 1970). An extension of a BCH code can be formed (Wolf 1969) by adding two columns to the $|H|$ matrix of the BCH (n, k, d) to give a new code of parameter ($n' = n+2$, $k' = k+2$, $d' = d$) and parity check matrix

$$[H_o] = \begin{bmatrix} 1 & 0 & & \\ 0 & 0 & & \\ 0 & \vdots & & \\ \vdots & \vdots & [H] & \\ & 0 & & \\ 0 & 1 & & \end{bmatrix}$$

4.5 Reed-Solomon Codes

Reed-Solomon codes are an important subclass of BCH codes (Reed Solomon 1960, Gorenstein & Zierler 1961). These codes are optimum in the sense that they have greater minimum distance in comparison with other linear codes having the same length and number of information digits. These codes have also been used extensively to

construct other powerful codes (Forney 1966, Justesen 1972).

A Reed-Solomon code is a BCH code over GF(q) of length $n = q - 1$; the roots of the polynomial x^n-1 are the elements of the $GF(q^m)$, where m is the smallest integer which n divides q^m-1, but $n = q - 1$, therefore $m = 1$ and the minimum polynomial of each element α^i is $M^{(i)}(x) = (x - \alpha^i)$, (theorems 6.24 and 6.17 Peterson & Weldon; also MacWilliams & Sloan, Chapter 10, 1978). Thus, the Reed-Solomon code of designed distance d has generator polynomial

$$g(x) = (x - \alpha)(x - \alpha^2) \ldots (x - \alpha^{d-1})$$

$$\alpha \in GF(q).$$

The dimension of the code K = n-degree $g(x) = n - (d-1)$, hence $d = n - k + 1$. Now, the minimum distance of a BCH (and hence RS) code is at least equal to the designed distance (BCH bound). On the other hand, the minimum distance of any (n, k) linear code is subject to the relation $d \leqslant n - k + 1$, (see Van Lint 1973, p. 71). Therefore, the minimum distance of a Reed-Solomon code is exactly

$$d = n - k + 1,$$

which is the minimum distance of a *maximum distance separable* code, and is therefore optimum. Also, the weight distribution of these codes is easily enumerated (see Peterson & Weldon 1972).

4.6 Iterated Codes and Concatenated Codes

Most known classes of linear codes reviewed so far do not have a positive asymptotic error correctability. In other words, for a fixed rate, the ratio of minimum distance to length of these codes is not lower bounded away from zero, i.e.,

$$d/n \longrightarrow 0 \quad \text{if} \quad n \longrightarrow \infty .$$

Long BCH codes are bounded by this property (theorem 13, Ch. 9, MacWilliams & Sloane 1978). However, certain classes of linear codes constructed from combinations of other linear codes have been shown to be

powerful linear codes with a positive asymptotic correctability; that is, these classes of codes, for a fixed rate, have a ratio of minimum distance to length bounded away from zero.

4.6.1 *Iterated Codes*

In this case, the output of the binary information source is arranged into a $k_2 k_1$ rectangular array. The rows of this matrix are encoded using a linear block (n_1, k_1, d_1) code, and the columns are encoded using a linear block (n_2, k_2, d_2) code (Figure 4-2).

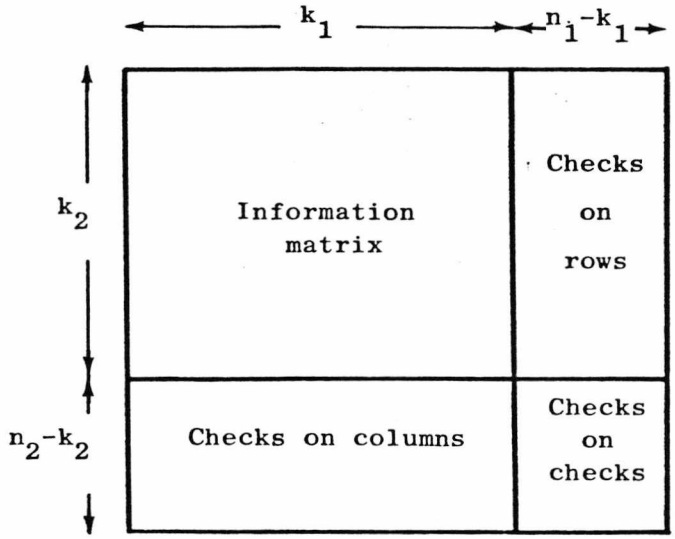


Figure (4-2) : General structure of iterated codes

The iteration process produces an $(n_1 n_2, k_1 k_2)$ linear code of Hamming distance $d_1 d_2$, (Elias 1954, see also Theorem 5.3 Peterson & Weldon). The resultant $(n_1 n_2, k_1 k_2, d_1 d_2)$ code can be iterated with another (n_3, k_3, d_3) code to produce an (n, k, d) code with parameters

$$\begin{cases} n = n_1 \cdot n_2 \cdot n_3 \\ k = k_1 \cdot k_2 \cdot k_3 \\ d = d_1 \cdot d_2 \cdot d_3 \\ R = k/n = k_1 \cdot k_2 \cdot k_3/n_i \cdot n_2 \cdot n_3 \end{cases}$$

The iteration of Hamming codes in the same manner produces Elias codes, (Elias 1954). In spite of decreasing the rate of the code by increasing the number of iterations, the rate of infinitely long Elias codes remains a positive value $R > 0$, (see Peterson & Weldon 1972, p. 135) such that

$$R > 1 - (m + 2)/2^{m-1}$$

provided $2^{m+1} \cdot p < 1$ and $m > 4$, where m is the number of parity checks in an extended Hamming code, and p is the BSC cross-over probability.

4.6.2 *Concatenated Codes*

The construction of these codes was first introduced by Forney (1966). They are defined as follows. The $k_2.k_1$ binary information symbols from GF(2) are divided into k_2 vectors of k_1 symbols. Each of these k_2 vectors is considered as a symbol in $GF(2^{k_1})$, and can be encoded by an outer encoder using an (N_2, k_2, d_2) Reed-Solomon code over $GF(2^{k_1})$, where any code word can be written as:

$$c_o\ c_1\ c_2\ \cdots\cdots\ c_{N_2-1} \qquad c_i \in GF(2^{k_1}) .$$

Each c_i is a k_1-tuple binary vector and is encoded by an inner encoder into an (N_1, k_1, d_1) code over GF(2). The resultant code has parameters (N_1N_2, k_1k_2) and the minimum distance is $d \geqslant d_1d_2$.

A simplified block diagram of a concatenated scheme encoder and decoder can be shown as Figure (4-3). Every uncorrected error pattern at the output of the inner decoder will appear as a single error at the

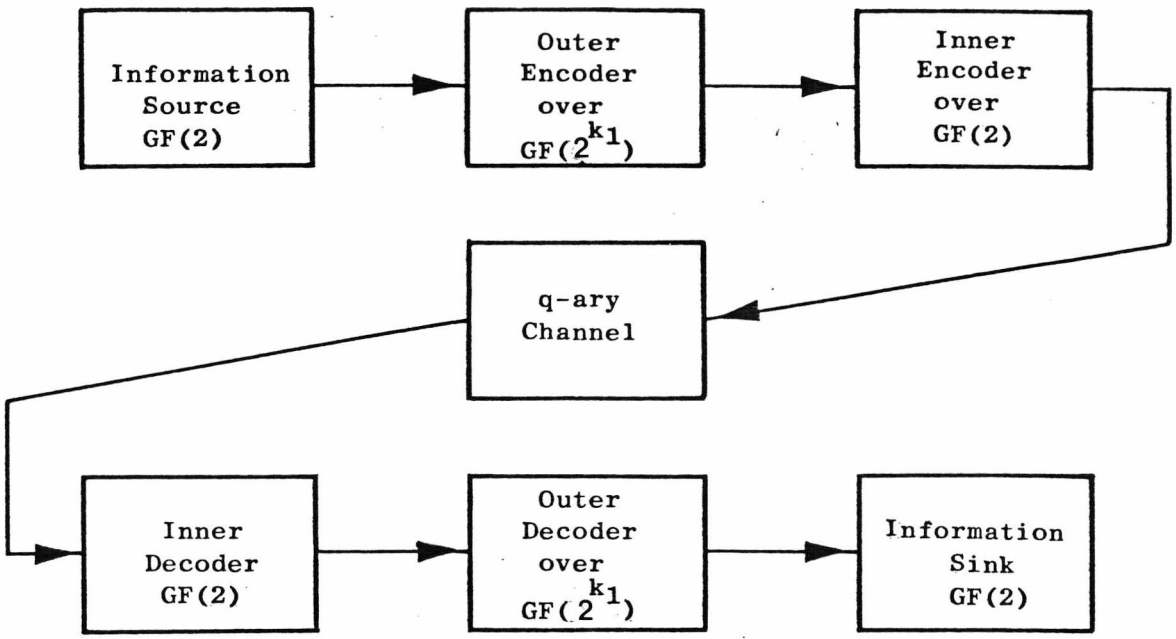


Figure (4-3) : Concatenated Coding Scheme

input of the outer decoder, since the outer decoder is a t-error-correcting Reed-Solomon code. Therefore all t-error patterns of length not bigger than N_1 are correctable.

4.6.3 *Justesen Codes*

Justesen (1972) presented a constructive sequence of codes, based on the concatenation concept of Forney's codes. The inner code, instead of being one code, is an ensemble of N_2 different (N_1, k_1) codes, such that each symbol of the outer code is encoded by a distinct inner code. The N_2 different inner codes are the randomly shifted codes of Wonzencraft, described by Massey (1963); (see also Cooper III, 1978). For any rate $0 < R < 1$, the ratio of distance to length of Justesen codes is lower bounded as

$$d/n > (1 - R/r)\, H^{-1}(1 - r) > 0,$$

where r is maximum at 0.5 and the solution to the rate

$$R = r^2/[1 + \log_2(1 - H^{-1}(1 - R))]$$

In other words the ratio of minimum distance to block length approaches a non-zero limit as the block length of the code increases. This limit is typically at R = 1/2 about 20 percent of the Varshamov Gilbert bound on the ahcievable distance. A simple example of a (12, 4, 4) Justesen code has been given in McWilliams & Sloane (1975, p.72).

4.7 Other Codes

There are many other important types of codes. The linear Goppa codes (Goppa, 1970 & 1971)) in general are non cyclic, but they include the class of primitive BCH codes. An important feature of Goppa codes is the fact that some of these codes satisfy the Varshamov-Gilbert bound on the ratio of minimum distance to block length. Also, these codes can be decoded with reasonable complexity (see Berlekamp 1973).

Furthermore, the Srivastava codes (see Berlekamp 1968, and Helgert 1967) and also the binary non-linear code of Nordstrom & Robinson (1967), which in a shortened version gives the two non-linear codes of Green (1966) and Nadler (1962), are optimum in the sense that they contain twice as many code words as the best linear codes of the same length and distance.

The array codes described by Smith (1978) and Farrell, et al (1979) which are generalizations of simple two-dimensional parity check codes to more than two dimensions, and to more complex parity checking arrangement, have interesting properties enabling the correction of the random and bursts of errors. These codes are efficient and may be easily decoded.

Also, a survey of coding theory by Wolf (1973), the introduction by Longo (1977) to algebraic coding theory, and the survey of con-

structive coding theory by Sloane (1972) may be referred to.

Lastly, but not least, in spite of all the attempts made by coding theorists, the problem of finding a family of codes which for any rate $0 < R < 1$ and block length n meets the Varshamov-Gilbert bound is unsolved. There are some existing families of codes, reviewed above, which are reasonably good for intermediate values of length, and also, fortunately, there exist a few families which though they do not meet the bound given by Gilbert, are asymptotically good, i.e. they form an infinite class of codes with both k/n and d/n bounded away from zero.

CHAPTER V

SEQUENCE CODES AND A RELATED BOUND

5.1 Simplex Codes (Maximum Length Codes, Pseudo-Noise Sequences, Uniform Codes)

The maximal-length codes or m-sequence codes are certain sequences of length $n = q^k - 1$, generated by linear sequential machines (feedback shift registers), whose feedback characteristic polynomials p(x) over GF(q) are primitive (McWilliams & Sloane 1978).

To construct an m-sequence code of length $n = q^k - 1$, the primitive polynomial

$$p(x) = x^k + c_{k-1}\, x^{k-1} + c_{k-2}\, x^{k-2} + \ldots + c_1 x + c_o$$

where $c_i \in GF(q)$

is chosen as the feedback characteristic polynomial of the k-bit shift register with scalar coefficient multipliers related to each $c_i \in GF(q)$, as shown in Figure (5 - 1).

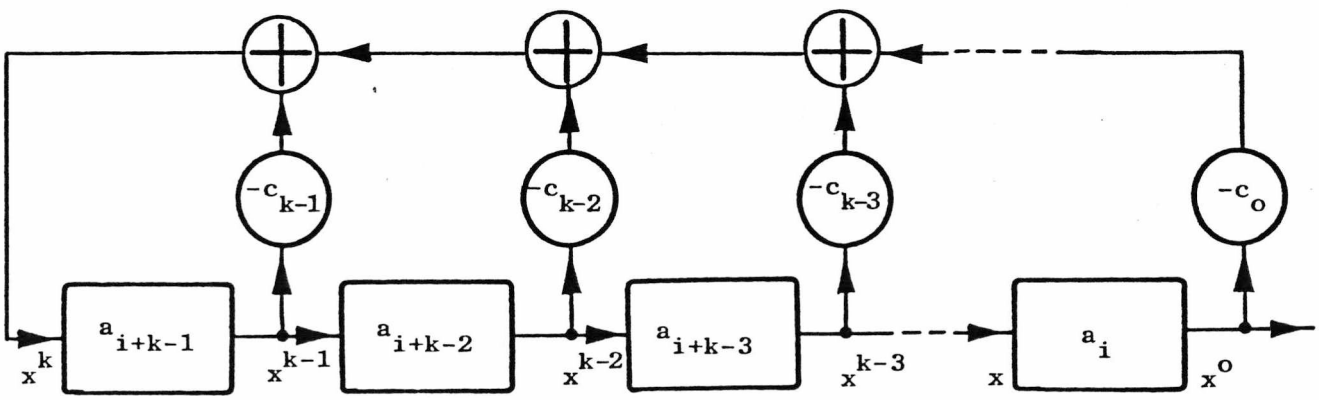


Figure (5-1) : A general sequential machine

For an initial state $(a_o, a_1, \ldots, a_{k-1}, a_k)$, this feedback shift register generates an infinite sequence in the form

$$a_o, a_1, a_2, \ldots a_i, \ldots \longrightarrow \infty$$

such that the elements inside a window of width $n_o = k$ digits slide

along the sequence are governed by a recursion rule of the form:

$$a_{i+k} = - c_{k-1}\, a_{i+k-1} - c_{k-2}\, a_{i+k-2} - \cdots\cdots c_1\, a_{i+1} - a_i c_o$$

where $i = 1, 2, \ldots$

There are q^k possible different states for a k-stage shift register, therefore the output sequence has a maximum periodicity of $p = q^k - 1$ (the all zero state is excluded), and consequently there are $q^k - 1$ different m-sequences, each starting from one of the $q^k - 1$ distinct states of the shift register. As an example, consider the primitive feedback polynomial $p(x) = x^3 + x + 1$ over GF(2) as realised in the SR circuit in Figure (5 - 2). The output sequence of the 3-bit shift register satisfies the recursion rule,

$$a_{i+3} = - a_{i+1} - a_i = a_{i+1} + a_i \pmod 2$$

If the initial state is $(a_2, a_1, a_o) = (1, o, o)$, the successive states are,

1	0	0
0	1	0
1	0	1
1	1	0
1	1	1
0	1	1
0	0	1

The right hand column is the output m-sequence related to the (1, 0, 0) initial state with a period of $2^3 - 1 = 7$; the others are the cyclic shifts of this column.

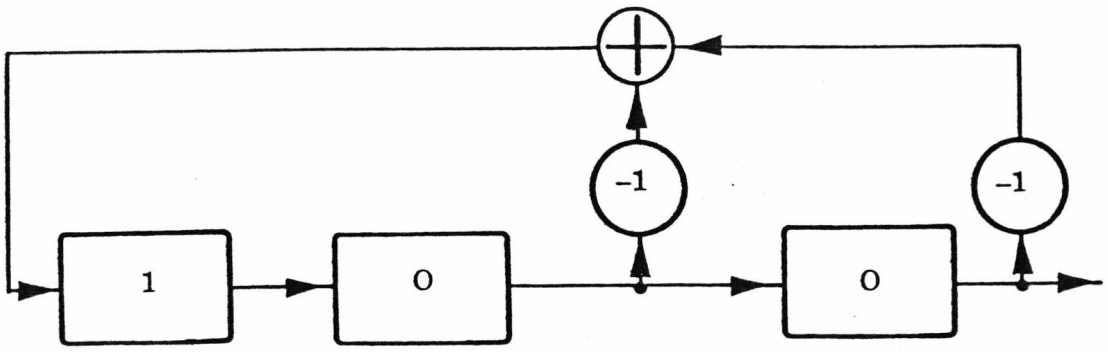


Figure (5-2) : Three stage maximum length sequence generator

The matrix [D] formed by this column as the first row and all the $2^k - 2$ other cyclic shifts together with the all-zero word, form the code book of a maximal length $(2^k - 1, k)$ code. Therefore the $2^k - 1$ non-zero words of this code have equal weight and the distance between any two code words is a constant

$$[D] = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$d = 2^{k-1}$. Thus the code is *uniform* with respect to its distance properties (equidistant). If the code words are fixed on the vertices of a unit n-dimensional cube, they form a regular simplex, so this code also is called a simplex code. Since the minimum and average distance of the m-sequence codes are equal, they meet the Plotkin bound and are optimum (Farrell, 1969), also the two-level autocorrelation function of the code words of these codes is an interesting and useful property when detecting (correlating) the code words when corrupted by noise (Viterbi 1966).

The m-sequence codes can be defined in the term of the generator polynomial $g(x) = (x^n - 1)/p(x)$ where $p(x)$ is the feedback

characteristic polynomial of degree k over GF(q) and $n = q^k - 1$. Alternatively, an m-sequence $[q^k-1, k, (q-1)q^{k-1}]$ code is an ideal cyclic code generated by the polynomial $g(x) = (x^n-1)/p(x)$, where p(x) is a primitive polynomial of degree k and $n = q^k-1$. Moreover the dual of an m-sequence code is a Hamming single-error-correcting $(q^k-1, n-k. 3)$ code (see Peterson & Weldon 1972). Therefore the modular representation matrix of these codes is perfect, in other words there is no zero element in the modular representation of the codes, so the generator matrix of an m-sequence code has all column types.

5.2 The Griesmer Bound and M-Sequence Codes

The Griesmer bound (Griesmer 1960) is a lower bound on the block length n of a linear binary block (k, d_o) code, given by:

$$n \geq \sum_{i=o}^{k-1} \left[(d_o + 2^i - 1)/2^i \right]$$

The derivation of this bound is based upon the possibility of a linear block (n, k, d_o) being repeatedly partitioned into smaller dimension linear block codes.

Consider the generator matrix [G] of a linear block (n, k, d_o) code. Since at least one of the code words has weight d_o, the generator matrix [G] of this code, by premultiplying on the left with a non singular matrix, can be put into such a form that this code word is the k^{th} row of [G]. Postmultiplying the resulting [G] matrix on the right by a suitable permutation matrix, the columns of [G] are rearranged in such a way that the last d_o elements of the k^{th} row are all non-zero, as follows:

$$[G] = \left[\begin{array}{c|c} G_0 & G_1 \\ \hline 0 & 1 \end{array}\right] \Longrightarrow \begin{array}{|c|c|} \hline Q_0 & Q_1 \\ \hline Q_0 & I_1 \\ \hline \end{array}$$

where Q_0 and Q_1 are the row spaces of the $[G_0]$ and $[G_1]$ matrices. Therefore the row space of the row vectors of the $[G_1]$ matrix, including the all ones k^{th} row, is the row space of $[G_1]$ and its logical complement I_1. On the other hand, the row space of the $[G_0]$ matrix including the k^{th} all zero row is the row space of $[G_0]$ and its exact copy Q_0. Thus the row space of $[G_0]$ is an $(n - d_0, k-1, d_1)$ linear binary block code. If $\bar{x}$ is a code word in the $[Q_0 \mid Q_1]$ part and if the elements positioned in Q_1 have weight c, then

$$d_1 + c \geq d_0,$$

but the related code word of $\bar{x}$ in the $[Q_0 \mid I_1]$ part has weight

$$d_1 + \bar{c} \geq d_0$$

or

$$d_1 + d_0 - c \geq d_0$$

$$\text{therefore} \quad 2d_1 \geq d_0 \quad \text{or} \quad d_1 \geq \frac{d_0}{2} \ .$$

Since d_1 is an integer, thus

$$d_1 \geq \left[\frac{d_0}{2}\right] ,$$

where $[x]$ means the greatest integer less than or equal to x. The Griesmer bound is obtained by repeated application of this result, considering that the minimum distance of an (n, 1) code is d = n; this process is schematically given by Figure (5-3):

$$n \geq \sum_{i=1}^{k-1} \left[\left(d_o + 2^i - 1 \right) / 2^i \right]$$

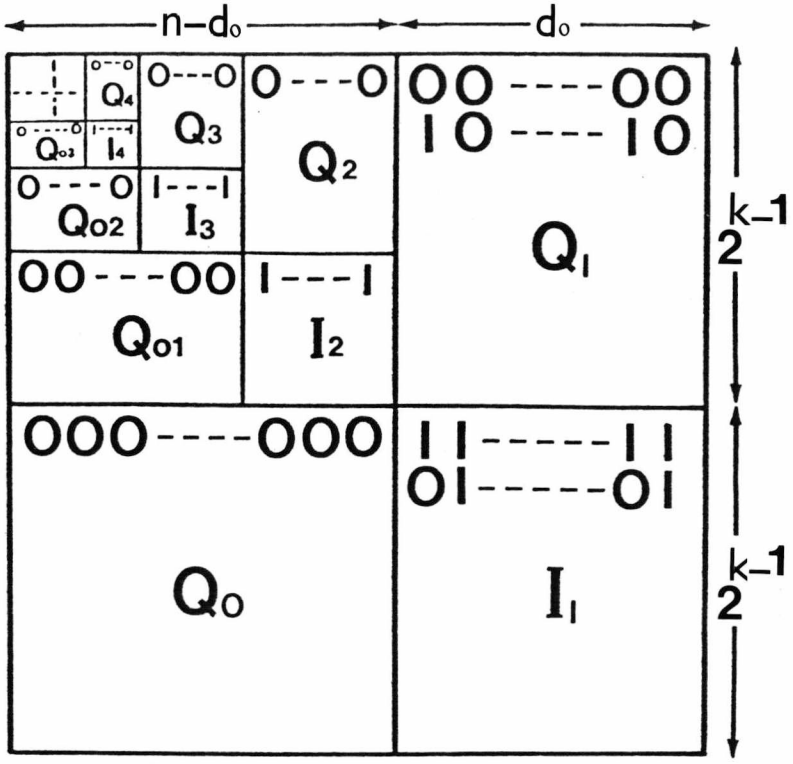


Figure (5-3) : Griesmer lower bound and a repeatedly partitioned linear binary block code.

The Griesmer bound is tight when d is large with respect to k. Therefore, the combination of the Griesmer bound in the region where d > k (i.e. where the bound is tight) with the Hamming bound in the region where k > d (the region where this bound is good) results overall in a better lower bound (Griesmer 1960).

Solomon and Stiffler (1965) generalised the Griesmer bound over an arbitrary finite field GF(q) to give

$$n \geq \sum_{i=o}^{k-1} \left[(d_o + q^i - 1)/q^i \right]$$

This states that the value of n for fixed values of k and d of a linear block code over GF(q) is lower bounded by the above inequality. The proof of this is parallel to the binary code, though when considering the possibility of a linear q-ary code being partitioned repeatedly into smaller dimension linear q-ary code, the argument is slightly different from the binary case. The generator matrix of a linear code over GF(q)

can be partitioned in a similar manner to that of the binary case, and has the following form:

$$[G] = \left[\begin{array}{c|c|c|c|c|c|c} G_0 & G_1 & G_2 & --- & G_i & ---- & G_{q-1} \\ \hline 0 & \alpha_0 & \alpha_1 & --- & \alpha_i & ---- & \alpha_{q-2} \end{array}\right]$$

where $\alpha_i \in GF(q)$ and $[\alpha_i]$ is a row vector of which all elements are α_i. The $[G_o]$ matrix including the k^{th} all $(n-d_o)$ zeros row spans the row space of a linear $(n - d_o, k - 1, d_1)$ code over GF(q) and $(q - 1)$ copies of it as it is shown in Figure (5-4). The row space of the individual matrices $\left[\begin{array}{c} G_i \\ \hline \alpha_i \end{array}\right]$ is seen to be the vectors formed by the addition of $c\alpha_i$ where $c \in GF(q)$, to the vectors of the row space of $[G_i]$ thereby forming (q-1) translates. This can be formed by writing the row space of $[G_i]$, consisting of q^{k-1} vectors, the next q^{k-1} elements of total row space formed by adding α_i to those above, and this process then repeated (q-1) times.

If $\overline{X}$ is a code vector with at least d_1 of its first $n - d_o$ elements non-zero, and d_2 of the last d_o element also non-zero, then

$$d_1 + d_2 \geq d_o .$$

There are (q-1) vectors, all of which begin with $n - d_o$ all-zero elements (see the left-most side of Figure (5-4); each vector is a multiple of the other vector; i.e.:

$$1 \quad \overline{x} = (0, 0, \ldots. 0, x_1, x_2, x_3 \ldots x_{d_o})$$

$$\alpha_1 \quad \overline{x} = (0, 0, \ldots. 0, \alpha_1 x_1, \alpha_1 x_2, \ldots \alpha_1 x_{d_o})$$

$$\alpha_{q-2}\, \overline{x} = (0, 0, \ldots. 0, \alpha_{q-2}\, x_1, \ldots. \alpha_{q-2}\, x_{d_o})$$

These are d_2 non-zero elements in the previous d_o elements of $\overline{X}$; each of these d_2 elements agrees with one of the elements of each corresponding column of $1\overline{x}$, $\alpha_1\, \overline{x}$, ... $\alpha_{q-2}\, \overline{x}$, since each element of GF(q)

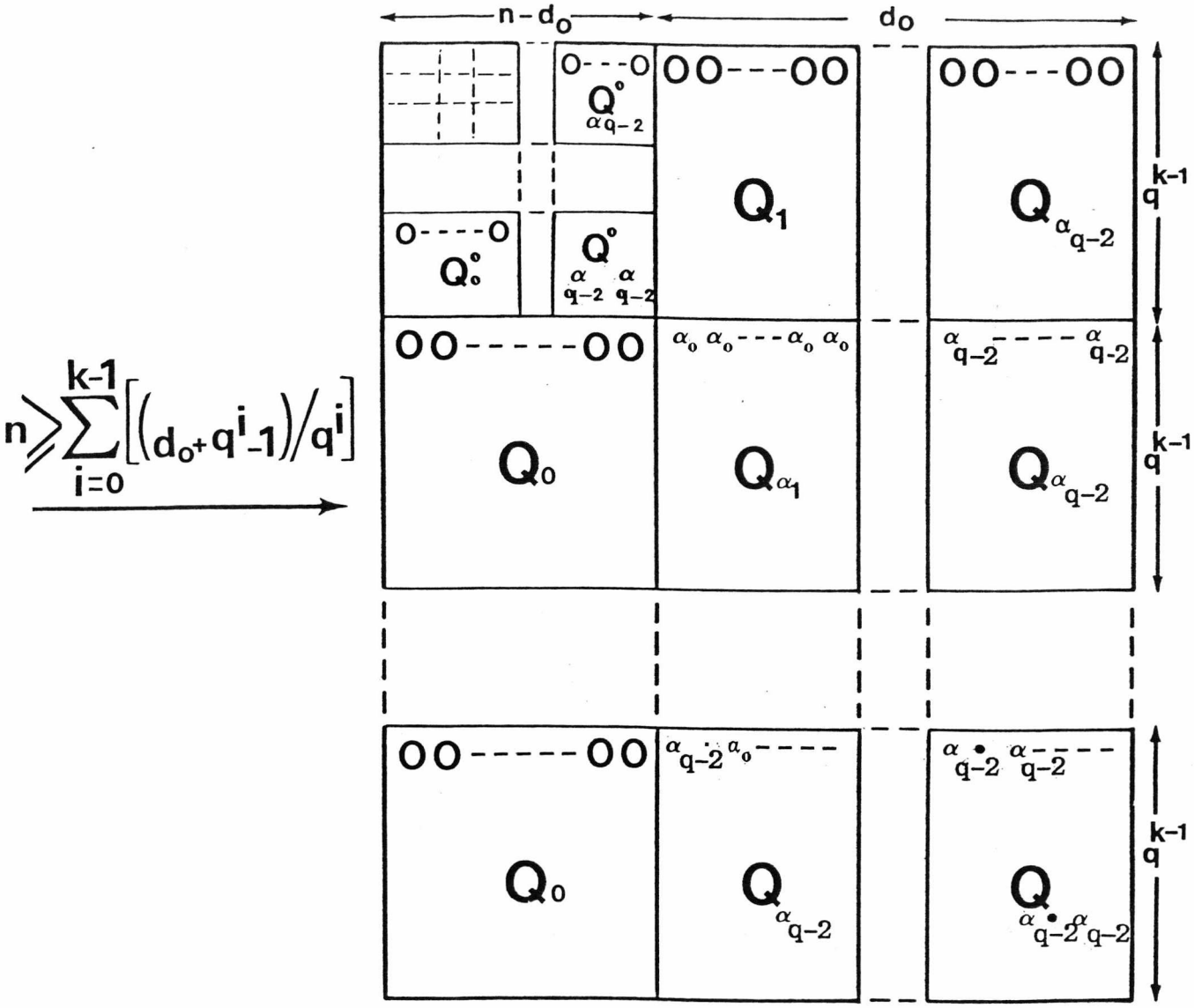


Figure (5-4) : Generalised Griesmer bound by Solomon & Stiffler and a repeatedly partitioned linear q-ary block code.

occurs exactly once in each column. Therefore the average agreements per code word $\overline{X}$ is r

$$r \geq d_2/(q-1) .$$

If $\overline{X}$ is subtracted from any of the code words $\overline{x}$, $\alpha_1 \overline{x}$, .. $\alpha_{q-2} \overline{x}$, there will be d_1 non-zero elements in the first $n - d_o$ elements, and $d_o - r$ non-zero elements in last d_o positions. Then:

$$d_1 + d_o - r \geq d_o$$

or

$$d_1 \geq r \geq d_2/(q-1)$$

we had $d_1 + d_2 \geq d_o$

$$\text{therefore} \quad d_1 \geq \frac{d_o}{q}$$

by the repeated application of this result we obtain:

$$n \geq \sum_{i=o}^{k-1} \left[(d_o + q^i - 1)/q^i \right] .$$

This bound is achieved by the class of linear (n, k, d) codes over GF(q) which have in general the following parameters:

$$\begin{cases} n = \beta_o (q^k - 1)/(q-1) - \sum_i \beta_i \, (q^{\ell_i} - 1)/(q-1) \\ \\ d = \beta_o \, q^{k-1} - \sum_i \beta_i \, q^{\ell_i - 1} \end{cases}$$

where $\beta_i = (1, 2, \ldots, q-1)$ and $\beta_o \geq \max \beta_i$ and also

$$\begin{cases} \sum_i \ell_i \leq k \\ \\ 1 \leq \ell_i < k. \end{cases}$$

Codes with these parameters may be obtained by deleting (puncturing) certain columns of one or more copies of a linear $[n = q^k - 1, \; k, \; d = q^{k-1}(q-1)]$ maximal length code. The columns of a linear maximal length code over GF(q) from a linear algebra (Ayres 1965, p. 219) generated by any k linearly independent columns (also this linear algebra is a vector space with respect to the scalar multiplication over GF(q). Therefore the columns of the maximal length code can be paritioned into the (q-1) distinct cosets (classes) such that if x is a column in a coset no scalar multiple of x is in the same coset. Any coset contains $(q^k - 1)/(q-1)$ columns and any row has weight q^{k-1}. If $\beta \leq (q-1)$ cosets are deleted from the original

maximal length code a linear

$$\left[n = (q-1-\beta)(q^k-1)/(q-1),\ k,\ d = (q-1-\beta)q^{k-1}\right]$$

code over GF(q) will result. Alternatively, any sub-algebra (sub-additive group, subspace) of dimension ℓ_i of the k dimensional algebra (vector space) can be partitioned into (q-1) cosets and β_i of these cosets in a similar way may be deleted, which results in

$$\left[n = \beta_o(q^k-1)/(q-1) - \sum_i \beta_i(q^{\ell_i}-1)/(q-1),\ k,\ d = \beta_o(q^{k-1}) - \sum_i \beta_i(q^{\ell_i-1})\right]$$

linear block code, where $\beta_i = (1, 2, \ldots. q-1)$, and $\beta_o \geq \max \beta_i$, $\sum_i \ell_i \leq k$, $1 \leq \ell_i \leq k-1$, $\ell_i \neq \ell_j$ $(i \neq j)$.

If $\beta_o = (q-1)$ and $\beta_i = (q-1)$, then a linear

$$\left[n = q^k-1 - \sum_i q^{\ell_i}-1,\ k,\ d = (q-1)(q^{k-1} - \sum_i q^{\ell_i-1})\right]$$ non-repeated

column code over GF(q) results. For the same conditions, and where q = 2, which is the binary case, the resulting code is a

$$\left[(2^k-1) - \sum_i 2^{\ell_i}-1,\ k,\ 2^{k-1} - \sum_i 2^{\ell_i-1}\right]$$

linear binary code. If $\beta_o > (q-1)$ then a linear repeated column code is obtained.

The condition $\sum_i \ell_i \leq k$ provides a restriction on the resulting code parameters n, d, such that for any set of ℓ_i

$$n \geq 2^{k-1} \quad \text{and} \quad d > 2^{k-2}-1.$$

Therefore these codes are of low rate. Baumert and McEliece (1973) have shown that this condition on a repeated column code being such that

$\sum_i \ell_i \leq r\,k$, where r is the number of repetitions of the maximal length code, is a very weak sufficient condition on the dimension of the subspaces ℓ_i . McEliece showed that even for some value of $\sum_i \ell_i \geq r\,k$ it is possible to find proper subspaces with the property that these subspaces cover each column at most r times. However, the generator matrix of a linear maximal length code has all possible column types, therefore any linear block code is equivalent to a punctured maximal length code. The following chapters describe the development of a technique called the Anticode method which enables good choices of the columns to be deleted from a maximal length code for any $m \leq q^{k}-1-k$.

CHAPTER VI

LINEAR ANTICODE ANALYSIS AND CONSTRUCTION

6.1 Linear Sequence Codes and the Deletion Concept

An (n, k, d) linear code over GF(q), containing q^k code words is a subspace V_c of dimension k of the vector space V_n of all n-tuple vectors over GF(q) (see Section 3.1). Therefore a linear code can be defined (identified) with any k linearly independent code words, which form a k × n matrix known as the generator matrix $[G]$ of the code. The columns of $[G]$ are a subset of the set of all k-tuple vectors over GF(q). The generator matrix $[G]_{ML}$ of a linear maximal length $(q^k-1, k, (q-1)q^{k-1})$ code over GF(q), consists of all distinct non-zero k-tuple vectors arranged in some order; alternatively, the columns of the matrix $[G]_{ML}$ and the rowspace (the code book of the maximal length code) of this matrix form all distinct possible column types of any linear code. Thus the n columns of the generator matrix of any linear code is a subset of the columns of the generator matrix of a linear maximal length code in some order, and so for any linear (n, k, d) code over GF(q) there is a set of m complementary distinct columns, such that the concatenation of these m with n columns of the code results in columns which are the $q^k-1 = n_o$ distinct columns of a maximal length code in some order. Alternatively, deleting any set of $(q^k-k-1) \geq m \geq 0$ columns, having a non-zero row of weight at most δ from a maximal length code called the parent maximal length code results in a linear $[q^k-1-m, k, (q-1)q^{k-1} - \delta]$ non-repeated column code. Griesmer (1960), using this idea, discovered four classes of binary linear codes of dimension k and maximum distance

$$d = h2^{k-1} + 2^{k-2} + \ldots\ldots + 2^{k-p} + l$$

where

$$\begin{cases} h \geq 0 \\ l = -1, 0, 1, 2 \\ 2 \leq P \leq k-1 \end{cases}$$

which have the minimum value of block length predicted by him as a lower bound. Therefore these codes are optimum, and are also included in the Solomon and Stiffler codes, as reviewed in Section 5.2.

However, the Solomon and Stiffler optimum codes, due to their suggested construction, have a lower bounded block length and minimum distance. For a fixed k the block length n and minimum distance d of the Solomon and Stiffler linear block codes derived from the maximal length codes are bounded by

$$\begin{cases} n \geq q^{k-1} - 1 \\ d \geq q^{k-2}(q-1) - 1. \end{cases}$$

Therefore the block length n of these codes increases much faster (exponentially) than k, so these codes turn out to be low rate codes, in other words $k/n \longrightarrow 0$ when $n \longrightarrow \infty$.

Farrell (1969) introduced an infinite class of subsets of the column set of a maximal length code, which are known as anticodes, and result in an infinite class of linear optimum or near optimum codes with parameters

$$\begin{cases} q^k - 1 \geq n \geq k \\ k \\ q^{k-1}(q-1) \geq d \geq 1 \end{cases}$$

derived with the same deletion concept as Solomon and Stiffler. The maximal length code from which the columns of an anticode are deleted is called the parent anticode. Anticodes have been studied by Farrell (1970) and also by Farrell and Farrag (1974) and Farrag (1976). The latter, by an exhaustive computer search, synthesised all possible

optimum anticodes for $k \leq 6$ and some anticodes of $k = 7$. However, the computation time for this exhaustive search technique for $k > 6$ was rather excessive.

6.2 The Linear Binary Parent Anticodes

The generator matrix $[G]_{PAC} = [M]$ of a linear binary non-repeated column parent anticode has all the (2^k-1) possible k-tuple column types, in natural order of binary numbers, increasing from left to right, i.e.,

$$[G]_{PAC} = [M] = [(1)_2, (2)_2, \ldots, (i)_2, \ldots (2^k-1)_2]$$

where $(i)_2$ is a column of k binary digits which is the binary representation of the number i. As an example, for $k = 3$, the matrix is:

$$[G]_{PAC} = [M] = [(1)_2, (2)_2, (3)_2, \ldots (7)_2]$$

$$= \begin{bmatrix} 1\,0\,1\,0\,1\,0\,1 \\ 0\,1\,1\,0\,0\,1\,1 \\ 0\,0\,0\,1\,1\,1\,1 \end{bmatrix}$$

The row space of matrix $[M]$ forms the code book of the linear binary parent anticode.

6.2.1 *The Iterative Property of the Parent Anticode*

The parent anticode $[C]$ can be formed by spanning the row space of the matrix $[M]$, by multiplying on the left with the matrix $[M_o]^T$, where the matrix $[M_o]^T$ is the transpose of the matrix $[M]$ including the all zero k-tuple row. The matrix $[M_o]^T$ has all the 2^k rows of the k variable truth table (Figure (6-2)). Therefore the matrix $[C] = [M_o]^T[M] = [C_{ij}]$ is the 2^k linear combination of the rows of the matrix $[M]$ and is the code book of the parent anticode. As an example consider the matrices $[M]$ and $[M_o]^T$ for $k = 4$ which are shown below:

$$[M] = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Figure (6-1) : Parent Anticode Generator Matrix

$$[M_o]^T = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix} \begin{matrix} \\ \\ \\ = [M_o]^T_{(1)} \\ \\ \\ \\ = [M_o]^T_{(2)} \\ \\ \\ \\ = [M_o]^T_{(3)} \\ \\ \\ \\ \end{matrix}$$

Figure (6-2) : Partitioned Version of $[M_o]^T$

Continuing the above example, the matrix $[M_o]^T$ spans (linearly combines) the row space of the matrix $[M]$. In forming the product $[M_o]^T[M]$ the first $2^2 = k$ rows of $[M_o]^T$, denoted by $[M_o]^T_{(1)}$ in Figure (6-2) are responsible for the linear combination of only the first two rows of $[M]$ shown in Figure (6-1), thus forming the first four rows of the anticode book $[C]$ in Figure (6-5). The next four rows of $[M_o]^T$, denoted by $[M_o]^T_{(2)}$ combine the third row of $[M]$ with the previously formed rows of $[C]$. In forming the second four rows of $[C]$, those

elements of the first four rows of [C] which correspond to zero elements in the third row of [M] are copied into the next four rows without change, but those elements that correspond to a one element are inverted (logically complemented). The matrix $[M_o]^T$ has as its first three columns, the first three columns of the stacked $[M_o]^T_{(1)}$ and $[M_o]^T_{(2)}$, the fourth column being all ones. The last eight rows of [C] are formed by copying those elements of the first eight rows which correspond to zeros in the fourth row of [M], and inverting those that correspond to ones.

Therefore, the columns of the parent anticode have the property that, every 2^i (i = 1, 2, 3, ..., k) top elements of each column can be divided into two equal parts, each containing 2^{i-1} elements, for which either the top and bottom halves are identical, or the bottom half is the logical complement of the top half. As an example the column of type eleven of the previous [C] matrix is shown in Figure (6-3).

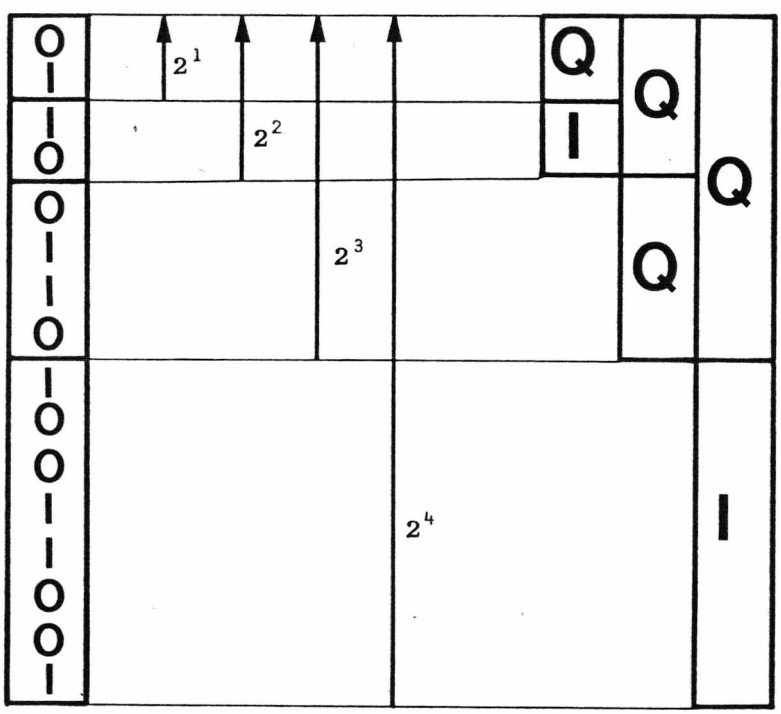


Figure (6-3) : Column of Type Eleven in Q-I Systematic Form.

Since the matrix $[M_o]$ is the same as the matrix $[M]$, but includes the all-zero k-tuple, therefore $[M_o]^T[M_o]$ is identical to $[C_o]$, the parent anticode book including all zero 2^k tuple as the first column. Thus $[C_o]$ is a symmetric matrix and the rows have the same property as the columns.

This property is called the Q-I systematic property of the parent anticode (anticode). Due to this property the parent anticode $[C]$ can be partitioned into smaller dimension subspaces and their logical complements as shown in Figure (6-5), where the generator matrix of each subspace is the truncated elements of the generator matrix $[M]$ of the parent anticode, as in Figure (6-4).

$$[M] = \left[\begin{array}{ccc|c|ccc|c|ccc|c|ccc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ \hline 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right]$$

Figure (6-4) : The Truncated Version of the Generator Matrix of the Parent Anticode.

The row space of the truncated elements then are spanned due to the 3-tuple zeros or ones positioned under these elements. For any zero 3-tuples the row space of the truncated elements are copied, and for any one 3-tuples the row space of the truncated elements is inverted. The resultant row space will be repeatedly stacked again due to the rest of the 3-tuples.

The generator matrix $[M]$ of the parent anticode $[C]$ has one of each of the 2^k-1 different column types, hence the modular representation matrix $[N]_{PAC}$ of order $1 \times 2^k-1$ is

$$[N]_{PAC} = [n_1, n_2, \ldots, n_{2^k-1}] = [1, 1, 1, \ldots, 1, 1]$$

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0
1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	0
0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0
1	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0
1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0
0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	0
1	1	0	0	1	1	0	1	0	0	1	1	0	0	1	0
0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0
1	0	1	1	0	1	0	1	0	1	0	0	1	0	1	0
0	1	1	1	1	0	0	1	1	0	0	0	0	1	1	0
1	1	0	1	0	0	1	1	0	1	1	0	1	1	0	0

Figure (6-5) : Partitioned Version of the PAC(15, 4, 8).

Alternatively, the parent anticode words may be regarded as the vertices of a regular simplex. The parent anticode is a simplex code and shares the properties inherent in this family of codes, thus the weight distribution matrix $[W]_{PAC}$ of the order $2^k \times 1$ is

$$[\mathbf{W}]_{PAC} = [C]\,[N]^T_{PAC} = \begin{bmatrix} w_o \\ w_1 \\ \vdots \\ w_i \\ \vdots \\ w_{2^k} \end{bmatrix} = \begin{bmatrix} o \\ 2^{k-1} \\ \vdots \\ 2^{k-1} \\ \vdots \\ 2^{k-1} \end{bmatrix}$$

and it is said to be a uniform code.

If the rows of the generator matrix $[M]$ are labelled as $g_1, g_2, \ldots, g_i, \ldots g_k$, from the Q-I property of the code book of the parent anticode $[C]$ and the way that $[M_o]^T$ combines the rows of the matrix $[M]$, it is evident that the N^{th} parent anticode word results from the N^{th} row of the matrix $[M_o]^T$ which is the k-digit binary representation of integer $(N-1)_2$. Hence the N^{th} parent anticode word can be identified in terms of the linear combination of the rows $g_1, g_2, \ldots, g_i, \ldots, g_k$ of the generator matrix $[M]$ of the parent anticode as:

$$\begin{cases} (N-1)_2 = \sum_{i=1}^{k} r_i \, 2^{i-1} \\ [C]_{(N)} = \sum_{i=1}^{k} r_i \, g_i \end{cases}$$

where r_i is either zero or one, and the values of r_i for which there exists non-zero r_i indicates the g_i that contributes to the N^{th} parent anticode word. Also, the N^{th} parent anticode word is denoted by $[C]_{(N)}$. As an example, consider the identification of the 14^{th} parent anticode word,

$$(N-1)_2 = (14-1)_2 = \sum_{i=1}^{k} r_i \, 2^{i-1}$$

$$13 = r_1 2^0 + r_2 2^1 + r_3 2^2 + r_4 2^3$$

Therefore $r_1 = 1, \; r_2 = 0, \; r_3 = 1, \; r_4 = 1$

and

$$[C]_{(14)} = \sum_{i=1}^{k} r_i g_i = r_1 g_1 + r_3 g_3 + r_4 g_4$$

$$= g_1 + g_3 + g_4$$

Hence the 14^{th} parent anticode word is the linear combination of the 1st, 2nd and 4th rows of the generator matrix $[M]$ of the parent anticode.

6.3 Linear Binary Anticode Analysis

A linear binary anticode is a set of 2^k m-digit sequences (words) over the finite field GF(2). The m different columns of an anticode is a subset of the (2^k-1) distinct columns of the parent anticode. Therefore the set of the 2^k rows of the anticode (anticode book) are the elements of a subspace of the m-tuple vector space over GF(2); alternatively, they are the element of an abelian group.

Hence an anticode can be specified by its generator matrix $[G]_{AC}$, but an anticode may or may not have a repeated row which is inherent in the Q-I property of the parent anticode, and also depends on the parameters of the anticode. Therefore the generator matrix $[G]_{AC}$ of an anticode, in contrast to a linear code, may have all-zero rows; in other words, the generator matrix of an anticode in the canonical form may have $k_i = 0, 1, 2, \ldots$ all-zero element rows. The generator matrix of an anticode in row canonical form denoted by $[G]_{CAC}$ is shown below.

$$[G]_{AC} \simeq [G]_{CAC} = \begin{bmatrix} G_{BAC} & k_o \\ \hline \begin{matrix} 0 \\ \vdots \\ 0 \end{matrix} & k_i \end{bmatrix}$$

The row rank k_o of the generator matrix of an anticode is $k_o \leq k = k_o + k_i$, therefore the row space of the generator matrix of an anticode is an "iterated subspace" of the vector space V_m of all m-tuple vectors. As an example, consider the anticode consisting of the 3 columns of the parent anticode of Section 6.2.1. of types $\{4, 10, 14\}$ as follows:

0	0	0
0	0	0
0	1	1
0	1	1
1	0	1
1	0	1
1	1	0
1	1	0
0	1	1
0	1	1
0	0	0
0	0	0
1	1	0
1	1	0
1	0	1
1	0	1

The generator matrix of this anticode is:

$$[G]_{AC} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

and the row canonical form of $[G]_{AC}$ is

$$[G]_{CAC} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ \hline 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} G_{BAC} \\ \hline 0 \\ 0 \end{bmatrix} \begin{matrix} k_o = 2 \\ \\ k_i = 2 \end{matrix}$$

6.3.1 *The Linear Binary Basic Anticodes*

The first k_o rows of the generator matrix $[G]_{CAC}$ of an anticode in the row canonical form, which form the k_o rows of the matrix $[G]_{BAC}$, are linearly independent, thus the row space of this matrix is a sub-space of dimension k_o of the vector space V_m of all m-tuple vectors.

The anticode generated by $[G]_{BAC}$ has no repeated rows and is called a basic anticode. Continuing the above example:

$$[G]_{BAC} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \longrightarrow \begin{matrix} \text{basic} \\ \text{anticode} \end{matrix} \quad \begin{matrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{matrix} .$$

The original anticode generated by $[G]_{AC}$ is the iteration of the basic anticode generated by $[G]_{BAC}$, $N = 2^{k_i}$ times. In the above example the $2^2 = 4$ copies of the basic anticode have been iterated (interleaved) to form the original anticode. The integer $\psi = k_i$ is called the iteration degree, and it is said that the original anticode (iterated anticode) has degree $\psi_i = 2$ and that the basic anticode has iteration degree of zero. The number of rows of $[G]_{AC}$, $k = k_o + k_i = k_o + \psi$ is called the dimension of the anticode. A basic anticode whose generator matrix has all the different k-tuple column types, is called a maximal length anticode or simply an m-sequence anticode. It is evident that an m-sequence AC(m, k, δ, ψ) has parameters

$$\left[m = 2^k - 1,\ k,\ \delta = 2^{k-1},\ \psi = 0\right]$$

which are the parameters of those parent anticodes and maximal length codes with the same parameters, therefore these anticodes cannot be deleted from parent anticodes with the same parameters. However, these may be used as the units of construction in order to construct bigger dimension anticodes (see Section 6.4). In general any anticode that is used to construct a bigger dimension anticode is called a unit anticode, and it is prefixed by the related name when necessary, such as, the unit basic anticode.

6.3.2 *The Maximum Hamming Distance of an Anticode*

An anticode of dimension k and anticode length m contains 2^k anticode words. The maximum Hamming distance of an anticode, denoted by δ, is defined to be the maximum value of the Hamming distance between all pairs of distinct anticode words, i.e.:

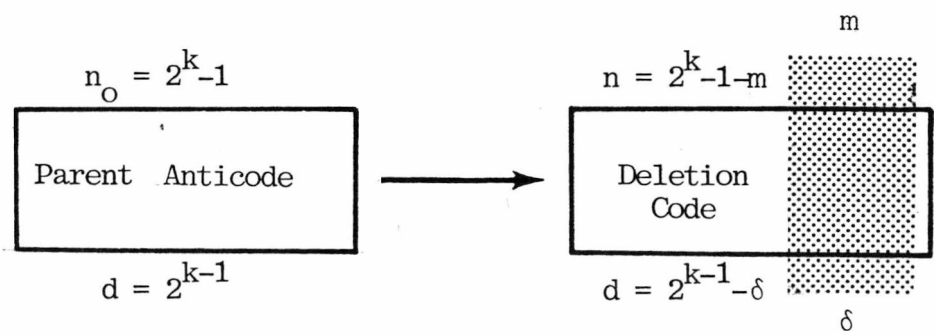
$$\delta = \max W(\bar{x}_i \theta_2 \bar{x}_j) \quad \text{for all } i \neq j$$

$$i, j = 1, 2, \ldots, 2^k$$

where $W(\bar{x})$ is the Hamming weight of $\bar{x}$. The rows of the anticode form a linear iterated subspace, thus the linear combination of any two anticode words is another one, so that:

$$\delta = \max W(\bar{x}_i) \qquad i = 1, 2, \ldots, 2^k$$

The integers m, k, δ, ψ, are called the anticode parameters denoted as AC(m, k, δ, ψ); when $\psi = 0$, it is simplified as AC(m, k, δ). The m columns of the AC(m, k, δ, ψ) are deleted from the parent anticode $(2^k-1, k, 2^{k-1})$ shown symbolically as:



The maximum distance δ and the word length m of AC(m, k, δ, ψ), determine the minimum distance d and the block length n of the resulting (n, k, d) code, called the deletion code, and denoted by DC(n, k, d), where

$$\begin{cases} n = 2^k - m - 1 \\ d = 2^{k-1} - \delta \end{cases} .$$

Thus for given anticode parameters m, k, ψ, the maximum distance δ

is required to be the smallest possible value, since this minimum value of δ is related to a maximum value of minimum distance d in the deletion code. An anticode which has the minimum possible value of δ for a given m, k, ψ, is called an optimum anticode; alternatively, an anticode is optimum if the related deletion code is optimum. However, the minimum Hamming distance d of an anticode is also an important parameters, and is defined to be the minimum value of the Hamming distance between all pairs of distinct anticode words, i.e.

$$d = \text{min. } W(\bar{x}_i \theta_2 \bar{x}_j) \quad \text{for all } i \neq j$$
$$i, j = 1, 2, \ldots, 2^k$$

or simply,

$$d = \text{min. } W(\bar{x}_i) \quad \text{for all } i = 1, \ldots, 2^k$$

6.3.3 *The Modular Representation and Weight Distribution of an Anticode.*

Any AC(m, k,δ,ψ) can be specified by the generator matrix $[G]_{AC}$ of the anticode; but also, given a possible permutation of its columns, by the modular representation matrix $[N]_{AC} = [n_1, n_2, \ldots, n_i, \ldots n_{2^k-1}]$, where n_i is the number of column type i as defined in Section 6.2 and

$$m = \sum_{i=1}^{2^k-1} n_i$$

Since the columns of the AC(m, k, δ) are deleted from the parent anticode, this is the same as deleting the columns of the generator matrix AC(m, k, δ) from the columns of the generator matrix of the present anticode. Thus the modular representation of an anticode $[N]_{AC}$ is the logical complement of the modular representation matrix $[N]_{DC}$ of the deletion code DC(n, k, d), such that

$$[N]_{AC} + [N]_{DC} = [N]_{PAC} \cdot$$

As an example, consider the AC(3, 2, 3), having the generator matrix

$$[G]_{AC} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \longrightarrow [G]_{DC} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Hence

$$[N]_{AC} = [1\ 0\ 0\ 1\ 1\ 0\ 0]$$

$$[N]_{DC} = [0\ 1\ 1\ 0\ 0\ 1\ 1]$$

$$[N]_{PAC} = [1\ 1\ 1\ 1\ 1\ 1\ 1]$$

A list of the weights of the 2^k anticode words of $[W]^T_{AC} = [w_o, w_1, \ldots w_i, \ldots, w_{2^k}]$ can be obtained by post multiplication of the parent anticode $[C]$ by the matrix $[N]^T_{AC}$, as

$$[W]_{AC} = [C][N]^T_{AC} \quad \text{where} \quad \begin{cases} w_i = \sum_{j=1}^{2^k} c_{ij}\, n_j \leqslant \delta \\ m = \sum_{i=1}^{2^k-1} n_i \end{cases}$$

From the above equations it is evident that for a given list of weights, an anticode is optimum if the $[N]^T_{AC}$ has the maximum number of elements.

6.3.4 *The Weight Enumerator Polynomial of an Anticode and the Corresponding Deletion Code.*

If the number of anticode words of weight i is denoted by A_i then the polynomial

$$W_{AC}(x) = \sum_{i=o}^{m} A_i x^i \quad \text{where} \quad \sum_{i=o}^{m} A_i = 2^k$$

is called the weight enumerator polynomial of the AC(m, k, δ, ψ), (A_i is the number of elements of the weight distribution matrix $[W]$ equal

to i).

For any anticode word of weight i in the AC(m, k, δ), there is a code word of weight (2^{k-1} - i) in the corresponding deletion code. An AC(m, k, δ, ψ) has $A_o = 2^{\psi} = 2^{k_i}$ anticode words of weight zero. $(A_o-1) = 2^{\psi}-1$ of these anticode words are related (are identified) with the A_o-1 code words of the DC(n, k, δ) that have weight 2^{k-1}, and the one remaining anticode word of weight zero is related to the single all zero code word of the DC(n, k, d). Therefore, the anticode weight enumerator polynomial $W_{AC}(x)$ is related to the corresponding deletion code weight enumerator polynomial $W^c(x)$ by the polynomial:

$$W^c_{AC}(x,y) = 1 + (A_o-1)\, y^{2^k-1} + \sum_{i=o}^{m} A_i x^i y^{2^{k-1}-i}$$

$$= 1 + (2^{\psi}-1)\, y^{2^k-1} + \sum_{i=o}^{m} A_i x^i y^{2^{k-1}-i}$$

Since the AC(m, k, δ, ψ) is the iteration of the basic Ac(m_o, k_o, δ_o, $\psi_o = 0$), $N = 2^{\psi}$ times, therefore the combined weight enumerator polynomial is simplified into the polynomial

$$W^{c}_{BAC}(x,y) = 1 + \sum_{i=1}^{m} A^{o}_{i} x^{i} y^{(2^{k_o - 1} - i)}$$

where $A^{o}_{i} = A_i / 2^{\psi}$

As an example, consider the AC(9, 5, 6) and the corresponding DC(22, 5, 10). The AC(9, 5, 6) has the generator matrix

$$[G]_{AC} = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and the weight distribution

$$A_o = 2,\ A_1 = A_2 = 0,\ A_3 = 4,\ A_4 = 6,\ A_5 = 12,\ A_6 = 8$$

$$A_7 = A_8 = A_9 = 0$$

The weight enumerator polynomial of the AC(9, 5, 6) is

$$W_{AC}(x) = \sum_{i=o}^{m} A_i x^i$$

$$= 2 + 4x^3 + 6x^4 + 12x^5 + 8x^6$$

where

$$\sum_{i=o}^{m} A_i = 2^k = 32,$$

and the weight enumerator polynomial of the anticode and related deletion code, when $\psi = k_o = 1$, is as

$$W^c_{AC}(x,y) = 1 + (2^{\psi}-1)\, y^{2^{k-1}} + \sum_{i=1}^{m} A_i x^i y^{2^{k-1}-i}$$

$$= 1 + y^{16} + 4x^3y^{13} + 6x^4y^{12} + 12\, x^5y^{11} + 8x^6y^{10} .$$

The weight distribution of the DC(22, 5, 10) is

$$A_o = 1,\ A_2 \to A_9 = 0,\ A_{10} = 8,\ A_{11} = 12,\ A_{12} = 6$$

$$A_{13} = 4,\ A_{14} = A_{15} = 0,\ A_{16} = 1$$

$$A_{17} \to A_{22} = 0 .$$

The weight enumerator polynomial of the unit basic anticode and related unit deletion code is,

$$W^c_{BAC}(x,y) = 1 + \sum_{i=1}^{m} A_i\, x^i y^{(2^{k_o-1} - i)}$$

$$= 1 + 2x^3y^5 + 3x^4y^4 + 8x^6y^2$$

The relationship between the weight enumerator polynomials of both the unit basic anticode and the anticode formed from it extends to the respective deletion codes. We observe from this fact that the con-

struction of large dimension codes from smaller ones is related to the constructing of larger anticodes from the basic one (in general from smaller anticodes, see Section 6.4.6.1). In fact, we can go further and say that the techniques that we might apply for the construction of larger anticodes from the smaller one can be explicitly applied to the construction of larger codes from these of smaller dimension.

6.3.4.1 Anticode Weight Attribute

The set of integers specifying the number of anticode words with weight S and the specific distribution α of these anticode words in the anticode, and β the number of consecutive rows of weight S, is called the weight attribute of the AC(m, k, δ, ψ). This composite anticode parameter is denoted by $W(S; \alpha, \beta)$.

As an example consider the AC(3, 4, 2, 2):

0 0 0
0 0 0
1 0 1
1 0 1
0 1 1
0 1 1
1 1 0
1 1 0
0 0 0
0 0 0
1 0 1
1 0 1
0 1 1
0 1 1
1 1 0
1 1 0

The number of words of weight S = 0 packed in β = 2 consecutive words is started from α = 1 row, then

$$W(S; \alpha, \beta) = W(0, 1, 2) = 4.$$

The number of words of weight S = 2 packed in $\beta = 2$ consecutive words is started from $\alpha = 3^{rd}$ row, then

$$W(S;\ \alpha,\ \beta) = W(2,\ 3,\ 2) = 12\ .$$

It is evident that $W(S;\ \alpha,\ \beta) = A_i$ in the weight enumerator polynomial of the anticode.

The weight attribute of an anticode is also the indicator of the weight density distribution of an anticode. Two anticodes have opposite weight attribute, if the rows of the first anticode which have biggest weight, coincide with rows of the second anticode which have smallest weight.

6.4 Linear Binary Anticode Constructions

The concept of an anticode and some related definitions and properties were introduced and discussed in the previous sections. The iteration property of the parent anticode and the Q-I property of the columns of the parent anticode is inherited by any anticode and by any individual anticode column. Therefore, in constructing any anticode the only permissible columns are the columns which have the above properties.

From the iteration and Q-I properties, it is evident that, in general, a unit anticode $UAC(m_o,\ k_o,\ \delta_o,\ \psi_o)$ can be used to construct an $AC(m,\ k,\ \delta,\ \psi)$ with different parameters. The $UAC(m_o,\ k_o,\ \delta_o,\ \psi_o)$ can be any anticode, even a single column anticode $UAC(1,\ k_o \geq 1,\ 1,\ \psi_o)$.

6.4.1 *The Mapped-Map Stacking*

This anticode construction process is performed by adding to the generator matrix $[G]_{UAC}$, $UAC(m_o,\ k_o, \delta_o, \psi_o)$, k_1 = a rows of all zero elements as the first a rows and k_2 = b rows of all zero elements as the last b rows. The resultant anticode has generator matrix $[G]_{AC}$ given by

$$[G]_{AC} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \quad k_1 = a$$

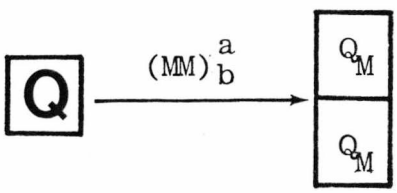
$$[G]_{UAC} \quad k_o$$

$$\begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \quad k_2 = b$$

The linear combination of any subset of the rows of the generator matrix $[G]_{UAC}$ of the UAC(m_o, k_o, δ_o, ψ_o) with any of the newly added rows does not introduce any new anticode word. Thus the maximum distance of the resultant anticode will be the same as that of UAC(m_o, k_o, δ_o, ψ_o). The resultant AC(m, k, δ, ψ) has parameters

$$\left[m = m_o,\ k = k_o + a + b,\ \delta = \delta_o,\ \psi = \psi_o + a + b\right] .$$

The process of mapped map stacking is schematically shown as,



and formulated by

$$UAC(m_o, k_o, \delta_o, \psi_o) \xrightarrow{(MM)\,{}^{a}_{b}} AC(m, k, \delta, \psi) .$$

As an example, consider the UAC(3, 2, 2). Since $\psi_o = 0$, this unit anticode is a basic anticode, with generator matrix,

$$[G]_{BAC} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

This generator matrix contains all 3-tuple column types. Therefore this basic anticode is also an m-sequence anticode.

If $k_1 = a = 1$, and $k_2 = b = 1$, then

$$BAC(3,\ 2,\ 2) \xrightarrow{(MM)_1^1} AC(3,\ 4,\ 2,\ 2)$$

and

$$[G]_{AC} = \begin{bmatrix} 0 \\ \hline [G]_{BAC} \\ \hline 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

the row space of this matrix forms the AC(3, 4, 2, 2):-

0	0	0
0	0	0
1	0	1
1	0	1
0	1	1
0	1	1
1	1	0
1	1	0
0	0	0
0	0	0
1	0	1
1	0	1
0	1	1
0	1	1
1	1	0
1	1	0

the columns of AC(3, 4, 2, 2) consist of the column types (2, 4, 6) of the parent anticode PAC(15, 4, 8). If the AC(3, 4, 2, 2) is deleted from PAC(15, 4, 8), the DC(12, 4, 6) results. This process is formulated as

$$AC(3,\ 4,\ 2,\ 2) \longrightarrow DC(12,\ 4,\ 6)\ .$$

K.E.N. LIBRARY UNIVERSITY

The resultant DC(12, 4, 6) is an optimum code, as this code meets the Griesmer bounds, and also has the same parameters as shown in the table of Helgert & Stinaff (1973). The AC(3, 4, 2, 2) has the modular

representation matrix

$$[N]_{AC} = [\, 0,\ 1,\ 0,\ 1,\ 0,\ 1,\ 0,\ \ldots,\ 0\,]$$

and the modular representation matrix $[N]_{DC}$ of DC(12, 4, 6) is the logical complement of $[N]_{AC}$, therefore

$$[N]_{DC} = [\, 1,\ 0,\ 1,\ 0,\ 1,\ 0,\ 1,\ \ldots,\ 1,\ 1\,] \ .$$

6.4.1.1 Special Cases

(a)

$$\text{If, } \begin{cases} a = o \\ b \neq o \end{cases}$$

then the mapped-map stacking method is simply the $(MM)_b^o$ process, which is a simple stack of the UAC(m_o, k_o, δ_o, ψ_o), 2^b times, and is called the simple stack process. Continuing the previous example, if a = o, b = 2 then

$$\text{UAC}(3,\ 2,\ 2) \xrightarrow{(MM)_2^o} \text{AC}(3,\ 4,\ 2,\ 2)$$

$$[G]_{AC} = \left[\begin{array}{c} \\ |G|_{UAC} \\ \\ \hline \mathbf{0} \\ \mathbf{0} \end{array}\right] = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

This generator matrix consists of the column types (1, 2, 3) of the parent anticode PAC(15, 4, 8), which are:

$$
\begin{array}{ccc}
0 & 0 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
0 & 0 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
0 & 0 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
0 & 0 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
1 & 1 & 1
\end{array}
$$

(b)

$$
\text{If,} \begin{cases} a \neq o \\ b = o \end{cases}
$$

then the mapped-map stacking is the $(MM)_o^a$ process. In performing this process any row of the UAC(m_o, k_o, δ_o, ψ_o) will be repeated 2^a times. In other words, any row is mapped into 2^a rows. This process is called simple mapped stacking. Considering the UAC(3, 2, 2)

if a = 2, b = o, then

$$
\text{UAC}(3,\ 2,\ 2) \xrightarrow{(MM)_o^2} \text{AC}(3,\ 4,\ 2,\ 2)
$$

and

$$
\left[G\right]_{AC} = \left[\begin{array}{c} \mathbf{0} \\ \mathbf{0} \\ \hline [G]_{UAC} \end{array}\right] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}
$$

This generator matrix consists of the column types (4, 8, 12) of the PAC(15, 4, 8), which are

0	0	0
0	0	0
0	0	0
0	0	0
1	0	1
1	0	1
1	0	1
1	0	1
0	1	1
0	1	1
0	1	1
0	1	1
1	1	0
1	1	0
1	1	0
1	1	0

6.4.2 *The Mapped-Inversion Stacked Anticode*

This anticode construction process is performed by adding to the generator matrix of the unit anticode k_1 = a rows of all-zero elements as the first a rows, and k_2 = b rows of all-one elements as the last b rows. The resultant anticode has the generator matrix

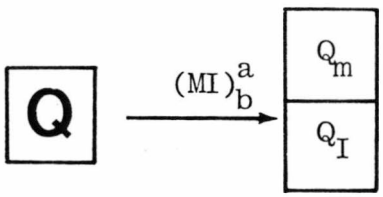
$$[G]_{AC} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \hline [G]_{UAC} \\ \hline \vdots \end{bmatrix} \begin{matrix} \\ \\ k_1 = a \\ \\ k_o \\ \\ k_2 = b \end{matrix}$$

The linear combination of any subset of the rows of the matrix $[G]_{UAC}$, with any subset of the newly added first k_1 = a rows does not introduce any new anticode word. But the linear combination of

any odd number of the newly added $k_2 = b$ last rows with any subset of the rows of the $[G]_{UAC}$ gives a new anticode word. This new anticode word is the logical complement of an anticode word generated by $[G]_{UAC}$. Since the $UAC(m_o, k_o, \delta_o, \psi_o)$ has an all-zeros row, then in the process of forming $(MI)_b^a$ it is inverted into the all-ones row. Therefore, the maximum distance of the resultant anticode is equal to the block length m_o. The resultant anticode $AC(m, k, \delta, \psi)$ has the parameters

$$\left[m = m_o,\ k = k_o + a + b,\ \delta = m_o,\ \psi = \psi_o + a + b - 1 \right] .$$

The process of mapped inversion stacking is schematically shown as



and formulated by

$$UAC(m_o, k_o, \delta_o, \psi_o) \xrightarrow{(MI)_b^a} AC(m, k, \delta, \psi) .$$

As an example, consider the UAC(3, 2, 2) of the previous section. If $a = 1$, $b = 1$, then

$$UAC(3, 2, 2) \xrightarrow{(MI)_1^1} AC(3, 4, 3, 1)$$

where

$$[G]_{AC} = \begin{bmatrix} 0 \\ [G]_{UAC} \\ I \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

The row space of this matrix forms the AC(3, 4, 3, 1), which is:

$$
\begin{array}{ccc}
0 & 0 & 0 \\
0 & 0 & 0 \\
1 & 0 & 1 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
0 & 1 & 1 \\
1 & 1 & 0 \\
1 & 1 & 0 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
0 & 1 & 0 \\
0 & 1 & 0 \\
1 & 0 & 0 \\
1 & 0 & 0 \\
0 & 0 & 1 \\
0 & 0 & 1
\end{array} .
$$

This anticode consists of the column types (10, 12, 14) of parent anticode PAC(15, 4, 8). The modular representation matrix of the AC(3, 4, 3, 1) is

$$[N]_{AC} = [0, \ldots\ldots, 0, 1, 0, 1, 0, 1, 0] .$$

6.4.2.1 Special Cases

(a)

$$\text{If,} \begin{cases} a = o \\ b \neq o \end{cases}$$

then the mapped-inversion stacking is the $(MI)_b^o$ process which is performed on the UAC(m_o, k_o, δ_o, ψ_o) by inverting and stacking this anticode 2^b times. This process is called the sequential inversion stacking process. If a = o, b = 1, it is called the simple inversion stacking process, $(MI)_1^o$.

Considering the UAC(3, 2, 2), if a = o, b = 2 then,

$$\text{UAC}\ (3,\ 2,\ 2) \xrightarrow{(\text{MI})_2^{o}} \text{AC}(3,\ 4,\ 3,\ 1)$$

$$[G]_{AC} = \left[\begin{array}{c} [G]_{UAC} \\ \hline \vdots \end{array}\right] = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

(Although the generator matrix $[G]_{AC}$ does not have an all-zero row, the iteration degree of this anticode is one; i.e., the generator matrix in row canonical form has an all-zero row.) The generator matrix $[G]_{AC}$ consists of the column types (13, 14, 15) of the PAC(15, 4, 8) which are

0 0 0
1 0 1
0 1 0
1 1 0
1 1 1
0 1 0
1 0 1
0 0 1
1 1 1
0 1 0
1 0 1
0 0 1
0 0 0
1 0 1
0 1 1
1 1 0

(b)

$$\text{If} \begin{cases} a \neq o \\ b = o \end{cases}$$

then mapped inversion stacking is the $(\text{MI})_o^a$ process,

which is performed on the UAC(m_o, k_o, δ_o, ψ_o) by adding $k_1 = a$ all-zero rows to the $[G]_{UAC}$ as the first $k_1 = a$ rows, which is identical to the simple mapped stacking $(MM)_o^a$ process of Section 6.4.1.1.a, so

$$(MM)_o^a \equiv (MI)_o^a \quad .$$

(where the symbol $\equiv$ reads "has the same construction").

6.4.3 *The Inverted-Inversion Stack*

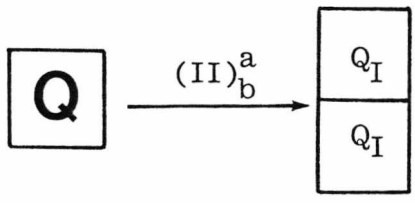
This anticode construction process is performed by adding to the generator matrix $[G]_{UAC}$ of the UAC(m_o, k_o, δ_o, ψ_o), $k_1 = a$ rows of all-one elements as the first a rows and $k_2 = b$ rows of all-one elements as the last b rows. The resultant anticode has the generator matrix

$$[G]_{AC} = \begin{bmatrix} \vdots \\ [G]_{UAC} \\ \vdots \end{bmatrix} \begin{matrix} k_1 = a \\ k_o \\ k_2 = b \end{matrix}$$

The linear combination of any subset of the rows of the matrix $[G]_{UAC}$, with any odd number of newly added rows results in a new anticode word. This new anticode word is the logical complement of an anticode word generated by the matrix $[G]_{UAC}$, therefore the resulting anticode generated by $[G]_{AC}$, contains the all-one row and has the parameters

$$\left[m = m_o,\ k = k_o + a + b,\ \delta = m_o,\ \psi = \psi_o + a + b - 1 \right] .$$

The process of inverted-inversion stacking is schematically shown as



and formulated by

$$UAC(m_o, k_o, \delta_o, \psi_o) \xrightarrow{(II)_b^a} AC(m, k, \delta, \psi).$$

Considering the UAC(3, 2, 2), if $a = 1$, $b = 1$ then

$$UAC(3, 2, 2) \xrightarrow{(II)_1^1} AC(3, 4, 3, 1)$$

where

$$\left[G\right]_{AC} = \begin{bmatrix} I \\ \hline [G]_{UAC} \\ \hline I \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

The rowspace of this matrix forms the AC(3, 4, 3, 1), as

0	0	0
1	1	1
1	0	1
0	1	0
0	1	1
1	0	0
1	1	0
0	0	1
1	1	1
0	0	0
0	1	0
1	0	1
1	0	0
0	1	1
0	0	1
1	1	0

This anticode consists of the column types (11, 13, 15) of the PAC(15, 4, 8).

6.4.3.1 Special Cases

(a)

$$\text{If} \begin{cases} a = o \\ b \neq o \end{cases}$$

then inverted-inversion stacking is the $(II)_b^o$ process which is identical to the sequential inversion process $(MI)_b^o$. Therefore, $(II)_b^o \equiv (MI)_b^o$. Considering the UAC(3, 2, 2), if a = o, b = 2, then

$$\text{UAC}(3,\ 2,\ 2) \xrightarrow{(II)_2^o} \text{AC}(3,\ 4,\ 3,\ 1)$$

and

$$[G]_{AC} = \left[\begin{array}{c|c} [G]_{UAC} & \\ \hline \vdots & \end{array}\right] = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

This anticode has the same parameters and the same generator matrix as AC(3, 4, 3, 1) generated by the $(MI)_b^o$ process.

(b)

$$\text{If} \begin{cases} a \neq o \\ b = o \end{cases}$$

then inverted-inversion stacking is the $(II)_o^a$ process. This process is performed on the UAC(m_o, k_o, δ_o) by repeatedly inverting the rows of this anticode 2^a times, and is called interleaved inversion stacking. Considering the UAC(3, 2, 2), if a = 2, b = o, then

$$\text{UAC}(3,\ 2,\ 2) \xrightarrow{(II)_o^2} \text{AC}(3,\ 4,\ 3,\ 1)$$

$$[G]_{AC} = \left[\begin{array}{c|c} \vdots & \\ \hline [G]_{UAC} & \end{array}\right] = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

this generator matrix consists of the column types (7, 11, 15) of the PAC(15, 4, 8), as follows:

$$
\begin{array}{ccc}
0 & 0 & 0 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
0 & 0 & 0 \\
1 & 0 & 1 \\
0 & 1 & 0 \\
0 & 1 & 0 \\
1 & 0 & 1 \\
0 & 1 & 1 \\
1 & 0 & 0 \\
1 & 0 & 0 \\
0 & 1 & 1 \\
0 & 1 & 0 \\
1 & 0 & 1 \\
1 & 0 & 1 \\
0 & 1 & 0
\end{array}
$$

6.4.4 *Inverted-Map Stacking*

This anticode construction process is performed by adding to the generator matrix $[G]_{UAC}$, k_1 = a rows of all-ones as the first a rows and k_2 = b rows of all-zero elements as the last b rows.

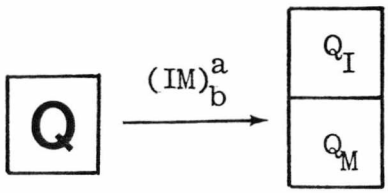
The resultant anticode has the generator matrix

$$
[G]_{AC} = \left[\begin{array}{c|l}
 & k_1 = a \\
|G|_{UAC} & k_o \\
\begin{array}{c} 0 \\ \vdots \\ 0 \end{array} & k_2 = b
\end{array}\right]
$$

The linear combination of any subset of newly added first k_1 = a rows with any subset of the rows of $[G]_{UAC}$ results in a new anticode word. Therefore the resultant anticode contains the all-one anticode word, and this anticode has parameters:

$$\left[\, m = m_o,\ k = k_o + a + b,\ \delta = m_o,\ \psi = \psi_o + (a-1) + b \,\right] \quad .$$

The process of inverted-map stacking is schematically shown as



and formulated by:

$$UAC(m_o,\ k_o,\ \delta_o,\ \psi_o) \xrightarrow{(IM)_b^a} AC(m,\ k,\ \delta,\ \psi).$$

Considering the UAC(3, 2, 2), if a = 1, b = 1 then,

$$UAC(3,\ 2,\ 2) \xrightarrow{(IM)_1^1} AC(3,\ 4,\ 3,\ 1)$$

where

$$\left[G\right]_{AC} = \begin{bmatrix} I \\ \hline [G]_{UAC} \\ \hline O \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

The row space of this matrix forms the AC(3, 4, 3, 1), as follows:

0	0	0
1	1	1
1	0	1
0	1	0
0	1	1
1	0	0
1	1	0
0	0	1
0	0	0
1	1	1
1	0	1
0	1	0
0	1	1
1	0	0
1	1	0
0	0	1

This anticode consists of the column types (3, 5, 7) of the PAC(15, 4, 8).

6.4.4.1 Special Cases

(a)

$$\text{If} \begin{cases} a = o \\ b \neq o \end{cases}$$

then inverted-map stacking is $(IM)^{o}_{b}$, which has the same construction as $(MM)^{o}_{b}$, so

$$(IM)^{o}_{b} \equiv (MM)^{o}_{b}$$

(b)

$$\text{If} \begin{cases} a \neq o \\ b = o \end{cases}$$

then $(IM)^{a}_{o}$ has the same construction as $(II)^{a}_{o}$, and therefore,

$$(IM)^{a}_{o} \equiv (II)^{a}_{o}$$

The following table shows the different two sided-map-inversion processes

$$\begin{array}{ccc} (MM)^{a}_{b} & (MM)^{o}_{b} & (MM)^{a}_{o} \\ & & \| \\ (MI)^{a}_{b} & (MI)^{o}_{b} & (MI)^{a}_{o} \\ & \| & \\ (II)^{a}_{b} & (II)^{o}_{b} & (II)^{a}_{o} \\ & & \| \\ (IM)^{a}_{b} & (IM)^{o}_{b} & (IM)^{a}_{o} \\ & \| & \\ & (MM)^{o}_{b} & \end{array}$$

6.4.5 *Concatenation of the Elements of the Two Sided-Map-Inversion*

The useful feature of the two sided-map-inversion process is the information gained by the analysis of the generator matrix thus formed. This information is in the form of the weight attribute of the anticodes, and provides all the information necessary for the choice of the best

element of the table, for concatenation with a given anticode. It is in this sense that even non-optimum anticodes may be "good" as they can be used to construct larger anticodes.

6.4.5.1 Anticode Concatenation

An anticode with a specific weight attribute $W_{AC}(S;\ \alpha,\ \beta)$ can be concatenated with a different anticode from the two sided-map-inversion table with $W_{AC}(S_o;\ \alpha_o,\ \beta_o)$. The second anticode is chosen such that β_o rows of minimum weight coincide with β rows of maximum weight of the first anticode. As an example, consider the AC(66, 7, 44). The weight attribute of the words of weight 44 is,

$$W(S;\ \alpha,\ \beta) = W(44,\ 2,\ 3) = 3\ .$$

This means that the three consecutive rows starting from the $\alpha = 2$ (2nd) row has the maximum weight 44. The remaining words have weight $\leqslant 36$. AC(15, 7, 8) can be obtained from BAC(15, 4, 8) by the process $(MM)_o^3$, as

$$\text{BAC}(15,\ 4,\ 8) \xrightarrow{(MM)_o^3} \text{AC}(15,\ 7,\ 8,\ 3)\ .$$

From Section 6.4.1.1.b, the $(MM)_o^a$ process does not change the maximum distance of the BAC(15, 4, 8) but it changes the iteration degree from 0 to 3. This means that the weight attribute of the resultant AC(15, 7, 8, 3) for all-zero rows, is W(0, 1, 8) = 8. Therefore, concatenation of these two anticodes results in:

$$\text{AC}(66,\ 7,\ 44) + \text{AC}(15,\ 7,\ 8,\ 3) \equiv \text{AC}(81,\ 7,\ 44)$$

and

$$\text{AC}(81,\ 7,\ 44) \longrightarrow \text{DC}(46,\ 7,\ 20)\ .$$

Although AC(66, 7, 44) is very far from optimality, the resultant AC(81, 7, 4) gives the DC(46, 7, 20), which is very near to the best code predicted by the table of optimum linear codes of Helgert & Stinoff (1973), where the best linear code of n = 46, k = 7 has

d = 20 - 21. Thus it is evident as to how near optimality the resultant AC(46, 7, 20) is.

A third anticode can be concatenated with the resultant AC(81, 7, 44); e.g., the AC(1, 7, 1, 6) resulting from

$$UAC(1, 1, 1) \xrightarrow{(MM)_o^6} AC(1, 7, 1, 6).$$

Therefore,

$$AC(81, 7, 44) + AC(1, 7, 1, 6) \equiv AC(82, 7, 44) \longrightarrow DC(45, 7, 20)$$

The DC(45, 7, 20) is an optimum linear block code.

6.4.5.2 Simple Concatenation

Two or more elements of the two sided-map-inversion table can be concatenated, to build a new anticode. A simple concatenation is performed by the following process:

$$UAC(m_o, k_o, \delta_o) \xrightarrow{(MM)_1^o + (MI)_1^o} AC(2m_o, k+1, 2\delta)$$

and

$$[G]_{AC} = \left[\begin{array}{c|c} [G]_{UAC} & [G]_{UAC} \\ \hline O & I \end{array}\right]$$

As an example, consider the UAC(82, 7, 44). Then

$$UAC(82, 7, 44) \xrightarrow{(MM)_1^o + (MI)_1^o} AC(164, 8, 88) \longrightarrow DC(90, 8, 40) \ .$$

6.4.6 *The General Anticode Construction Technique*

The iteration properties of the parent anticode, and the Q-I systematic form of the columns of the parent anticode, enable the partitioning of the parent anticode, and also an anticode, by a

given column permutation, into smaller dimension anticodes. This was discussed and shown by an example in Section 6.2.1. The PAC(15, 4, 8), as shown in Figure (6-5), was partitioned into smaller dimension basic anticodes and their logical complements. If every basic anticode is symbolized by Q and every logical complement symbolized by I, then Figure (6-5) can be put into the form of Figure (6-6a) and equivalently into the form of Figure (6.6b).

Q	0	Q	0	Q	0	Q	
Q	I	I	0	Q	I	I	
Q	0	Q	I	I	I	I	
Q	I	I	I	I	0	Q	

Figure (6-6a) : Compact Version of the PAC(15, 4, 8)

Q	Q	Q	Q	0	0	0
Q	I	Q	I	I	0	I
Q	Q	I	I	0	I	I
Q	I	I	Q	I	I	0

Figure (6-6b) : Equivalent Compact Form of the PAC(15, 4, 8)

Each of the zeros and ones in Figure (6-6a) are respectively a four-zeros element column, and a four-ones element column. The set of the zeros and ones are the AC(3, 4, 2, 2), put into AC(3, 2, 2) by the reverse process of $(MM)_0^2$. This is a one-to-one mapping of each of the four zeros into one zero and each of the four ones into a single one.

Discarding this AC(3, 3, 2, 2), and mapping one-to-one every Q onto a 0 and every I onto a 1, such that

$$\alpha : Q \longrightarrow 0, \quad I \longrightarrow I,$$

the set of images of Q's and I's, excluding the first all zero's column, in Figure (6-7b) form the linear binary AC(3, 2, 2).

Q	Q	Q	Q
Q	I	Q	I
Q	Q	I	I
Q	I	I	Q

Figure (6-7a) : AC(3, 2, 2) in Q-I Systematic Form.

0	0	0	0
0	I	0	I
0	0	I	I
0	I	I	0

Figure (6-7b) : Resulting Binary AC(3, 2, 2)

6.4.6.1 Iterative Image Stacking

In the previous section the parent anticode, of Figure (6-7), by an iterative process of mappings, was transferred into the smaller dimension anticode. This process was performed as outlined below:

(a) the parent anticode was partitioned into basic anticodes and their logical complements, with interleaved columns.

(b) the partitioned elements (basic anticodes and their logical complements) of the parent anticode were mapped one-to-one onto the set $\{ Q,I \}$ of Figure (6-6a).

(c) the elements of the interleaved columns form the AC(3, 4, 2, 2), by the reverse process $(MM)_o^2$, were mapped one-to-one onto the elements of AC(3, 2, 2) of Figure (6-6b).

(d) the AC(3, 2, 2) was discarded. Figure (6-7a) was the result.

(e) the elements of the set $\{ Q,I \}$ of Figure (6-7a) were mapped one-to-one onto the set of $\{ 0,1 \}$ of Figure (6-7b).

(f) the all-zero column was discarded, and the AC(3, 2, 2) was obtained.

Since every sub-process was performed by one-to-one mapping onto, therefore the process of construction of an anticode from a larger dimension anticode can be reversed. This anticode construction method is called the iterative image stacking method. In general, the process of the iterative image stacking can be summarised as follows:

(a-1) An optimum $AC(m_d, k_d, \delta_d, \psi_d)$ is chosen. An all-zeros column is added to this anticode to form the anticode $(m_d+1, k_d, \delta_d, \psi_d)$, This anticode is called the Domain anticode and is denoted symbolically

$$D_{AC}(m_d+1, k_d, \delta_d, \psi_d) .$$

This is the inverse of the process (f).

(b-1) The optimum $AC(m_r, k_r, d_r, \psi_r)$ is chosen as the image of each "O" element of the $D_{AC}(m_d + 1, k_d, \delta_d, \psi_d)$. The logical complement of the $AC(m_r, k_r, \delta_r, \psi_r)$ is the image of every element "1" in the $D_{AC}(1 + m_d, k_d, \delta_d, \psi_d)$. The $AC(m_r, k_r, d_r, \psi_r)$ is called the First-map anticode and is symbolized as

$$F_{AC}(m_r, k_r, \delta_r, \psi_r) \longleftrightarrow Q$$

The above mapping

$$\alpha = O \longleftrightarrow Q, \quad 1 \longleftrightarrow I$$

results in an anticode with parameters;

$$\begin{cases} m_i = (m_d + 1) \cdot m_r \\ k_i = k_d + k_r \\ \delta_i = (1 + m_d) \cdot \delta_r \\ \psi_i = \psi_d + \psi_r \end{cases}$$

The resultant anticode is called the Image anticode and symbolized as

$$I_{AC}(m_i, k_i, \delta_i, \psi_i) \longleftrightarrow \{Q, I\} \quad .$$

These processes are the inverses of the subprocesses e and b. The whole process so far is formulated as:

$$D_{AC}(1 + m_d, k_d, \delta_d, \psi_d) \xrightarrow{\alpha} F_{AC}(m_r, k_r, \delta_r, \psi_r) \equiv I_{AC}(m_i, k_i, \delta_i, \psi_i) \quad .$$

This reads "Domain anticode $(1 + m_d, k_d, \delta_d, \psi_d)$, α-plus, first mapped anticode, has the same construction as the Image anticode".

The first row of $I_{AC}(m_i, k_i, \delta_i, \psi_i)$ in Figure (6-7a) has only Q elements, therefore the rows of $I_{AC}(m_i, k_i, \delta_i, \psi_i)$ corresponding to these Q's have the biggest weight, δ_i . But, the other rows, due to Q and I combinations have

$$\delta_i \geq \text{max. weight} \geq m_r \cdot \delta_d$$

The smaller the δ_d, which is the maximum distance of the domain anticode, then the smaller max. weight for the Q and I combinations. That is why the $D_{AC}(1 + m_d, k_d, \delta_d, \psi_d)$ is chosen to be optimum. Also the number of Q's which are related to minimum distance d of the domain anticode must not be less than a "proper number" (see the example given in this section).

(c-1) The $AC(m_s, k_s = k_i, \delta_s, \psi_s)$, is chosen in such a way that the 2^{k_r} first rows of this anticode consist of all-zero rows. These 2^{k_r} all zeros rows coincide with the 2^{k_r} first rows of the image anticode which have the maximum weight δ_i. This anticode is called the second mapped anticode, and is symbolized by

$$S_{AC}(m_s, k_s, \delta_s, \psi_s),$$

and is obtained by performing the $(MM)_a^b$ process, on an anticode which is called the second domain anticode and symbolized as $S_{DAC}(m_c, k_c, \delta_c, \psi_c)$, thus

$$S_{DAC}(m_c, k_c, \delta_c, \psi_c), \xrightarrow{\alpha'=(MM)_a^{b=k_r}} S_{AC}(m_c = m_s.\ k_s = k_c + b,\ \delta_s = \delta_c,\ \psi_s = \psi_c + k_r)$$

The concatenation of the second mapped anticode and the image anticode gives the desired linear anticode.

The process (c-1) is the inverse process to that explained in (c) and (d) and shown by figures (6-6b), (6-7b) and (6-5). The whole set of processes (a-1), (b-1) and (c-1) can be formulated as:

$$D_{AC}(1+m_d, k_d, \delta_d, \psi_d) \overset{\alpha}{\dashv} F_{AC}(m_r, k_r, \delta_r, \psi_r) \overset{\alpha|\alpha'}{\dashv} S_{AC}(m_s, k_s, \delta_s, \psi_s)$$

$$\equiv AC(m, k, \delta, \psi) \longrightarrow DC(n, k, d)\ .$$

The symbol $\overset{\alpha|\alpha'}{\dashv}$ reads α-plus-α prime.

The resultant AC(m, k, δ, ψ) has parameters as follows:

$$\begin{cases} m = (m_d + 1)m_r + m_s \\ k = k_d + k_r \\ \delta = m_d \cdot \delta_r \\ \psi = \psi_d + \psi_r \end{cases}$$

As an example, consider the construction of the AC(40, 6, 22), from the optimum AC(10, 4, 6).

(a-1) The Domain anticode is formed as in Figure (6-8) by adding an all-zero column to the AC(10, 4, 6), then:

$$D_{AC}(1 + m_d, k_d, \delta_d, \psi_d) \equiv D_{AC}(1 + 10, 4, 6).$$

(Domain anticode has $\psi_d = 0$).

0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	1	1	0	1	1	0
0	1	0	1	1	0	1	1	0	1	0
0	1	1	0	1	1	0	1	1	0	0
0	0	0	0	1	1	1	0	0	0	1
0	0	1	1	1	0	0	0	1	1	1
0	1	0	1	0	1	0	1	0	1	1
0	1	1	0	0	0	1	1	1	0	1
0	0	0	0	0	0	0	1	1	1	1
0	0	1	1	0	1	1	1	0	0	1
0	1	0	1	1	0	1	0	1	0	1
0	1	1	0	1	1	0	0	0	1	1
0	0	0	0	1	1	1	1	1	1	0
0	0	1	1	1	0	0	1	0	0	0
0	1	0	1	0	1	0	0	1	0	0
0	1	1	0	0	0	1	0	0	1	0

Figure (6-8) - Domain Anticode (1+10, 4, 6)

(b-1) The optimum AC(3, 2, 2) is chosen as the first mapped anticode. Thus,

$$F_{AC}(m_r, k_r, \delta_r, \psi_r) \equiv F_{AC}(3, 2, 2).$$

Each of the zero and one elements of the $D_{AC}(1+10, 4, 6)$ are respectively mapped one-to one onto the $F_{AC}(3, 2, 2) \equiv Q$, and the logical complement of this anticode, I. The resultant image anticode is shown in Figure (6-9):

δ_i	o ← 22										
	18_{16}	Q	Q	I	I	Q	I	I	Q	I	I	Q
	18_{16}	Q	I	Q	Q	I	Q	I	I	Q	I	Q
	18_{16}	Q	I	I	Q	I	I	Q	I	I	Q	Q
	12_{18}	Q	Q	Q	Q	I	I	I	Q	Q	Q	I
	18_{16}	Q	Q	I	I	I	Q	Q	Q	I	I	I
	18_{16}	Q	I	Q	I	Q	I	Q	I	Q	I	I
	18_{16}	Q	I	I	Q	Q	Q	I	I	I	Q	I
	12_{18}	Q	Q	Q	Q	Q	Q	Q	I	I	I	I
	18_{16}	Q	Q	I	I	Q	I	I	I	Q	Q	I
	18_{16}	Q	I	Q	I	I	Q	I	Q	I	Q	I
	18_{16}	Q	I	I	Q	I	I	Q	Q	Q	I	I
	18_{16}	Q	Q	Q	Q	I	I	I	I	I	I	Q
	12_{18}	Q	Q	I	I	I	Q	Q	I	Q	Q	Q
	12_{18}	Q	I	Q	I	Q	I	Q	Q	I	Q	Q
	12_{18}	Q	I	I	Q	Q	Q	I	Q	Q	I	Q

Figure (6-9) : Image Anticode (33, 6, 22)

Hence,

$$D_{AC}(1+m_d,\ k_d,\ \delta_d,\ \psi_d) \overset{\alpha|}{+} F_{AC}(m_r,\ k_r,\ \delta_r,\ \psi_r) \equiv$$

$$I_{AC}(m_i,\ k_i,\ \delta_i,\ \psi_i) \ .$$

The image anticode has parameters

$$\begin{cases} m_i = (m_d+1)m_r = 11 \ . \ 3 = 33 \\ k_i = k_d + k_1 = 4 + 2 = 6 \\ \delta_i = (1 + m_d) \ . \ \delta_r = (1 + 10) \ . \ 2 = 22 \\ \psi_i = \psi_d + \psi_r = 0 \ . \end{cases}$$

Thus

$$D_{AC}(1 + 10,\ 4,\ 6) \overset{\alpha|}{+} F_{AC}(3,\ 2,\ 2) \equiv I_{AC}(33,\ 6,\ 22) \ .$$

The first row of I_{AC} in Figure (6-9) is formed from 11 copies of FAC(3, 2, 2), thus the first row of the $I_{AC}(33, 6, 22)$ is the all-zero row and the 2nd and 3rd and 4th row have weight $\delta = 22$. The remaining rows of the I_{AC} have weight $\leqslant 18$.

(c-1) The AC(7, 3, 4) is chosen as the second domain anticode, hence

$$S_{DAC}(m_c,\ k_c,\ \delta_c,\ \psi_c) = S_{DAC}(7,\ 3,\ 4) \ .$$

The second mapped anticode is obtained as,

$$S_{DAC}(7,\ 3,\ 4) \xrightarrow{\alpha'=(MM)_1^2} S_{AC}(7,\ 6,\ 4,\ 3)$$

The concatenation of this anticode and the image anticode results in AC(40, 6, 22) as in Figure (6-10). Deleting the AC(40, 6, 22) from PAC(63, 6, 32) results in an optimum DC(23, 6, 10). The whole process can be formulated as:

$$D_{AC}(1+10,\ 4,\ 6) \overset{\alpha|}{+} F_{AC}(3,\ 2,\ 2) \overset{\alpha|\alpha'}{+} S_{AC}(7,\ 6,\ 4,\ 3) \equiv AC(40,\ 6,\ 22)$$

$$\longrightarrow DC(23,\ 6,\ 10) \ , \quad \alpha' = (MM)_1^2 \ .$$

Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	Q	0	0	0	0	0	0	0
Q	Q	I	I	Q	I	I	Q	I	I	Q	0	0	1	1	0	1	1
Q	I	Q	I	I	Q	I	I	Q	I	Q	0	1	0	1	1	0	1
Q	I	I	Q	I	I	Q	I	I	Q	Q	0	1	1	0	1	1	0
Q	Q	Q	Q	I	I	I	Q	Q	Q	I	1	1	1	1	0	0	0
Q	Q	I	I	I	Q	Q	Q	I	I	I	1	1	0	0	0	1	1
Q	I	Q	I	Q	I	Q	I	Q	I	I	1	0	1	0	1	0	1
Q	I	I	Q	Q	Q	I	I	I	Q	I	1	0	0	1	1	1	0
Q	Q	Q	Q	Q	Q	Q	I	I	I	I	0	0	0	0	0	0	0
Q	Q	I	I	Q	I	I	I	I	Q	I	0	0	1	1	0	1	1
Q	I	Q	I	I	Q	I	Q	Q	Q	I	0	1	0	1	1	0	1
Q	I	I	Q	I	I	Q	Q	I	I	I	0	1	1	0	1	1	0
Q	Q	Q	Q	I	I	I	I	Q	I	Q	1	1	1	1	0	0	0
Q	Q	I	I	I	Q	Q	I	I	Q	Q	1	1	0	0	0	1	1
Q	I	Q	I	Q	I	Q	Q	Q	Q	Q	1	0	1	0	1	0	1
Q	I	I	Q	Q	Q	I	Q	I	I	Q	1	0	0	1	1	1	0

Figure (6-10) : Compact Inversion of the AC(40, 6, 22). (Every zero and one in Figure (6-7) is a four-zero column and a four-one column)

The iterative image stacking process enables the use of data already available; i.e., the use of already built anticodes to construct larger anticodes. Hence, using this process, the explicit construction of the anticodes with dimension > 7 is possible. This method can also be extended to use the anticodes related to the known optimum codes, from many sources.

6.4.6.2 The Generator Matrix of the Iterated Images Stack Anticode

The stacking and Q-I properties of the parent anticode were discussed, and were illustrated in Figures (6-3) and (6-5), in Section 2.1. These properties are inherited from the properties of the generator matrix of the parent anticode. Thus

Considering these properties and the equation

$$(N-1)_2 = \sum_{i=1}^{k} r_i \, 2^{i-1}$$

of Section 6.2.1., the generator matrix $[G]_{AC}$ of the iterated images anticode is as follows:

The first row of the $I_{AC}(m_i, k_i, \delta_i, \psi_i)$ of Figure (6-9) is the concatenation of the m+1 copies of the $Q = F_{AC}(m_r, k_r, \delta_r, \psi_r)$. The rest are the elements of the first row, stacked due to the elements of the $N^{th} = (1+2^i)^{th}$, (i = o, 1, ..., k-1) rows of the I_{AC} which are exactly the rows of the generator matrix of the $D_{AC}(1+m_d, k_d, \delta_d, \psi_d)$ where each zero is a Q and each one is an I. Therefore, the first k_r rows of the generator matrix of the I_{AC} are the rows of m+1 concatenated copies of the $F_{AC}(m_r, k_r, \delta_r, \psi_r)$. The rest are the rows of the generator matrix of $D_{AC}(1+m_d, k_d, \delta_d, \psi_d)$ where each zero element is mapped into the m = 3 zero row and each one element is mapped into m = 3 ones. The concatenation of the resultant generator matrix with the generator matrix of the $S_{AC}(m_c, k_c, \delta_c, \psi_c)$ is the generator matrix $[G]_{AC}$ of the AC(m, k, δ, ψ).

As an example, consider the generator matrix of the AC(40, 6, 22) of the previous section.

The component generator matrices are

$$[G]_{F_{AC}} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} = G_Q \quad \& \quad [G]_{D_{AC}} = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

and

$$[G]_{S_{AC}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Therefore the generator matrix of AC(40, 6, 22) is,

$$[G]_{AC} = \left[\begin{array}{c|c|c|c|c|c|c|c|c|c|c|c} G_Q & G_Q & G_Q & G_Q & G_Q & G_Q & G_Q & G_Q & G_Q & G_Q & G_Q & \begin{matrix}0000000\\0000000\end{matrix} \\ \hline 000 & 000 & 111 & 111 & 000 & 111 & 111 & 000 & 111 & 111 & 000 & 1010101 \\ 000 & 111 & 000 & 111 & 111 & 000 & 111 & 11 & 000 & 111 & 000 & 0110011 \\ 000 & 000 & 000 & 000 & 111 & 111 & 111 & 000 & 000 & 000 & 111 & 0001111 \\ 000 & 000 & 000 & 000 & 000 & 000 & 000 & 000 & 111 & 111 & 111 & 0000000 \end{array}\right]$$

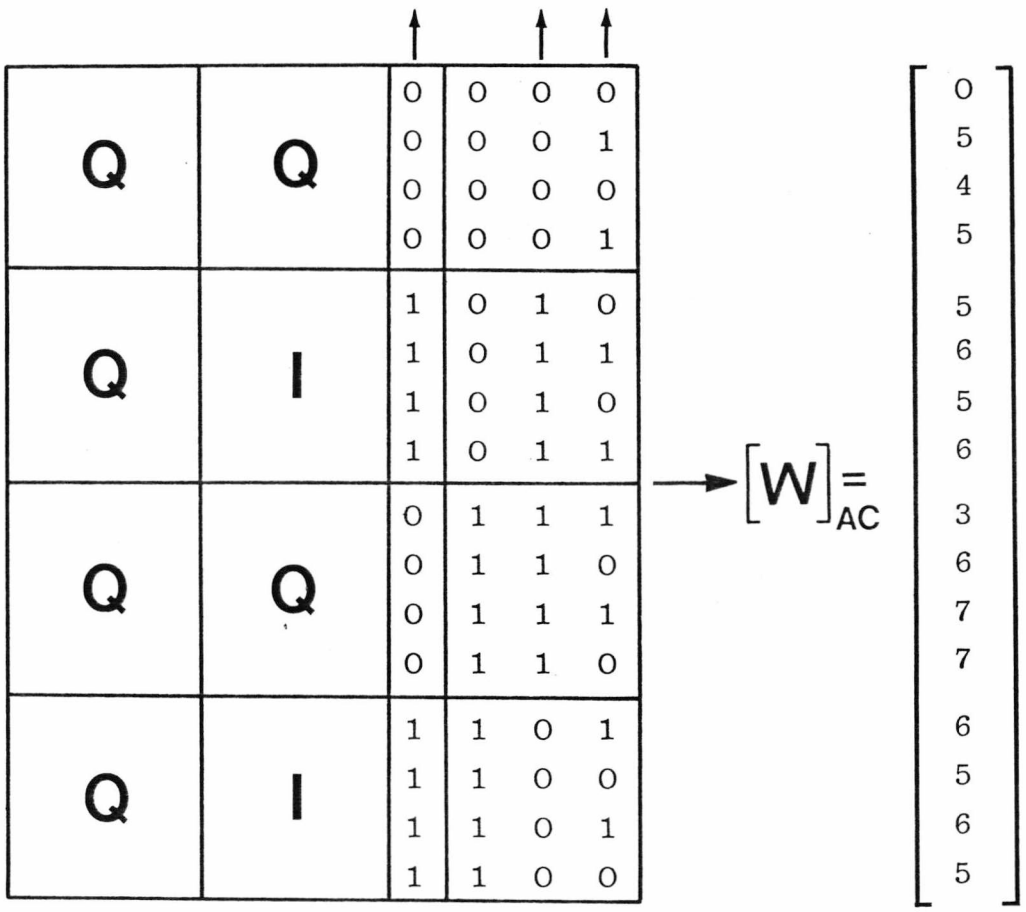
The generator matrix of D_{AC} and S_{AC} has no repeated columns. Thus the resultant AC(m, k, δ, ψ) has no repeated columns. This can be easily verified for the above $[G]_{AC}$.

6.4.7 *The Column Refinement Process*

An AC(m, k, δ, ψ), constructed by the processes explained in previous sections, if it does not meet the desired parameters; that is,does not have parameters equal or nearly equal to those parameters defined by the functional relationships between the anticode parameters known as bounds;

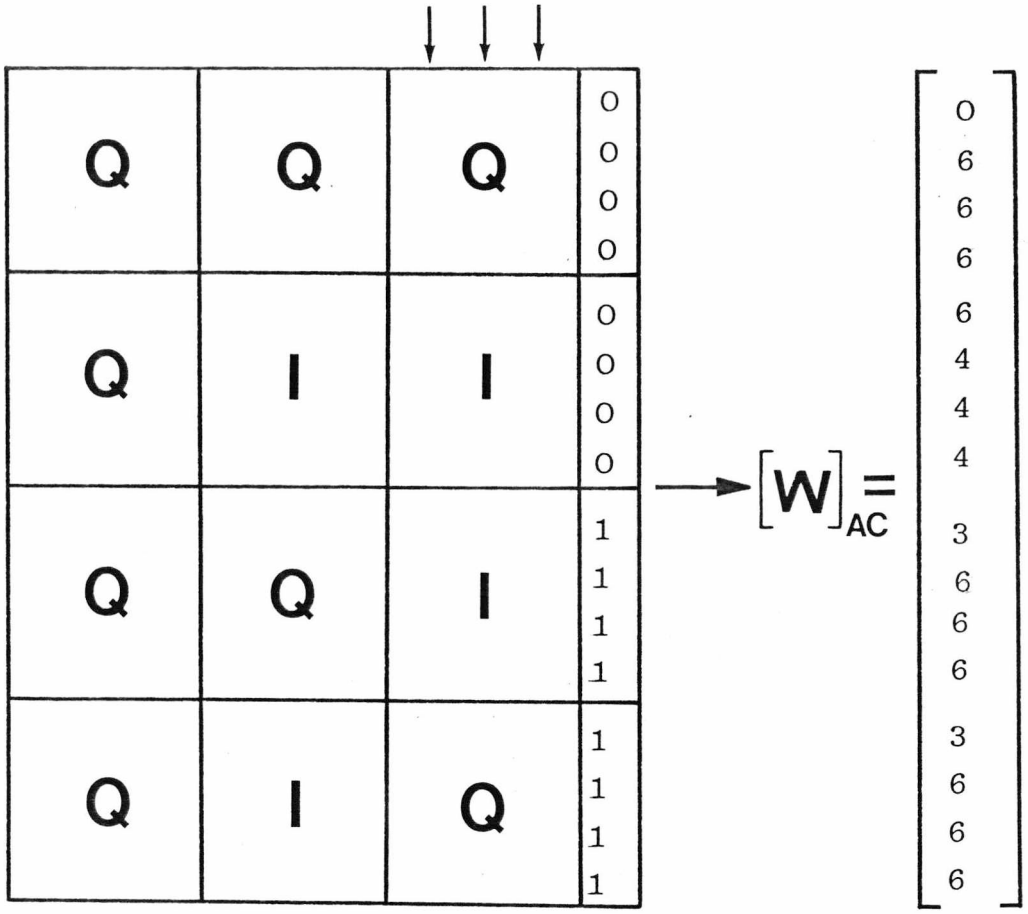
it is either not constructable, or the columns of this anticode have not been suitably chosen. To obtain an optimum or a near-optimum anticode from this anticode, it can be refined (reconstructed) by simultaneously extending and puncturing (or in the reverse order) the columns of the constructed anticode. As an example, consider the construction of AC(10, 4, 6), from the non-optimum constructed AC(10, 4, 7).

The AC(10, 4, 7) has the weight distribution vector $[W]$ and construction as



where Q = AC(3, 2, 2) and I is the logical complement of the AC(3, 2, 2).

If the columns indicated by arrows are deleted, and the anticode is simultaneously extended by the same number of columns but different and suitable columns, those of types $\{13, 14, 15\}$, the optimum AC(10, 4, 6) is resultant, as follows:



Another example is the refinement of AC(144, 8, 77). The AC(144, 8, 77) gives the DC(111, 8, 51). The values (50 - 54) are the values of the lower and upper bound for the minimum distance d of the DC(111, 8, d). The AC(144, 8, 77) can be extended as:

$$\text{AC}(144, 8, 77) + 3 \text{ column types } \{158, 173, 179\} \equiv \text{AC}(147, 8, 78).$$

The AC(147, 8, 78) can be punctured as

$$\text{AC}(147, 8, 78) \overset{D^*}{-} \text{column types } \{25, 70, 128\} \equiv \text{AC}(144, 8, 76)$$
$$\rightarrow \text{DC}(111, 8, 52) .$$

Thus the minimum Hamming distance of the DC(111, 8, 51) has been increased by one. The process of the refinement of a non-optimum anticode by simultaneously extending and puncturing or vice versa, is identical to the puncturing and extending of the related deletion code.

---

* The symbol $\overset{D}{-}$ reads D-minus.

6.4.8 *The Explicit Derivation of the Anticode Weight Distribution*

The Q-I property of each column of an anticode, is used to enumerate the weight distribution of an anticode, and thus of the related code.

Each column of the generator matrix $[G]_{AC}$ can be spanned, by using this property; i.e., the first two top elements of each column in $[G]_{AC}$ are spanned into a 4-tuple column in the anticode. This 4-tuple column is stacked with respect to the 3rd element of the related column in the $[G]_{AC}$. The resultant 8-tuple column again will be stacked with respect to the 4th element of the related column in $[G]_{AC}$, to give a 16-tuple column in the anticode. This process of stacking is continued to the last element in the related column in $[G]_{AC}$, and the number of stackings will be k - 2. Therefore, instead of spanning the row space of the $[G]_{AC}$, by finding $m.2^k$ different linear combination of the columns, which for a fairly big k needs a large amount of computation time, it can be done simply by stacking each column as explained and find the weight shape of the column in the anticode weight distribution. For each column there are (k - 2) stackings, and a total of m(k - 2) stacking operations, which is a big reduction in computation time.

6.4.9 *The Anticode Complementary Computer Search*

The anticode construction methods for obtaining deletion error correcting codes, from the related parent anticode, were studied in previous sections. It was shown that the anticode construction process previously described has the advantage that the data already available concerning the parameters, and construction of the anticodes and codes can be used to construct many anticodes with large dimension. If the constructed anticode is not optimum or near optimum, the anticode can be refined by using simultaneous extension and puncturing processes,

which results in an optimum or near optimum anticode. The process of refinement of a non-optimum anticode can be performed in many different ways, explained as follows:

(a) A non-optimum anticode might be optimized by deleting unsuitable columns of this anticode to obtain an optimum or near-optimum shorter anticode. We attempt to delete as few columns as possible, but at the same time to reduce the maximum distance δ of the anticode as much as possible. This is equivalent to the addition of as few columns as possible to a code, but at the same time to increase the minimum distance d of the code as much as possible.

(b) A non-optimum anticode may be extended by adding suitable columns. We attempt to add as many columns as possible, which increases the maximum distance δ of the anticode as little as possible. This is equivalent to deleting as many columns as possible from the code, which decreases the minimum Hamming distance d of the code as little as possible. From Section 6.3.4.1, the suitable columns in both cases (a) and (b) are the columns which have weight attribute opposite to the anticode weight attribute.

(c) An optimum anticode constructed by the processes described in the previous section, or other suitable anticodes, may be extended or shortened, to obtain other optimum or near optimum anticodes with different parameters.

It was possible to write a computer programme in the Fortran IV language, to implement on an ICL 2960 machine, in order to refine in different ways, the existing anticodes constructed by the previous methods, being either optimum or non-optimum. The programme is based on the Q-I and stacking properties of the columns of the generator matrix of the anticode and also the explicit derivation of the anticode weight distribution. These provide a fast procedure for the choice of

the most suitable columns of the parent anticode, excluding those of the anticode being refined.

CHAPTER VII

FURTHER PROPERTIES OF THE LINEAR BINARY ANTICODE AND LINEAR BINARY ANTICODE CONSTRUCTION RESULTS

7.1 Anticodes and Codes-related Search Facilities

In the selection of an anticode, alternative anticodes may be compared on the basis of various probabilistic measure of the performance of the related codes. In the process of comparison, knowledge of some common properties of the anticodes which are being compared may be useful in narrowing the search for the best selection of anticodes (codes) among the others. When comparing anticodes with the same parameters m and k, the optimality of one with respect to the others may be defined in terms of the performance of the related code, as the one having the smallest average probability of error. In the classification of anticodes the relation "are combinatorially equivalent" partitions the set of anticodes (also the related codes) into equivalence classes. Each equivalence class consists of the anticodes which are combinatorially equivalent, therefore the related codes are in the same equivalence class and have the same performance (for random errors). Hence only one anticode (code) in an equivalence class is compared with each one selected from other equivalence classes. Although the properties of anticodes are opposite to those of related codes, any property of an anticode has a corresponding opposite property in the related code, and vice versa.

7.1.2 *Equivalent Anticodes and Related Equivalent Codes*

An anticode is an iterated subspace of a vector space. Therefore, it can be formed, spanning the rows of its generator matrix $[G]_{AC}$, by premultiplying with the $2^k \times k$ matrix $[M_o]^T$ of Section 6.2, i.e.

$$[M_o]^T \cdot [G]_{AC} \equiv AC(m, k, \delta, \psi).$$

If $[M_o]^T$ is multiplied by a(k × k) non-singular matrix (elementary column transformation matrix, see Section 2.9), $[S]$, since the 2^k rows of $[M_o]^T$ are different then the resultant matrix,

$$[B] = [M_o]^T \cdot [S],$$

is the same as matrix $[M_o]^T$, where the rows have been rearranged (permuted). Thus

$$[M_o]^T \cdot [S] = [A]\ [M_o]^T \ .$$

In the case of linear codes the elementary row transformation matrix $[A]$ was called by Fontaine & Peterson (1966) A-permutation matrix.

The parent anticode $[C_o]$, (without loss of generality the parent anticode $[C]$ excluding the all-zero row is being considered in this section) is the row space of the generator matrix $[M]$, spanned by the matrix $[M]^T$.

The effect of A-permutation on $[M_o]^T$ or, identically, on $[M]^T$, has the same effect on the rows of $[C_o]$, i.e.

$$[A] \cdot [C_o] = [A] \cdot [M]^T \cdot [M] \ .$$

If both sides of this equation are multiplied on the left by the A-permutation matrix $[P]$ where $[P]$ is chosen to be,

$$[P] \cdot [M]^T = [M]^T \cdot [U] \quad \text{and} \quad [U] = [S^{-1}]^T$$

then

$$\begin{aligned}
[A] \cdot [C_o] \cdot [P]^T &= [A] \cdot [M]^T \cdot [M] \cdot [P]^T \\
&= [A] \cdot [M]^T \cdot \left([P] \cdot [M]^T \right)^T \\
&= [M]^T \cdot [S] \cdot \left([M]^T \cdot [U] \right)^T \\
&= [M]^T \cdot [S] \cdot \left([U]^T \cdot [M] \right) \\
&= [M]^T \cdot [S] \cdot [S^{-1}]^T \cdot [M] = [C_o].
\end{aligned}$$

Since $[P]$ is a non-singular matrix, then $[P]^T = [P^{-1}]$, and

$$[A] \cdot [C_o] = [C_o] \cdot [P] \cdot$$

From this equation it is evident that row permutation on the parent anticode, or alternatively, multiplying the matrix $[C_o]$ on the left by a permutation matrix $[A]$, results in a related A-permutation $[P]$ on the columns of the parent anticode $[C_o]$, and vice versa.

In the study of equivalent anticode we consider two cases.

7.1.2.1 The Column Equivalent Anticodes

Consider the AC(m, k, δ, ψ), rearranging the columns of this anticode, or alternatively multiplying the generator matrix $[G]_{AC}$ of this anticode on the right by a permutation matrix $[T]$, gives:

$$[G]_{ACT} = [G]_{AC} \cdot [T] \cdot$$

The matrix $[G]_{ACT}$ is the same as $[G]_{AC}$, except that the columns have been rearranged. Therefore the same columns will be generated by $[G]_{ACT}$ as well as by $[G]_{AC}$. Hence identical columns will be deleted from the parent anticode and the resulting codes are the same, (opposite to definition for codes).

As an example, consider AC(3, 3, 2, 1) generated by

$$[G]_{AC} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix},$$

then

$$[M_o]^T \cdot [G]_{AC} = \begin{matrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{matrix} = \text{AC}(3, 3, 2, 1)$$

If the generator matrix $[G]_{AC}$ is multiplied by the permutation matrix

$$[T] = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Then,

$$[G]_{AC} \cdot [T] = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} = [G]_{ACT}$$

Therefore

$$[M_o]^T \cdot [G]_{ACT} = \begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{matrix} = AC_T(3, 3, 2)$$

AC(3, 3, 2) and $AC_T(3, 3, 2)$ have the same columns, but in different order and therefore have the same modular representation matrix $[N]$ as,

$$[N]_{AC} = \begin{bmatrix} 1, 1, 1, 0, 0, 0, 0 \end{bmatrix}$$
$$[N]_{ACT} = \begin{bmatrix} 1, 1, 1, 0, 0, 0, 0 \end{bmatrix} .$$

Thus the modular representation matrices of the related codes are the same, i.e.

$$[N]_{DC} = \begin{bmatrix} 0, 0, 0, 1, 1, 1, 1 \end{bmatrix}$$
$$[N]_{DCT} = \begin{bmatrix} 0, 0, 0, 1, 1, 1, 1 \end{bmatrix} .$$

Also, the weight distribution matrix $[W]_{AC}$ and $[W]_{ACS}$ are the same,

$$[W]_{AC}^T = \begin{bmatrix} 0, 2, 2, 2, 0, 2, 2, 2 \end{bmatrix}$$

and

$$[W]^T_{ACT} = \left[0,\ 2,\ 2,\ 2,\ 0,\ 2,\ 2,\ 2 \right] \quad ,$$

(Note: all-zero anticode words are included),

which are identified by

$$[W]_{AC} = [W]_{ACT} = [C][N]^T \quad .$$

7.1.2.2 Row Equivalent Anticodes

Consider the AC(m, k, δ, ψ), rearranging the rows of the generator matrix of this anticode, or alternatively multiplying the generator matrix $[G]_{AC}$ of this anticode on the left by a non-singular matrix $[S]$, gives

$$[G]_{ACS} = [S] \cdot [G]_{AC} \quad .$$

The anticode generated by $[G]_{ACS}$ can be formed as

$$[M]^T \cdot [G]_{ACS} = [M]^T \cdot [S] \cdot [G]_{AC} \quad .$$

From Section 7.1.2,

$$[M]^T \cdot [S] = [A] \cdot [M]^T \quad .$$

Therefore

$$[M]^T \cdot [G]_{ACS} = [A] \cdot [M]^T \cdot [G]_{AC}$$

where $[A]$ is a $(2^k-1) \times (2^k-1)$ row permutation matrix. From this equation it is evident that the rows of the anticode generated by $[G]_{ACS}$ are the same as those of the anticode generated by $[G]_{AC}$, but with different order, thus with different columns. Therefore, two different anticodes which result in two trivially different codes (again, an opposite property).

From the above equation it is also evident that the weight distribution vectors $[W]_{AC}$ and $[W]_{ACS}$ are related, as

$$[W]_{ACS} = [A] \cdot [W]_{AC} \quad .$$

Therefore the two weight vectors are related by the matrix $[A]$. The modular representation matrix $[N]_{ACS}$ can be formed as

$$[N]^T_{ACS} = [C]^{-1} \cdot [W]_{ACS} = [C]^{-1} \cdot [A] \cdot [W]_{AC} =$$

$$= [P] \cdot [C^{-1}] \cdot [W]_{AC} = [P] \quad [N]^T_{AC} \cdot$$

Thus the modular representation $[N]_{ACS}$ is the same as $[N]_{AC}$, but with a related element permutation. As a simple example, consider the AC(3, 3, 2, 1) of the previous section:

$$[G]_{AC} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad AC(3,\ 3,\ 2,1) = \begin{matrix} 0\ 0\ 0 \\ 1\ 0\ 1 \\ 0\ 1\ 1 \\ 1\ 1\ 0 \\ 0\ 0\ 0 \\ 1\ 0\ 1 \\ 0\ 1\ 1 \\ 1\ 1\ 0 \end{matrix} .$$

If the generator matrix $[G]_{AC}$ is multiplied by the non-singular matrix $[S]$, where

$$[S] = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} ,$$

then

$$[G]_{ACS} = [S] \cdot [G]_{AC} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} .$$

The row space of the matrix $[G]_{ACS}$ is formed as follows:

$$[M_o]^T \cdot [G]_{ACS} = \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 0 \end{matrix} = AG_S(3, 3, 2)$$

The AC(3, 3, 2,1)and $AC_S(3,3,2,1)$ are formed from different columns, hence these two anticodes produce two different codes.

The generator matrices of the two anticodes AC(3, 3, 2, 1) and $AC_S(3, 3, 2,1)$ are different, so the modular representation and weight matrices are different but related by the A-permutation matrices. Since

$$[A] \cdot [M]^T = [M]^T \cdot [S] \ ,$$

and

$$[S] = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} ,$$

it is readily verfied that,

$$[A] = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} .$$

The weight matrix of the $AC_S(3,3, 2,1)$ can be formed as

$$[W_o]_{ACS} = [A] \cdot [W]_{AC} \quad \text{then} \quad [W_o]^T_{ACS} = \begin{bmatrix} 0, & 0, & 2, & 2, & 2, & 2, 2, 2 \end{bmatrix} .$$

The modular representation matrix of the $AC_S(3, 2, 2)$ can be formed as:

$$[N]^T_{ACS} = [P]\ [N]^T_{AC}$$

where

$$[P]\ [M]^T = [M]^T\ [U]\ ,\ [S^{-1}]^T = [U]$$

Since

$$[S^{-1}] = [S] = [U]\ ,\ \text{thus}\ [P] = [A]\ ,$$

(Note, for the sake of simplicity $[S]$ has been chosen such that $[S^{-1}] = [S]$).

Therefore

$$[N]^T_{ACS} = [A]\ [N]^T_{AC}\ ,\ \text{and}$$

$$[N]_{ACS} = [0,\ 1,\ 0,\ 1,\ 0,\ 1,\ 0]\ .$$

Since the modular representation of the related code is the logical complement of the $[N]_{ACS}$ then

$$[N]_{DCS} = [1,\ 0,\ 1,\ 0,\ 1,\ 0,\ 1]\ .$$

This means that a row operation on the generator matrix of an anticode, or alternatively, multiplying the generator matrix of an anticode on the left by a non-singular matrix, results in an identical related permutation $[A]$ on the weight distribution matrix of the anticode and related code. Also an identical related permutation $[P]$ results on the modular representation of the anticode and related code. However, column permutation of the generator matrix of an anticode affects neither the weight distribution matrix, nor the modular representation matrix. This leads to the conclusion that equivalent anticodes and related equivalent codes share the same permutation property when an elementary row operation is performed on the generator matrix of the anticode. Also, if the modular representation and weight distribution matrices of two anticodes, or the corresponding codes are related by an A-permutation matrix they are in the same equivalence class. An interesting result may be the use of this correspondence

when considering the equivalence classes of a code. Since $n_{DC} + m_{AC} = n_{PAC}$, thus the one will be considered which has the smallest value of n_{DC} or m_{AC}, where n_{DC} and m_{AC} are block length of the deletion code and related anticode.

7.1.3 *Maximum Distance Bounds for Linear Anticodes*

The maximum Hamming distance of a linear anticode is an important parameter for evaluation of the performance of the related code. Thus it is quite useful to know the ultimate value of the maximum distance for a certain block length and the dimension of an anticode. These values have been given in the form of a functional relationship between the anticode parameters, m, k and δ. These relationships are in the form of bounds which relate any one of the given parameters in terms of the remaining two.

Since the iteration degree ψ of an anticode has no effect on the maximum distance, the bounds are considered for anticodes having $\psi = 0$, alternatively anticodes of no repeated columns; thus $k = k_o$. A linear AC(m, k, δ, $\psi = 0$) is a linear subspace V_k of the vector space V_m of all the m-tuple vectors over GF(q). Thus, if $\bar{x}$ be an anticode word then the number of words in V_m which are exactly at a distance d from $\bar{x}$ is the number of resulting vectors, when all the vectors of weight d are added to vector $\bar{x}$ which is,

$$\binom{m}{d}(q-1)^d \quad .$$

The total number of words of V_m which are at distance $\delta + i$ where $i = 1, 2, \ldots, m - \delta$ from $\bar{x}$ must be

$$\binom{m}{\delta+1}(q-1)^{\delta+1} + \binom{m}{\delta+2}(q-1)^{\delta+2} + \ldots. +$$

$$\binom{m}{m}(q-1)^m = \sum_{i=\delta+1}^{m} \binom{m}{i}(q-1)^i$$

This is the number of vectors that are at the distance greater than δ from $\bar{x}$ and they cannot be in $V_k \equiv AC(m, k, \delta)$.

Since there are $|V_k|$ anticode words, then the total number of possible vectors of V_m which are at distance greater than δ and are excluded from $V_k \equiv AC(m, k, \delta)$ are

$$|V_k| \sum_{i=\delta+1}^{m} \binom{m}{i}(q-1)^i .$$

But, in general this number will be greater than all the absent vectors in V_k.

For an anticode word $\bar{x}$ we may have excluded a vector $\bar{y} \in V_m$ because of a distance $\delta+i$ away from $\bar{x}$, but for a different anticode word $\bar{z} \in V_k$ we could have excluded $\bar{y}$ on the basis of being $\delta+j$ away. Thus, it reveals that the number of vectors of anticode $AC(m, k, \delta)$ and calculated number of vectors which are absent is greater or equal to V_m, so

$$|V_k| + |V_k| \sum_{i=\delta+1}^{m} \binom{m}{i}(q-1)^i \geq V_m$$

and

$$1 + \sum_{i=\delta+1}^{m} \binom{m}{i}(q-1)^i \geq \frac{|V_m|}{|V_k|}$$

but

$$|V_k| = q^k \quad \text{and} \quad |V_m| = q^m$$

Then

$$q^{m-k} - 1 \leq \sum_{i=\delta+1}^{m} \binom{m}{i}(q-1)^i$$

Using the following inequality (Peterson & Weldon 1972, Appendix A, Park 1969):

$$\sum_{i=m\lambda}^{m} \binom{m}{i} \leq (\mu)^{-\mu m} (\lambda)^{-\lambda m}$$

where $\lambda > \frac{1}{2}$ and also $\mu = 1 - \lambda$ and $m >> 1$. Then, for the binary case we have

$$2^{m-k}-1 \leq \left(\frac{m - (\delta+1)}{m}\right)^{-\left[m-(\delta+1)\right]} \cdot \left(\frac{\delta+1}{m}\right)^{-(\delta+1)} = 2^{mH\left(\frac{\delta+1}{m}\right)}$$

If $2^{m-k} >> 1$, taking the logarithm of both sides, we have

$$m - k \leq mH\left(\frac{\delta+1}{m}\right)$$

or

$$\delta \leq m-1$$

$$1 - \frac{k}{m} \leq H\left(\frac{\delta+1}{m}\right)$$

where $H\left(\frac{\delta+1}{m}\right)$ is the entropy function of $\left(\frac{\delta+1}{m}\right)$. This inequality provides a lower bound on the ratio of min-maximum distance δ over the length m of any anticode.

This bound, with a different derivation procedure, was also found by Hashim (1974). Farrell (1969) has established a lower bound on the min-maximum distance of an anticode, by taking into account the fact that the min-maximum weight δ of any anticode AC(m, k, δ) is greater or equal to the average weight of the anticode words:

$$m\, q^{k-1} (q - 1)/q^k - 1 \leq \delta .$$

This can be written as

$$\delta \geq \frac{m}{q} (q - 1) \frac{q^k}{q^k-1}$$

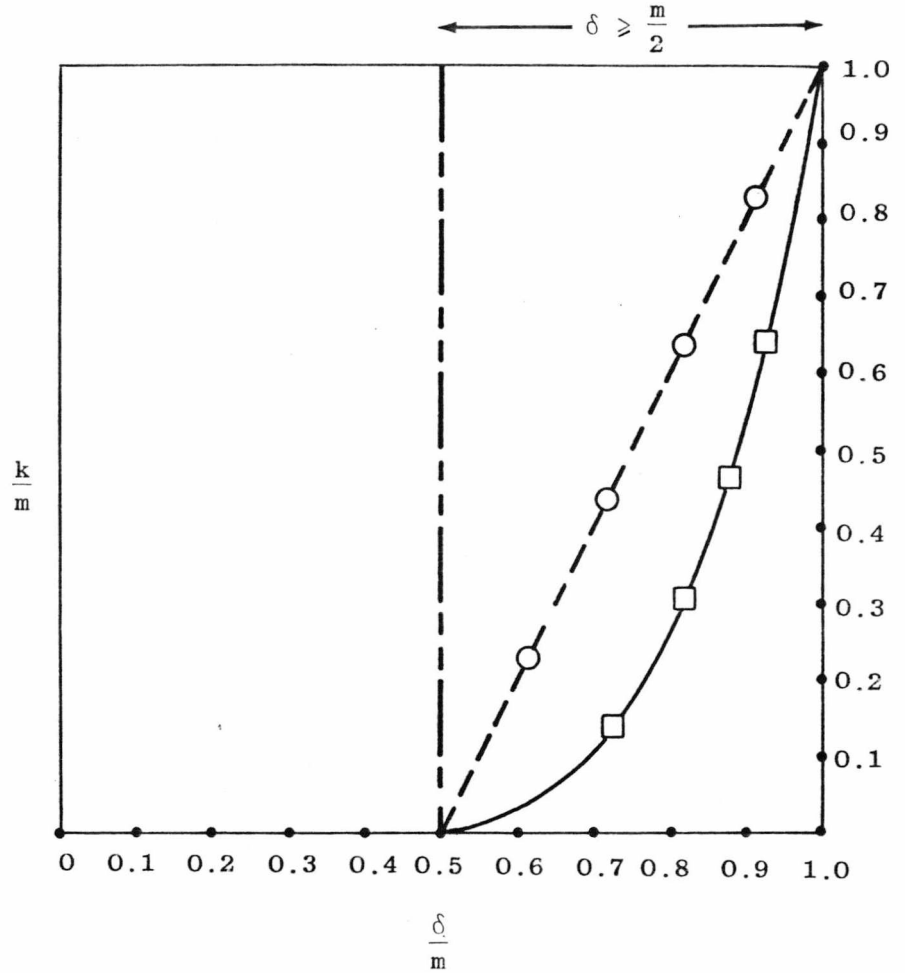
since $q^k/q^k - 1$ is always larger than and near to one, then

$$\delta \geq \frac{m}{q}(q - 1)$$

and for the binary case $q = 2$, so

$$\delta \geq \frac{m}{2}$$

which is a Plotkin-like bound. These bounds are plotted in Figure (7-1).



□ Bound of Section 7.1.3

○ Farrell, Plotkin-like Bound

Figure (7-1) : Anticode Bounds

7.1.4 *The Binary Anticode Construction Result*

Using the methods described in the previous chapter, it was possible to construct linear binary anticodes for $7 < k < 10$. A few small dimension anticodes ($k < 7$) have been chosen from the previously found anticodes by Farrag (1976) as the unit anticode in the construction of bigger dimension anticodes; these are marked by the symbol " * " whenever they are used.

Although it would have been possible to use anticodes related to optimum codes from various other sources, in fact only anticodes which have been built up from unit smaller dimension anticodes have been used. Thus the unit optimum anticode construction is important in the sense that it enables construction of a large number of other optimum or new optimum codes.

The unit anticodes used as the basis for the construction of the bigger dimension anticodes are given in this section, by the set of column types of their generator matrix. The anticodes constructed with the previously explained processes are introduced by means of the same formulation as in Sections 6.4.6. and 6.4.7. The newly built deletion codes related to these anticodes are symbolical as:-

$$AC(213,8,112) \longrightarrow \overline{DC(42,8,16)}_{(16\text{-}18)} \quad .$$

The subscripts in the parenthesis (16-18) are the values of the lower and upper bound for the minimum distance d of the DC(42,8,d), given by

the table of Solomon & Stiffler (1973). However, the range of the table is restricted to $1 \leqslant n \leqslant 127$ and $1 \leqslant k \leqslant n$. Thus for $n > 127$ the subscripts have been evaluated from the Griesmer upper bound of Section 5.2 on the minimum distance of a linear block code over GF(q) by means of a simple computer programme. A single value subscript means the constructed code has the same value as predicted by the bound, whereas a double value ($n > 127$) subscript means that the right hand value is the upper bounded value of minimum distance given by Griesmer and the left hand value is the actual distance of the constructed code. If the results of two different construction processes give the same anticode, the processes are shown to be the same by the symbol $\Longleftrightarrow$. The results of the linear binary anticode constructions for $7 \leqslant k \leqslant 10$ are given as follows:

The generator matrix of the $AC^{*}(21,5,12)$ consists of column types $\{$ 1-2, 4-7, 9-12, 16-17, 19-23, 25-28 $\}$ and has min distance $d = 8$.

The generator matrix of the AC(3,2,2) consists of column types $\{1, 2, 3\}$.

The generator matrix of the AC(15,4,8) consists of column types $\{1 - 15\}$. Therefore,

$$D_{AC}(1+21,5,12) \overset{\alpha}{+} F_{AC}(3,2,2) \overset{\alpha \mid \alpha'}{+} S_{AC}(15,7,8,3) \equiv$$

$$AC(81,7,44) \longrightarrow \boxed{DC(46,7,20)_{(20-21)}} \qquad \alpha' = (MM)_{o}^{2}$$

$$AC(81,7,44) + \text{column types } \{17, 19, 34, 53, 64, 68\} \equiv$$

$$AC(87,7,47) \longrightarrow \boxed{DC(40,7,17)_{(16-18)}}$$

$$AC(81,7,44) + \text{column types } \{17, 19, 34, 53, 64\} \equiv$$

$$AC(86,7,44) \longrightarrow \boxed{DC(41,7,17)_{(16-18)}}$$

$AC(81,7,44)$ + column types $\{19, 34, 53, 64\} \equiv$

$$AC(85,7,46) \longrightarrow \boxed{DC(42,7,18)_{(17-19)}}$$

$AC(81,7,44)$ + column type $\{64\} \equiv$

$$AC(82,7,44) \longrightarrow \boxed{DC(45,7,20)_{(20)}}$$

The generator matrix of the $AC^*(36,6,20)$ consists of column types $\{3, 5\text{-}7, 9\text{-}14, 17\text{-}22, 24\text{-}26, 28, 33\text{-}38, 40\text{-}42, 44, 48\text{-}50, 52, 56, 63\}$, and min distance d = 16. Therefore,

$$AC^*(36,6,20) \xrightarrow{(MI)_1^o} \boxed{DC(36,7,16)^* \longrightarrow AC(91,7,48)}$$

$AC(91,7,48)$ + column types $\{67, 81, 74, 71\} \equiv$

$$AC(95,7,51) \longrightarrow \boxed{DC(32,7,13)_{(12-14)}}$$

The generator matrix of the $AC^*(48,6,26)$ consists of column types $\{3\text{-}7, 9\text{-}13, 15\text{-}21, 24, 26, 29\text{-}31, 33\text{-}46, 48\text{-}49, 51\text{-}57, 59, 61\text{-}62\}$.

The generator matrix of the $AC(63,6,32)$ consists of column types $\{1 - 63\}$. Therefore,

$$D_{AC}(1+48,6,26) \overset{\alpha|}{+} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{+} S_{AC}(63,6,32) \equiv$$

$$AC(210,8,110) \longrightarrow \boxed{DC(45,8,18)_{(18-20)}} \qquad \alpha' = (MM)_o^2$$

$AC(210,8,110)$ + column types $\{5, 33, 65\} \equiv$

$$AC(213,8,112) \longrightarrow \boxed{DC(42,8,16)_{(16-18)}}$$

The generator matrix of the $AC(45,6,24)$ consists of column types $\{3, 5\text{-}7, 9\text{-}13, 15\text{-}21, 24, 26, 29\text{-}31, 33\text{-}46, 48, 51\text{-}52, 54\text{-}57, 59, 61\text{-}62\}$.

$$D_{AC}(1+45,6,24) \overset{\alpha|}{+} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{+} S_{AC}(63,8,32,2) \equiv$$

$$AC(201,8,104) \longrightarrow \boxed{DC(54,8,24)_{(24)}} \qquad \alpha' = (MM)_o^2$$

$$D_{AC}(1+45,6,24) \xrightarrow{\alpha|} F_{AC}(3,2,2) \xrightarrow{\alpha|\alpha'} S_{AC}(48,8,26,2) \equiv$$

$$\Updownarrow \qquad AC(186,8,98) \longrightarrow \overline{|DC(69,8,30)_{(30-32)}} \qquad \alpha' = (MM)_o^2$$

$$D_{AC}(1+40,6,22) \xrightarrow{\alpha|} F_{AC}(3,2,2) \xrightarrow{\alpha|\alpha'} S_{AC}(63,8,32,2) \equiv$$

$$AC(186,8,98) \longrightarrow DC(69,8,30)$$

Where the generator matrix of the AC(40,6,22) consists of column types $\{1\text{-}8,\ 16\text{-}19,\ 21\text{-}24.\ 29\text{-}35,\ 37\text{-}40,\ 45\text{-}47,\ 49\text{-}51,\ 53\text{-}56,\ 61\text{-}62\}$.

$$D_{AC}(1+45,6,24) \xrightarrow{\alpha|} F_{AC}(3,2,2) \xrightarrow{\alpha|\alpha'} S_{AC}(45,8,24,2) \equiv$$

$$AC(182,8,96) \longrightarrow \overline{|DC(72,8,32)_{(32-33)}} \qquad \alpha' = (MM)_o^2$$

$$D_{AC}(1+45,6,24) \xrightarrow{\alpha|} F_{AC}(3,2,2) \xrightarrow{\alpha|\alpha'} S_{AC}(41,8,23,2) \equiv$$

$$\Updownarrow \qquad AC(179,8,95) \longrightarrow \overline{|DC(76,8,33)_{(32-35)}}$$

$$AC(174,8,92) + \text{column types } \{7,\ 8,\ 10,\ 33,\ 64\} \equiv AC(179,8,95)$$

$$D_{AC}(1+45,6,24) \xrightarrow{\alpha|} F_{AC}(3,2,2) \xrightarrow{\alpha|\alpha'} S_{AC}(40,8,22,2) \equiv$$

$$\Updownarrow \qquad AC(178,8,94) \longrightarrow \overline{|DC(77,8,34)_{(33-36)}}$$

$$\begin{cases} AC(174,8,92) + \text{column types } \{7,\ 8,\ 10,\ 33,\ 64\} \equiv AC(179,8,95) \\ AC(179,8,95) \xrightarrow{D} \text{column type } \{126\} \equiv AC(178,8,94). \end{cases}$$

$$AC(174,8,92) + \text{column types } \{6,\ 9,\ 60\} \equiv AC(177,8,94)$$

$$\longrightarrow \overline{|DC(78,8,34)_{(34-36)}}$$

$$D_{AC}(1+45,6,24) \xrightarrow{\alpha|} F_{AC}(3,2,2) \xrightarrow{\alpha|\alpha'} S^*_{AC}(38,8,21,2) \equiv$$

$$AC(176,8,93) \longrightarrow \overline{|DC(79,8,35)_{(35-37)}} \qquad \alpha' = (MM)_o^2$$

$$D_{AC}(1+45,6,24) \xrightarrow{\alpha|} F_{AC}(3,2,2) \xrightarrow{\alpha|\alpha'} S_{AC}(37,8,20,2) \equiv$$

$$AC(175,8,92) \longrightarrow \overline{|DC(80,8,36)_{(36-38)}} \qquad \alpha' = (MM)_o^2$$

$$D^{*}_{AC}(1+37,6,20) \overset{\alpha}{\dashv} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S^{*}_{AC}(51,8,28,2) \equiv$$

$$\Updownarrow \qquad AC(165,8,88) \longrightarrow \boxed{DC(90,8,40)_{(40-43)}} \qquad \alpha' = (MM)^{2}_{o}$$

$$\begin{cases} AC(82,7,44) \xrightarrow{(MM)^{o}_{1} + (MI)^{o}_{1}} AC(164,8,88) \\ AC(164,8,88) + \text{column type } \{128\} \equiv AC(165,8,88) \ . \end{cases}$$

$$D_{AC}(1+37,8,20) \overset{\alpha}{\dashv} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S^{*}_{AC}(49,8,27,2) \equiv$$

$$AC(163,8,87) \longrightarrow \boxed{DC(92,8,41)_{(40-44)}} \qquad \alpha' = (MM)^{2}_{o}$$

$$D_{AC}(1+37,8,20) \overset{\alpha}{\dashv} F_{ac}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S_{AC}(48,8,26,2) \equiv$$

$$AC(162,8,86) \longrightarrow \boxed{DC(93,8,42)_{(41-44)}} \qquad \alpha' = (MM)^{2}_{o}$$

$$D_{AC}(1+37,8,20) \overset{\alpha}{\dashv} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S_{AC}(46,8,25,2) \equiv$$

$$AC(160,8,85) \longrightarrow \boxed{DC(95,6,43)_{(43-46)}} \qquad \alpha' = (MM)^{2}_{o}$$

$$D_{AC}(1+37,6,20) \overset{\alpha}{\dashv} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S_{AC}(45,8,24,2) \equiv$$

$$AC(159,8,84) \longrightarrow \boxed{DC(96,8,44)_{(44-46)}} \qquad \alpha' = (MM)^{2}_{o}$$

$$D_{AC}(1+37,6,20) \overset{\alpha}{\dashv} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S_{AC}(37,8,20,2) \equiv$$

$$AC(151,8,80) \longrightarrow \boxed{DC(104,8,48)_{(48-50)}} \qquad \alpha' = (MM)^{2}_{o}$$

$$AC(147,8,78) + \text{column types } \{169,\ 198,\ 239\} \equiv$$

$$AC(150,8,79) \longrightarrow \boxed{DC(105,8,49)_{(48-51)}}$$

$$AC(150,8,79) \overset{D}{-} \text{column type } \{128\} \equiv$$

$$AC(149,8,78) \longrightarrow \boxed{DC(106,8,50)_{(48.52)}}$$

The generator matrix of the AC(1+7,4,4,1) consists of column types $\{1 - 7\}$.

The generator matrix of the AC(15,4,8) consists of column types $\{1 - 15\}$. Therefore,

$D_{AC}(1+7,4,4,1) \xrightarrow{\alpha|} F_{AC}(15,4,8) \xrightarrow{\alpha|\alpha'} S_{AC}(15,8,8,4) \equiv$

$AC(135,8,72) \longrightarrow DC(120,8,56)$

$AC(135,8,72)$ + column types $\{129, 130, 132, 133, 135, 136, 140, 254, 255\} \equiv AC(144,8,77)$.

$AC(144,8,77)$ + 3 column types $\{158, 173, 179\} \equiv AC(147,8,78)$

$D_{AC}(1+37,6,20) \xrightarrow{\alpha|} F_{AC}(3,2,2) \xrightarrow{\alpha|\alpha'} S_{AC}(33,8,19,2) \equiv AC(147,8,78)$

$AC(147,8,78) \xrightarrow{D}$ column type $\{128\} \equiv$

$AC(146,8,77) \longrightarrow \boxed{DC(109,8,51)_{(50-53)}}$

$AC(147,8,78) \xrightarrow{D}$ column $\{$types 25, 70, 128$\} \equiv$

$AC(144,8,76) \longrightarrow \boxed{DC(111,8,52)_{(50-54)}}$

$AC(135,8,72)$ + column types $\{129, 30, 132, 136, 15\} \equiv$

$AC(140,8,75) \longrightarrow \boxed{DC(116,8,53)_{(52-55)}}$

$AC(140,8,75) \xrightarrow{D}$ column type $\{128\} \equiv AC(139,8,74)$

$\longrightarrow \boxed{DC(116,8,54)_{(53-56)}}$

$D_{AC}(1+7,4,4,1) \xrightarrow{\alpha|} F_{AC}(15,4,8) \xrightarrow{\alpha|\alpha'} S_{AC}(15,4,8,4) \equiv$

$AC(135,8,72) \longrightarrow \boxed{DC(120,8,56)_{(56-59)}}$ $\alpha' = (MM)_o^4$

$D_{AC}(1+3,3,2,1) \xrightarrow{\alpha|} F_{AC}(31,5,16) \xrightarrow{\alpha|\alpha'} S_{AC}(7,8,4,5) \equiv$

$AC(131,8,68) \longrightarrow \boxed{DC(124,8,60)_{(60-61)}}$ $\alpha' = (MM)_o^5$

$D_{AC}(1+31,6,16,1) \xrightarrow{\alpha|} F_{AC}(3,2,2) \xrightarrow{\alpha|\alpha} S_{AC}(31,8,16,2)$

$AC(127,8,64) \longrightarrow \boxed{DC(128,8,63)_{(63)}}$ $\alpha' = (MM)_o^2$

$AC(85,7,46) \xrightarrow{(MM)_1^o} AC(85,8,46,1)$

$\longrightarrow \boxed{DC(170,8,82)_{(80-84)}}$

$$AC(86,7,47) \xrightarrow{(MM)_1^o} AC(86,8,47,1)$$

$$\longrightarrow \boxed{DC(169,8,81)_{(80-83)}}$$

$$AC(95,7,51) \xrightarrow{(MM)_1^o} AC(95,8,51,1)$$

$$\longrightarrow \boxed{DC(160,8,77)_{(76-79)}}$$

$$AC(87,7,47) \xrightarrow{(MM)_1^o} AC(87,8,47,1)$$

$$\longrightarrow \boxed{DC(168,8,81)_{(80-82)}}$$

$$D^*_{AC}(1+108,7,56) \overset{\alpha}{\dashv} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S_{AC}(127,9,64,2) \equiv$$

$$AC(454,9,232) \longrightarrow \boxed{DC(57,9,24)_{(24-26)}}$$

$$D_{AC}(1+45,6,24) \overset{\alpha}{\dashv} F_{AC}(7,3,4) \overset{\alpha|\alpha'}{\dashv} S_{AC}(63,9,32,3) \equiv$$

$$AC(380,9,200) \longrightarrow \boxed{DC(126,9,56)_{(56-61)}} \qquad \alpha' = (MM)_o^3$$

$$D^*_{AC}(1+85,7,44) \overset{\alpha}{\dashv} F_{ac}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S_{AC}(127,9,64,2) \equiv$$

$$AC(376,9,196) \longrightarrow \boxed{DC(135,9,60)_{(60.64)}}$$

$$D^*_{AC}(1+79,7,42) \overset{\alpha}{\dashv} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S_{AC}(63,9,32,3) \equiv$$

$$AC(303,9,158) \longrightarrow \boxed{DC(208,9,98)_{(98-102)}}$$

$$D^*_{AC}(1+75,7,40) \overset{\alpha}{\dashv} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S_{AC}(63,9,32,3) \equiv$$

$$AC(292,9,152) \longrightarrow \boxed{DC(219,9,104)_{(104,108)}}$$

$$D^*_{AC}(1+71,7,38) \overset{\alpha}{\dashv} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S_{AC}(63,9,32,3) \equiv$$

$$AC(279,9,146) \longrightarrow \boxed{DC(232,9,110)_{(110-114)}}$$

$$D^*_{AC}(1+67,7,36) \overset{\alpha}{\dashv} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{\dashv} S_{AC}(63,9,32,3) \equiv$$

$$AC(270,9,140) \longrightarrow \boxed{DC(241,9,116)_{(116-120)}}$$

$$D_{AC}(1+7,4,4) \overset{\alpha}{\dashv} F_{AC}(31,5,16) \overset{\alpha|\alpha'}{\dashv} S_{AC}(7,9,4,6) \equiv$$

$$AC(263,9,136) \rightarrow \boxed{DC(248,9,120)_{(120-123)}} \quad \alpha' = (MM)_1^5$$

$$D_{AC}(1+3,3,2) \overset{\alpha}{\dashv} F_{AC}(63,6,32) \overset{\alpha|\alpha'}{\dashv} S_{AC}(7,9,4,4) \equiv$$

$$AC(259,9,132) \rightarrow \boxed{DC(252,9,124)_{(124-125)}} \quad \alpha' = (MM)_3^3$$

$$D_{AC}(1+3,3,2) \overset{\alpha}{\dashv} F_{AC}(56,6,30) \overset{\alpha|\alpha'}{\dashv} S_{AC}(3,9,2,7) \equiv$$

$$AC(227,9,120,1) \rightarrow \boxed{DC(284,9,136)_{(136-140)}} \quad \alpha' = (MM)_1^6$$

$$D_{AC}(213,8,112) \xrightarrow{(MM)_1^o} AC(213,9,112,1)$$

$$\rightarrow \boxed{DC(298,9,144)_{(144-147)}}$$

$$D_{AC}(1+48,6,26) \overset{\alpha}{\dashv} F_{AC}(3,3,2,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(63,9,32,3,1) \equiv$$

$$AC(210,9,110) \rightarrow \boxed{DC(301,9,146)_{(146-148)}}$$

$$D_{AC}(1+45,6,24) \overset{\alpha}{\dashv} F_{AC}(3,3,2,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(63,9,32,3) \equiv$$

$$AC(201,9,104,1) \rightarrow \boxed{DC(310,9,152)_{(152-153)}}$$

$$D_{AC}(1+45,6,24) \overset{\alpha}{\dashv} F_{AC}(3,3,2,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(48,9,26,3) \equiv$$

$$AC(186,9,98,1) \rightarrow \boxed{DC(325,9,158)_{(158-160)}}$$

$$D_{AC}(1+45,6,24) \overset{\alpha}{\dashv} F_{AC}(3,3,2,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(45,9,24,3) \equiv$$

$$\Updownarrow \qquad AC(183,9,96,1) \rightarrow \boxed{DC(328,9,160)_{(160-162)}}$$

$$D_{AC}(1+3,3,2) \overset{\alpha}{\dashv} F_{AC}(45,6,24) \overset{\alpha|\alpha'}{\dashv} S_{AC}(3,9,2,7) \equiv$$

$$AC(183,9,96,1)$$

$$D_{AC}(1+45,6,24) \overset{\alpha}{\dashv} F_{AC}(3,3,2,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(41,9,23,3) \equiv$$

$$AC(179,9,95,1) \rightarrow \boxed{DC(332,9,161)_{(161-164)}} \quad \alpha' = (MM)_o^3$$

$$D_{AC}(1+45,6,24) \overset{\alpha}{\dashv} F_{AC}(3,3,2,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(40,9,22,3) \equiv$$

$$AC(178,9,94,1) \rightarrow \boxed{DC(331,9,162)_{(162-164)}}$$

$$D_{AC}(1+45,6,24) \xrightarrow{\alpha|} F_{AC}(3,3,2,1) \xrightarrow{\alpha|\alpha'} S_{AC}(38,9,21,3) \equiv$$

$$AC(176,9,93,1) \rightarrow \boxed{DC(335,9,163)_{(163-166)}}$$

$$D_{AC}(1+45,6,24) \xrightarrow{\alpha|} F_{AC}(3,3,2,1) \xrightarrow{\alpha|\alpha'} S_{AC}(37,9,20,3) \equiv$$

$$AC(175,9,92,1) \rightarrow \boxed{DC(336,9,164)_{(164-166)}}$$

$$AC(166,8,88) \xrightarrow{(MM)_1^o} AC(166,9,88,1)$$

$$\rightarrow \boxed{DC(345,9,168)_{(168-171)}}$$

$$D_{AC}(1+40,6,22) \xrightarrow{\alpha|} F_{AC}(3,3,2,1) \xrightarrow{\alpha|\alpha'} S_{AC}(40,9,22,3) \equiv$$

$$AC(163,9,88,1) \rightarrow \boxed{DC(348,9,168)_{(168-172)}}$$

$$D_{AC}(1+3,3,2,1) \xrightarrow{\alpha|} F_{AC}(40,6,22) \xrightarrow{\alpha|\alpha'} S_{AC}(3,9,2,7) \equiv$$

$$AC(163,9,88,1)$$

$$AC(162,8,86) \xrightarrow{(MM)_1^o} AC(162,9,86,1)$$

$$\rightarrow \boxed{DC(349,9,170)_{(170-173)}}$$

$$AC(160,8,85) \xrightarrow{(MM)_1^o} AC(160,9,85,1)$$

$$\rightarrow \boxed{DC(351,9,171)_{(171-174)}}$$

$$AC(159,8,84) \xrightarrow{(MM)_1^o} AC(159,9,84,1)$$

$$\rightarrow \boxed{DC(352,9,172)_{(172-175)}}$$

$$D_{AC}(1+3,3,2,1) \xrightarrow{\alpha|} F_{AC}(37,6,20) \xrightarrow{\alpha|\alpha'} S_{AC}(3,9,2,7) \equiv$$

$$AC(151,9,80,1) \rightarrow \boxed{DC(360,9,176)_{(176-178)}} \quad \alpha' = (MM)_1^6$$

$$AC(151,8,80) \xrightarrow{(MM)_1^o} AC(151,9,80,1)$$

$$AC(150,8,79) \xrightarrow{(MM)_1^o} AC(150,9,79,1)$$

$$\rightarrow \boxed{DC(361,9,177)_{(177-179)}}$$

$$AC(149,8,78) \xrightarrow{(MM)_1^o} AC(149,9,78,1)$$

$$\longrightarrow \boxed{DC(362,9,178)_{(178-180)}}$$

$$AC(146,8,77) \xrightarrow{(MM)_1^o} AC(146,9,77,1)$$

$$\longrightarrow \boxed{DC(365,9,179)_{(179-181)}}$$

$$AC(144,8,76) \xrightarrow{(MM)_1^o} AC(144,9,76,1)$$

$$\longrightarrow \boxed{DC(367,9,180)_{(180-182)}}$$

$$AC(140,8,75) \xrightarrow{(MM)_1^o} AC(140,9,75,1)$$

$$\longrightarrow \boxed{DC(371,9.181)_{(181-184)}}$$

$$AC(139,8,74) \xrightarrow{(MM)_1^o} AC(139,9,74,1)$$

$$\longrightarrow \boxed{DC(372,9,182)_{(192-184)}}$$

$$AC(135,8,72) \xrightarrow{(MM)_1^o} AC(135,9,72,1)$$

$$\longrightarrow \boxed{DC(376,9,184)_{(184-187)}}$$

$$D_{AC}(1+3,3,2,1)^{\alpha} \; F_{AC}(31,6,16)^{\alpha\,\alpha'} \; S_{AC}(7,9,4,6) \equiv$$

$$AC(131,9,68,1) \longrightarrow \boxed{DC(380,9,188)_{(188-189)}}$$

$$AC(95,7,51) \xrightarrow{(MM)_2^o} AC(95,9,51,2)$$

$$\longrightarrow \boxed{DC(416,9,205)_{(205-207)}}$$

$$AC(87,7,47) \xrightarrow{(MM)_2^o} AC(87,9,47,2)$$

$$\longrightarrow \boxed{DC(424,9,209)_{(208-210)}}$$

$$AC(85,7,46) \xrightarrow{(MM)_2^o} AC(85,9,46,2)$$

$$\longrightarrow \boxed{DC(429,9,210)_{(210-212)}}$$

$$D_{AC}(1+108,7,76) \overset{\alpha}{\dashv} F_{AC}(3,3,2,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(127,10,64,3) \equiv$$

$$AC(454,10,232,1) \rightarrow \boxed{DC(569,10,280)_{(280-282)}} \quad \alpha' = (MM)_o^3$$

$$D_{AC}(1+82,7,44) \overset{\alpha}{\dashv} F_{AC}(3,8,2,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(127,10,64,3) \equiv$$

$$AC(376,10,196,1) \rightarrow \boxed{DC(647,10,316)_{(316-320)}}$$

$$D_{AC}(1+7,4,4,1) \overset{\alpha}{\dashv} F_{AC}(31,6,16) \overset{\alpha|\alpha'}{\dashv} S_{AC}(15,10,8,6) \equiv$$

$$AC(263,10,136,1) \rightarrow \boxed{DC(760,10,376)_{(376-379)}}$$

$$D_{AC}(1+79,7,42) \overset{\alpha}{\dashv} F_{AC}(3,3,2,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(63,10,32,4) \equiv$$

$$AC(303,10,158,1) \rightarrow \boxed{DC(720,10,354)}$$

$$D_{AC}(1+71,7,38) \overset{\alpha}{\dashv} F_{AC}(3,3,2,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(63,10,32,4) \equiv$$

$$AC(279,10,146,1) \rightarrow \boxed{DC(744,10,366)_{(366-370)}}$$

$$D_{AC}(1+67,7,36) \overset{\alpha}{\dashv} F_{AC}(3,3,2,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(63,10,32,4) \equiv$$

$$AC(270,10,140,1) \rightarrow \boxed{DC(753,10,372)_{(372-376)}}$$

$$D_{AC}(1+3,2,3,1) \overset{\alpha}{\dashv} F_{AC}(63,7,32) \overset{\alpha|\alpha'}{\dashv} S_{AC}(7,10,4,7) \equiv$$

$$AC(259,10,132,1) \rightarrow \boxed{DC(764,10,380)_{(380-381)}}$$

$$D_{AC}(1+31,6,16) \overset{\alpha}{\dashv} F_{AC}(7,4,4,1) \overset{\alpha|\alpha'}{\dashv} S_{AC}(31,10,16,5) \equiv$$

$$AC(255,10,128,1) \rightarrow \boxed{DC(768,10,384)_{(384)}}$$

$$D_{AC}(1+3,3,2,1) \overset{\alpha}{\dashv} F^*_{AC}(56,7,30) \overset{\alpha|\alpha'}{\dashv} S_{AC}(3,10,2,8) \equiv$$

$$AC(227,10,120,1) \rightarrow \boxed{DC(796,10,392)_{(392-396)}}$$

$$AC(213,8,112) \xrightarrow{(MM)_2^o} AC(213,10,112,2)$$

$$\rightarrow \boxed{DC(810,10,400)_{(400-403)}}$$

$$D_{AC}(1+48,6,26) \overset{\alpha}{\dashv} F_{AC}(3,4,2,2) \overset{\alpha|\alpha'}{\dashv} S_{AC}(63,10,32,4) \equiv$$

$$AC(210,10,110,2) \rightarrow \boxed{DC(813,10,402)_{(402-404)}}$$

$$D_{AC}(1+45,6,24) \overset{\alpha|}{+} F_{AC}(3,4,2,2) \overset{\alpha|\alpha'}{+} S_{AC}(63,10,32,4) \equiv$$

$$AC(201,10,104,2) \longrightarrow \boxed{DC(822,10,408)_{(408-409)}}$$

$$D_{AC}(1+45,6,24) \overset{\alpha|}{+} F_{AC}(3,4,2,2) \overset{\alpha|\alpha'}{+} S_{AC}(48,10,26,4) \equiv$$

$$AC(186,10,98,2) \longrightarrow \boxed{DC(837,10,414)_{(414-416)}}$$

$$AC(183,9,96) \xrightarrow{(MM)_1^o} AC(183,10,96,1)$$

$$\longrightarrow \boxed{DC(840,10,416)_{(416-418)}}$$

$$D_{AC}(1+45,6,24) \overset{\alpha|}{+} F_{AC}(3,4,2,2) \overset{\alpha|\alpha'}{+} S_{AC}(44,10,24,4) \equiv$$

$$AC(182,10,96,2) \longrightarrow \boxed{DC(841,10,416)_{(416-418)}}$$

$$D_{AC}(1+45,6,24) \overset{\alpha|}{+} F_{AC}(3,4,2,2) \overset{\alpha|\alpha'}{+} S_{AC}(41,10,23,4) \equiv$$

$$AC(179,10,95,2) \longrightarrow \boxed{DC(844,10,417)_{(417-420)}}$$

$$D_{AC}(1+45,6,24) \overset{\alpha|}{+} F_{AC}(3,4,2,2) \overset{\alpha|\alpha'}{+} S_{AC}(40,10,22,4) \equiv$$

$$AC(178,10,94,2) \longrightarrow \boxed{DC(854,10,418)_{(418-420)}}$$

$$D_{AC}(1+45,6,24) \overset{\alpha|}{+} F_{AC}(3,4,2,2) \overset{\alpha|\alpha'}{+} S_{AC}(38,10,21,4) \equiv$$

$$AC(176,10,93,2) \longrightarrow \boxed{DC(847,10,419)_{(419-422)}}$$

$$D_{AC}(1+45,6,24) \overset{\alpha|}{+} F_{AC}(3,4,2,2) \overset{\alpha|\alpha'}{+} S_{AC}(37,10,20,4) \equiv$$

$$AC(175,10,92,2) \longrightarrow \boxed{DC(848,10,420)_{(420-422)}}$$

$$AC(166,8,88) \xrightarrow{(MM)_2^o} AC(166,10,88,2)$$

$$\longrightarrow \boxed{DC(857,10,424)_{(424-427)}}$$

$$AC(162,8,86) \xrightarrow{(MM)_2^o} AC(162,10,86,2)$$

$$\longrightarrow \boxed{DC(861,10,426)_{(426-429)}}$$

$$AC(160,8,85) \xrightarrow{(MM)_2^o} AC(160,8,85,2)$$

$$\longrightarrow \boxed{DC(863,10,427)}_{(427-430)}$$

$$AC(159,8,84) \xrightarrow{(MM)_2^o} AC(159,10,84,2)$$

$$\longrightarrow \boxed{DC(864,10,428)}_{(428-431)}$$

$$D_{AC}(1+3,3,2,1) \xrightarrow{\alpha} F_{AC}(37,7,20) \xrightarrow{\alpha|\alpha'} S_{AC}(3,10,2,8) \equiv$$

$$\Updownarrow \qquad AC(151,10,80,2) \longrightarrow \boxed{DC(872,10,432)}_{(432-434)}$$

$$AC(151,8,80) \xrightarrow{(MM)_2^o} AC(151,10,80,2)$$

$$AC(150,8,79) \xrightarrow{(MM)_2^o} AC(150,10,79,2)$$

$$\longrightarrow \boxed{DC(873,10,433)}_{(433-435)}$$

$$AC(149,8,78) \xrightarrow{(MM)_2^o} AC(149,8,78,2)$$

$$\longrightarrow \boxed{DC(874,10,434)}_{(434-436)}$$

$$AC(146,8,77) \xrightarrow{(MM)_2^o} AC(146,10,77,2)$$

$$\longrightarrow \boxed{DC(877,10,435)}_{(435-437)}$$

$$AC(144,8,76) \xrightarrow{(MM)_2^o} AC(144,10,76,2)$$

$$\longrightarrow \boxed{DC(879,10,430)}_{(435-438)}$$

$$AC(140,8,75) \xrightarrow{(MM)_2^o} AC(140,8,75,2)$$

$$\longrightarrow \boxed{DC(883,10,437)}_{(437-440)}$$

$$AC(139,8,74) \xrightarrow{(MM)_2^o} AC(139,10,74,2)$$

$$\longrightarrow \boxed{DC(884,10,438)}_{(438-440)}$$

$$AC(135,8,72) \xrightarrow{(MM)_2^o} AC(135,10,72,2) \longrightarrow \boxed{DC(376,10,184)}_{(184-187)}$$

$$AC(95,7,51) \xrightarrow{(MM)_3^o} AC(95,10,51,3) \longrightarrow \boxed{DC(928,10,461)}_{(461-463)}$$

$$AC(87,7,47) \xrightarrow{(MM)_3^o} AC(87,10,47,3) \longrightarrow \boxed{DC(936,10,465)}_{(465-466)}$$

$$AC(86,7,46) \xrightarrow{(MM)_3^o} AC(86,7,46,3) \longrightarrow \boxed{DC(937,10,466)}_{(466-467)}$$

CHAPTER VIII

GENERALISATION OF THE LINEAR BINARY ANTICODES TO MULTI-LEVEL (Q-ARY) ANTICODES

8.1 The q-ary Anticode Analysis and Construction

Construction methods and some related properties of linear binary anticodes have been given in previous sections. The development of construction methods and some properties of linear anticodes whose elements are chosen from the general finite field GF(q), are studied in the next sections. The development is parallel to that of the linear binary anticode, and will therefore be somewhat condensed. However, particular emphasis has been given to 3-ary linear anticodes.

8.1.1 *Linear q-ary Parent Anticode Analysis*

The linear q-ary parent anticode PAC$[q^k-1,\ k,\ (q-1)\ q^{k-1}]$ is the subspace of the vector space of all n(= q^k-1) -tuple vectors. This is generated by the k . (q^k-1) matrix of its linearly independent generators. The generator matrix $[G_q]_{PAC}$ has all k-tuple vectors over GF(q) as its columns. Thus any row and column of the linear parent anticode contains the same number ($q^k/q = q^{k-1}$) of the non-zero symbols of the GF(q). However, the number of zero symbols in each row is one less than the number of the (q-1) other symbols. Therefore the generator matrix of the linear parent anticode can be partitioned as follows:

$$[G_q]_{PAC} = \left[\begin{array}{c|c|c|c|c|c} G_q^o & \begin{matrix}0\\ \vdots\\ 0\end{matrix} & G_q^o & \begin{matrix}0\\ \vdots\\ 0\end{matrix} & G_q^o & \begin{matrix}0\\ \vdots\\ 0\end{matrix}\ \ G_q^o \\ \hline 0 & \alpha_o = 1 & \alpha_1 & \cdots\cdots & & \alpha_{q-2} \end{array}\right]$$

where any $\alpha_i \in GF(q)$ stands for a row vector in which all q^{k-1} elements are α_i, and the zero element in the far left position stands for a $q^{k-1} - 1$ all-zero element vector. Also, the element α_o is the unity element of GF(q). Every $[G_q^o]$ forms the generator matrix of the $[n = q^{k-1} - 1, k-1, (q-1)\, q^{k-2}]$ parent anticode. The generator matrix $[G_q^o]$ can be further partitioned into the $[q^{k-2} - 1, k-2, (q-1)\, q^{k-3}]$ parent anticodes.

The row space of the $[G_q^o]$ matrices in the $[G_q]_{PAC}$, including the adjacent all-zero columns (except for first $[G_q^o]$ which does not include the all-zeros column), are spanned with respect to all the α_i last row segments, forming the row space of the $[G_q^o]$ and (q-1) translates as is shown in Figure (8-1). Therefore the columns of the q-ary parent anticode have the property that any q^i, (i = 1, 2, ..., k) top elements of each column can be divided into q equal parts, each part containing q^{i-1} elements and formed by adding $c\,\alpha_j$ to the first top q^{i-1} elements of the divided column, where $c \in GF(q)$ and α_j is the last element of the related column in the $[G_q]_{PAC}$ of the divided column. This leads to the same Q-I property of the linear binary parent anticode with only slight differences. This property is denoted by $Q\text{-}I_q$.

This can be illustrated by an example, considering the parent anticode of the parameters [26, 3, 18] over GF(3). The elements of $GF(3) = \{0, 1, \alpha\}$ are generated by the primitive polynomial $P(x) = x + 1$, where α is the primitive element, and also* $\alpha^2 = 1$. However, GF(3) is isomorphic to the field of integers $\{0, 1, 2\}$ mod-3, thus operations can be carried out over the field of integers mod-3.

---

* Note: The multiplication of the elements $\alpha_i \in GF(q)$, i = 1, 2, ..., q-2 in Figure (8-1) is performed considering $\alpha^{q-1} - 1 = 0$ for all $\alpha \in GF(q)$.

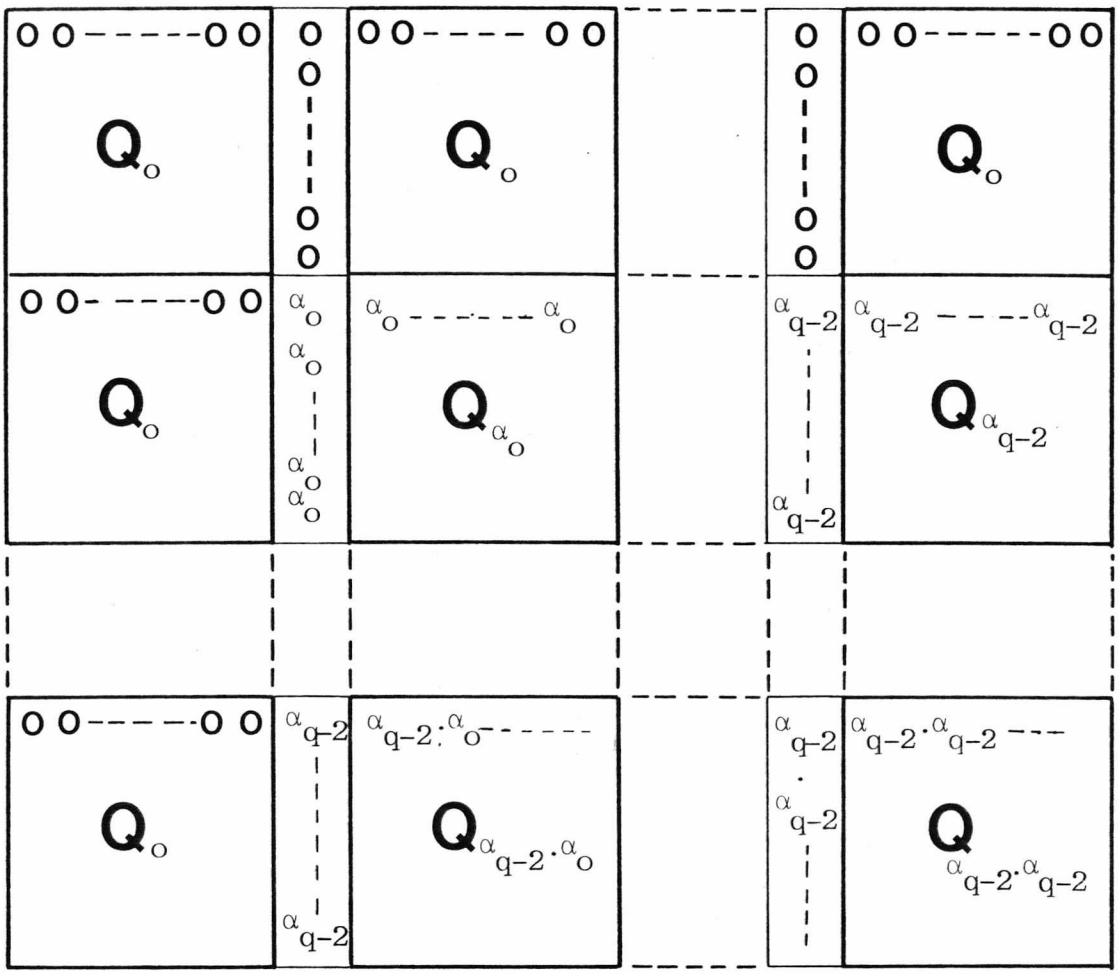


Figure (8-1) : General Partitioned Form of the q-ary Parent Anticode.

The generator matrix $[G_3]_{PAC} = [M_3]$ of a linear 3-ary parent anticode has all $(q^k - 1)$ possible k-tuple vectors over GF(q) as its columns. The column i is the ternary representation of the number i, where these numbers are ordered, increasing from left to right.

$$[M_3] = [G_3]_{PAC} = [(1)_3 \cdot (2)_3, \ldots\ldots, (q^k - 1)_3] \quad ,$$

hence,

$$[G_3]_{PAC} = \left[\begin{array}{ccccccccccccccccccccccccc} 1&2&0&1&2&0&1&2 & 0 & 1&2&0&1&2&0&1&2 & 0 & 1&2&0&1&2&0&1&2 \\ 0&0&1&1&1&2&2&2 & 0 & 0&0&1&1&1&2&2&2 & 0 & 0&0&1&1&1&2&2&2 \\ 0&0&0&0&0&0&0&0 & 1 & 1&1&1&1&1&1&1&1 & 2 & 2&2&2&2&2&2&2&2 \end{array}\right]$$

The row space of $[M_3]$, which is the 3-ary parent anticode, is formed by multiplying on the left by the matrix $[M_3^O]^T$, where $[M_3^O]^T$ is the transpose of matrix $[M_3]$ including an all-zeros element first row. Therefore

$$[M_3^O]^T \cdot [M_3] = PAC\left[q^k - 1 = 3^3 - 1 = 26,\ k = 3,\ \delta = d = (q-1)q^{k-1} = 18\right].$$

The PAC(26, 3, 18) is shown in Figure (8-2).

0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0
1 2 0 1 2 0 1 2 0	1 2 0 1 2 0 1 2 0	1 2 0 1 2 0 1 2 0
2 1 0 2 1 0 2 1 0	2 1 0 2 1 0 2 1 0	2 1 0 2 1 0 2 1 0
0 0 1 1 1 2 2 2 0	0 0 1 1 1 2 2 2 0	0 0 1 1 1 2 2 2 0
1 2 1 2 0 2 0 1 0	1 2 1 2 0 2 0 1 0	1 2 1 2 0 2 0 1 0
2 1 1 0 2 2 1 0 0	2 1 1 0 2 2 1 0 0	2 1 1 0 2 2 1 0 0
0 0 2 2 2 1 1 1 0	0 0 2 2 2 1 1 1 0	0 0 2 2 2 1 1 1 0
1 2 2 0 1 1 2 0 0	1 2 2 0 1 1 2 0 0	1 2 2 0 1 1 2 0 0
2 1 2 1 0 1 0 2 0	2 1 2 1 0 1 0 2 0	2 1 2 1 0 1 0 2 0
0 0 0 0 0 0 0 0 0	1 1 1 1 1 1 1 1 1	2 2 2 2 2 2 2 2 2
1 2 0 1 2 0 1 2 1	2 0 1 2 0 1 2 0 1	0 1 2 0 1 2 0 1 2
2 1 0 2 1 0 2 1 1	0 2 1 0 2 1 0 2 1	1 0 2 1 0 2 1 0 2
0 0 1 1 1 2 2 2 1	1 1 2 2 2 0 0 0 1	2 2 0 0 0 1 1 1 2
1 2 1 2 0 2 0 1 1	2 0 2 0 1 0 1 2 1	0 1 0 1 2 1 2 0 2
2 1 1 0 2 2 1 0 1	0 2 2 1 0 0 2 1 1	1 0 0 2 1 1 0 2 2
0 0 2 2 2 1 1 1 1	1 1 0 0 0 2 2 2 1	2 2 1 1 1 0 0 0 2
1 2 2 0 1 1 2 0 1	2 0 0 1 2 2 0 1 1	0 1 1 2 0 0 1 2 2
2 1 2 1 0 1 0 2 1	0 2 0 2 1 2 1 0 1	1 0 1 0 2 0 2 1 2
0 0 0 0 0 0 0 0 0	2 2 2 2 2 2 2 2 2	1 1 1 1 1 1 1 1 1
1 2 0 1 2 0 1 2 2	0 1 2 0 1 2 0 1 2	1 2 0 1 2 0 1 2 0
2 1 0 2 1 0 2 1 2	1 0 2 1 0 2 1 0 1	0 2 1 0 2 1 0 2 1
0 0 1 1 1 2 2 2 2	2 2 0 0 0 1 1 1 2	1 1 2 2 2 0 0 0 1
1 2 1 2 0 2 0 1 2	0 1 0 1 2 1 2 0 2	2 0 2 0 1 0 1 2 1
2 1 1 0 2 2 1 0 2	1 0 0 2 1 1 0 2 2	0 2 2 1 0 0 2 1 1
0 0 2 2 2 1 1 1 2	2 2 1 1 1 0 0 0 2	1 1 0 0 0 2 2 2 1
1 2 2 0 1 1 2 0 2	0 1 1 2 0 0 1 2 2	2 0 0 1 2 2 0 1 1
2 1 2 1 0 1 0 2 2	1 0 1 0 2 0 2 1 2	0 2 0 2 1 2 1 0 1

Figure (8-2) : Partitioned Version of the PAC(26,3,18)

The truncated elements of the generator matrix $[G_3]_{PAC}$ form three copies of the generator matrix $[G_3^o]_{PAC}$ of the PAC(8, 2, 6). Each of the $[G_3^o]$ spans the row space of the PAC(8, 2, 6), forming the three top copies of the PAC(8, 2, 6) in Figure (8-2); then any of these PAC(8, 2, 6) is translated by repeated addition of its respective row in the $[G_3]_{PAC}$.

If every PAC(8. 2. 6), generated by $[G_3^o]_{AC}$, is denoted by Q_o, then every translate is either a Q_o, Q_1 or Q_2, where Q_1 and Q_2 are respectively formed by adding a "1" or a "2" to the elements of Q_o, where the addition is performed mod-3; i.e. $Q_4 \equiv Q_1$ mod-3. Therefore the PAC(26, 3, 18) can be put in the form of Figure (8-3a), or equivalently in the form of Figure (8-3b), where each zero, one or two in Figure (8-3a) and (8-3b) respectively represents a 9-tuple all-zeros, -ones or -twos column.

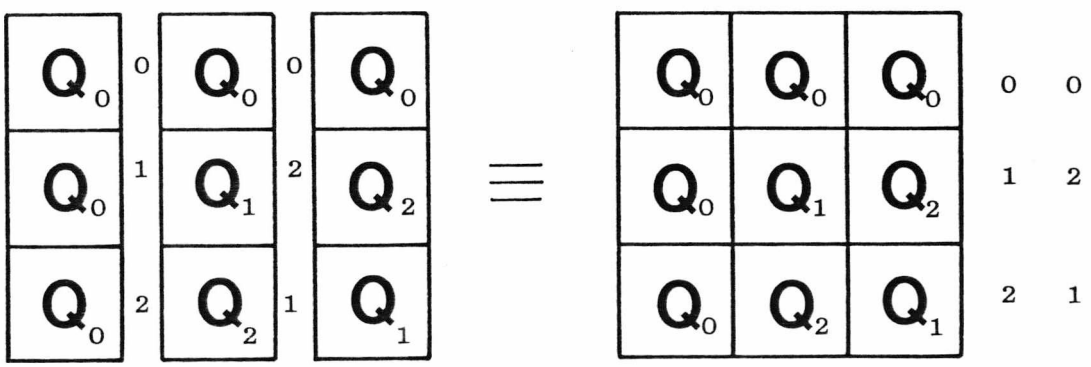


Figure (8-3a) - Truncated Form of the PAC(26,3,18).

Figure (8.3b) - Equivalent Translated Form of the PAC(26,3,18).

The set of the zeros, ones and twos, form the AC(2, 3, 2, 2), which in fact is the AC(2, 3, 2) $\xrightarrow{(MM)_o^2}$ AC(2, 3, 2, 2). Discarding this anticode in Figure (8.3b), and mapping Q_o, Q_1, Q_2 one-to-one on to 0, 1, and 2 respectively, i.e.

$$\alpha : \quad Q_0 \longleftrightarrow 0 \qquad Q_1 \longleftrightarrow 1 \qquad Q_2 \longleftrightarrow 2$$

The set of images in Figure (8-4), excluding the first all-zeros column, forms the linear 3-ary PAC(2, 1, 2).

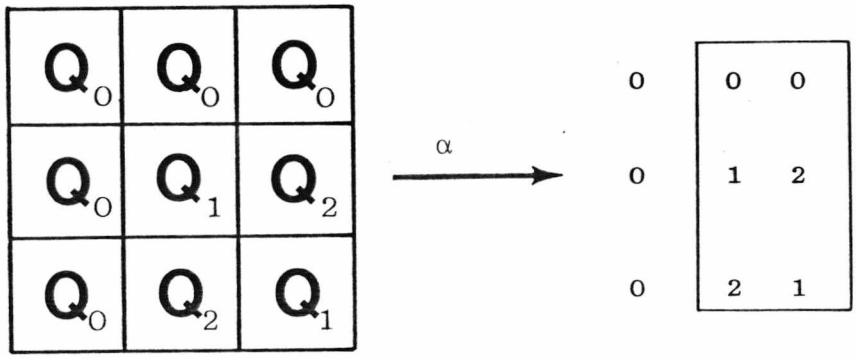


Figure (8-4) : Set of Images and the Resulting PAC(2,1,2)

8.1.2 *Iterative Image Stacking Process for q-ary Anticode Construction.*

The process of obtaining an anticode of large dimension from an anticode of smaller dimension is performed in a similar way to the binary case as in Section 6.4.6.1. Since each subprocess is performed by a "one-to-one mapping onto", thus the construction process can be reversed. This reverse process of the q-ary anticode construction is called iterative image stacking. In the iterative image stacking process the same symbols and formulation are used, and the subprocesses are parallel to those of the binary case, so it will only be explained briefly, except for those aspects which are of particular importance in the q-ary case. The construction of anticodes using this process can be illustrated by the construction of the AC(48, 4, 33) from AC(4, 2, 3).

The domain anticode $D_{AC}(1 + m, k, \delta, \psi)$ is formed by adding an all zero column to the optimum AC(4, 2, 3) to get $D_{AC}(1 + 4, 2, 3)$. The generator matrix of the $D_{AC}(1 + 4, 2, 3)$ has the form:

$$\left[\mathbf{G}_{q}\right]_{D_{AC}} = \begin{bmatrix} 0 & 0 & 1 & 2 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix} .$$

The rowspace of this matrix is the $D_{AC}(1 + 4, 2, 3)$ as shown in Figure (8-5).

0	0	0	0	0
0	0	1	2	1
0	0	2	1	2
0	1	1	1	0
0	1	2	0	1
0	1	0	2	2
0	2	2	2	0
0	2	0	1	1
0	2	1	0	2

Figure (8-5) : Domain Anticode (1 + 4, 2, 3)

The optimum AC(8, 2, 6) is chosen to be the first mapped anticode, and then

$$F_{AC}(m_r, k_r, \delta_r, \psi_r) \equiv F_{AC}(8, 2, 6) .$$

A mapping α maps the elements $\{ 0, 1, 2 \}$ of the $D_{AC}(1+ 4, 2, 3)$ one-to-one onto the set $\{ Q_0, Q_1, Q_2 \}$ of the image anticode of Figure (8-6), such that

$$\alpha : \quad 0 \longleftrightarrow Q_0$$
$$1 \longleftrightarrow Q_1$$
$$2 \longleftrightarrow Q_2$$

where $Q_o \equiv F_{AC}(8, 2, 6)$ and Q_1, Q_2 are formed by adding the element one or two to the elements of Q_o respectively; the addition is performed mod-3; i.e. $Q_4 \equiv Q_1$.

$$
\begin{array}{cc}
 & \begin{matrix} 0 \\ 30 \end{matrix} \\
\delta_i \longleftarrow & Q_0\ Q_0\ Q_0\ Q_0\ Q_0 \\
\begin{matrix} 24 \\ 27 \end{matrix} & Q_0\ Q_0\ Q_1\ Q_2\ Q_1 \\
\begin{matrix} 24 \\ 27 \end{matrix} & Q_0\ Q_0\ Q_2\ Q_1\ Q_2 \\
\begin{matrix} 24 \\ 27 \end{matrix} & Q_0\ Q_1\ Q_1\ Q_1\ Q_0 \\
\begin{matrix} 24 \\ 27 \end{matrix} & Q_0\ Q_1\ Q_2\ Q_0\ Q_1 \\
\begin{matrix} 24 \\ 27 \end{matrix} & Q_0\ Q_1\ Q_0\ Q_2\ Q_2 \\
\begin{matrix} 24 \\ 27 \end{matrix} & Q_0\ Q_2\ Q_2\ Q_2\ Q_0 \\
\begin{matrix} 24 \\ 27 \end{matrix} & Q_0\ Q_2\ Q_0\ Q_1\ Q_1 \\
\begin{matrix} 24 \\ 27 \end{matrix} & Q_0\ Q_2\ Q_1\ Q_0\ Q_2
\end{array}
$$

Figure (8-6) : Image Anticode (40,4,30)

Hence,

$$D_{AC}(1 + m_d, k_d, \delta_d, \psi_d) \overset{\alpha|}{+} F_{AC}(m_r, k_r, \delta_r, \psi_r)$$

$$\equiv I_{AC}(m_i, k_i, \delta_i, \psi_i) .$$

The image anticode has parameters

$$
\begin{cases}
m_i = (m_d + 1) m_r = 5 \cdot 8 = 40 \\
k_i = k_d + k_r = 2 + 2 = 4 \\
\delta_i = (1 + m_d)\delta_r = 5.6 = 30 \\
\psi_i = \psi_d + \psi_r = 0
\end{cases}
$$

Therefore:

$$D_{AC}(1 + 4, 2, 3) \overset{\alpha|}{+} F_{AC}(8, 2, 6) \equiv I_{AC}(40, 4, 30) .$$

The first row of the image anticode of Figure (8-6) is formed by all Q_o elements, therefore the rows corresponding to these Q_o's, except the first all-zero element row , have the biggest weight $\delta_i = (1 + m_d)\delta_r$. Alternatively, the image anticode has the weight attribute $W(\delta_i, 2, 8) = 8$. The rest of the rows, due to Q_o, Q_1, Q_2 combinations have

$$\delta_i \geq \text{max. weight} \geq m_r \cdot \delta_d$$

Also, the minimum distance d of the domain anticode should be considered when the δ_i is calculated. The smaller d is, the larger the number of Q_o's in the image anticode, and therefore the larger δ_i is.

The AC(8, 2, 6) is chosen as the second domain anticode, and the second mapped anticode is then obtained by the process

$$S_{DAC}(8, 2, 6) \xrightarrow{(MM)_o^2} S_{AC}(8, 4, 6, 2) \quad .$$

The $S_{AC}(8, 4, 6, 2)$ has the weight attribute $W(0, 1, 9) = 9$ which is opposite to the attribute $W(\delta_i, 2, 8) = 8$ of the image anticode. The concatenation of $I_{AC}(40, 4, 30)$, and $S_{AC}(8, 4, 6, 2)$ gives the optimum AC(48, 4, 33), which is shown in Figure (7-7). Deleting the resultant AC(48, 4, 33) from PAC(80, 4, 54) results in optimum DC(32, 4, 21). The whole process can be formulated as:

$$D_{AC}(1 + 4, 2, 3) \xrightarrow{\alpha|} F_{AC}(8, 2, 6) \xrightarrow{\alpha|\alpha'} S_{AC}(8, 4, 6, 2) \equiv AC(48, 4, 33)$$

$$\longrightarrow DC(32, 4, 21)_{21} \quad .$$

The subscript 21 is the upper bound value of d for the linear block code with n = 32, k = 4, by the Griesmer bound.

$$
\begin{array}{lll}
{}^{0}_{30} & Q_0\ Q_0\ Q_0\ Q_0\ Q_0 & 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \\
{}^{30}_{33} & Q_0\ Q_0\ Q_1\ Q_2\ Q_1 & 1\ 2\ 0\ 1\ 2\ 0\ 1\ 2 \\
{}^{30}_{33} & Q_0\ Q_0\ Q_2\ Q_1\ Q_2 & 2\ 1\ 0\ 2\ 1\ 0\ 2\ 1 \\
{}^{30}_{33} & Q_0\ Q_1\ Q_1\ Q_1\ Q_0 & 0\ 0\ 1\ 1\ 1\ 2\ 2\ 2 \\
{}^{30}_{33} & Q_0\ Q_1\ Q_2\ Q_0\ Q_1 & 1\ 2\ 1\ 2\ 0\ 2\ 0\ 1 \\
{}^{30}_{33} & Q_2\ Q_1\ Q_0\ Q_2\ Q_2 & 2\ 1\ 1\ 0\ 2\ 2\ 1\ 0 \\
{}^{30}_{33} & Q_0\ Q_2\ Q_2\ Q_2\ Q_0 & 0\ 0\ 2\ 2\ 2\ 1\ 1\ 1 \\
{}^{30}_{33} & Q_0\ Q_2\ Q_0\ Q_1\ Q_1 & 1\ 2\ 2\ 0\ 1\ 1\ 2\ 0 \\
{}^{30}_{33} & Q_0\ Q_2\ Q_1\ Q_0\ Q_2 & 2\ 1\ 2\ 1\ 0\ 1\ 0\ 2
\end{array}
$$

Figure (8-7) : Compact Version of the AC(48,4,33)

(Note: Every zero, one and two in Figure (8-7) is a 9-zero, -one, or -two column.)

The generator matrix of the anticode resulting from the iterative image stacking is formed exactly as in the way of the binary case, but now using the Q-I_q property of the q-ary parent anticode (or anticodes). Therefore the generator matrix of the AC(48, 4, 33) is obtained with the same procedure as in Section 6.4.6.2, as follows. The generator matrix of the domain anticode and the first anticode have the form:

$$
\left[G_q\right]_{D_{AC}} = \begin{bmatrix} 0 & 0 & 1 & 2 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix} \quad \& \quad G_{Q_0} = \left[G_q\right]_{F_{AC}} = \begin{bmatrix} 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 \\ 0 & 0 & 1 & 1 & 1 & 2 & 2 & 2 \end{bmatrix}
$$

The generator matrix of the second mapped anticode has the form:

$$\left[G_q\right]_{S_{AC}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 \\ 0 & 0 & 1 & 1 & 1 & 2 & 2 & 2 \end{bmatrix}$$

Thus the generator matrix of the resultant AC(48, 4, 33) has the form:

$$\left[G_q\right]_{AC} = \left[\begin{array}{c|c|c|c|c|cccccccc} G_{Q_o} & G_{Q_o} & G_{Q_o} & G_{Q_o} & G_{Q_o} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 2 & 1 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 \\ \hline 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 2 & 2 & 2 \end{array}\right],$$

where the zeros, ones and twos in each column related to G_{Q_o} are an 8-digit row of all zeros, ones and twos.

The iterative image stacking process reduces considerably the effort needed to search for linear q-ary anticodes (codes), in the sense that for any dimension k, a very small set of anticodes with small dimension is needed to construct an infinitely large set of anticodes. This process, however, can be extended using column refinement methods for q-ary anticodes.

8.2 Explicit 3-ary Anticode Construction Results

In this section the results of the construction of 3-ary anticodes using only the iterative stacking method are given. Although a computer programme based on the Q-I_q property of the parent anticode (or anticodes) could be used, to find infinite sets of 3-ary anticodes, this list reveals the simplicity and effectiveness of the method, which enables the construction of an anticode using a formula and some easy

manipulation.

The results of the 3-ary anticode construction for some anticodes of $4 \leqslant k \leqslant 6$ are given as follows:

The generator matrix of the AC(22,3,16) consists of the column types $\{1\text{-}18,\ 21,\ 24\text{-}26\}$, and $d_{min} = 14$.

The generator matrix of the AC(2,1,2) consists of the column types $\{1 - 2\}$.

The generator matrix of the AC(26,3,18), consists of the column types $\{1 - 26\}$. Therefore,

$$D_{AC}(1+22,3,16) \xrightarrow{\alpha|} F_{AC}(2,1,2) \xrightarrow{\alpha|\alpha'} S_{AC}(26,4,18,1) \equiv$$

$$AC(72,4,50) \longrightarrow \boxed{DC(8,4,4)_{(4)}} \quad \alpha' = (MM)_{o}^{1}$$

AC(72,4,50) plus any non-repeated column from PAC(80,4,54), results in the AC(73,4,51), therefore

$$AC(73,4,51) \longrightarrow \boxed{DC(7,4,3)_{(3)}} .$$

$$D_{AC}(1+22,3,16) \xrightarrow{\alpha|} F_{AC}(2,1,2) \xrightarrow{\alpha|\alpha'} S_{AC}(22,4,16,1) \equiv$$

$$AC(68,4,48) \longrightarrow \boxed{DC(12,4,6)_{(6\text{-}7)}} \quad \alpha' = (MM)_{o}^{1}$$

The generator matrix of the AC(19,3,14) consists of the column types $\{1 - 19\}$, then

$$D_{AC}(1+22,3,16) \xrightarrow{\alpha|} F_{AC}(2,1,2) \xrightarrow{\alpha|\alpha'} S_{AC}(19,4,14,1) \equiv$$

$$AC(65,4,46) \longrightarrow \boxed{DC(15,4,8)_{(8\text{-}9)}} \quad \alpha' = (MM)_{o}^{1}$$

AC(50,4,35) + any new column from PAC(80,5,54) $\equiv$

$$AC(51,4,36) \longrightarrow \boxed{DC(29,4,18)_{(18)}}$$

The generator matrix of the AC(13,3,9) consists of the column types $\{1,3\text{-}5,\ 9\text{-}17\}$, and $d_{min} = 9$, therefore

$$D_{AC}(1+13,3,9) \xrightarrow{\alpha|} F_{AC}(2,1,2) \xrightarrow{\alpha|\alpha'} S_{AC}(22,4,16,1) \equiv$$

$$AC(50,4,35) \longrightarrow \boxed{DC(30,4,19)_{(19)}} . \quad \alpha' = (MM)_{o}^{1}$$

The AC(48,4,33) of the next construction, plus any column from PAC(80,4,54) gives

$$AC(49,4,34) \longrightarrow \boxed{DC(31,4,20)_{(20)}}$$

The generator matrix of the AC(4,2,3) consists of the column types $\{1,\ 3\text{-}5\}$, and min d = 3 .

The generator matrix of the AC(8,2,6) consists of the column types $\{1 - 8\}$.

Therefore:

$$D_{AC}(1+4,2,3) \overset{\alpha|}{+} F_{AC}(8,2,6) \overset{\alpha|\alpha'}{+} S_{AC}(8,4,6,2) \equiv$$

$$AC(48,4,33) \longrightarrow \boxed{DC(32,4,21)_{(21)}} \qquad \alpha' = (MM)_o^2 .$$

The generator matrix of the AC(7,2,6) consists of the column types $\{1 - 7\}$; therefore

$$D_{AC}(1+4,2,3) \overset{\alpha|}{+} F_{AC}(8,2,6) \overset{\alpha|\alpha'}{+} S_{AC}(7,4,6,2) \equiv$$

$$AC(47,4,33) \longrightarrow \boxed{DC(33,4,21)_{(21)}} \qquad \alpha' = (MM)_o^2 .$$

The generator matrix of the AC(6,2,5) consists of the column types $\{1 - 6\}$ - then

$$D_{AC}(1+4,2,3) \overset{\alpha|}{+} F_{AC}(8,2,6) \overset{\alpha|\alpha'}{+} S_{AC}(6,4,5,2) \equiv$$

$$AC(46,4,32) \longrightarrow \boxed{DC(34,4,22)_{(22)}} \qquad \alpha' = (MM)_o^2$$

The generator matrix of the AC(5,2,4) consists of the column types $\{1,3 - 6\}$, then

$$D_{AC}(1+4,2,3) \overset{\alpha|}{+} F_{AC}(8,2,6) \overset{\alpha|\alpha'}{+} S_{AC}(5,4,4,2) \equiv$$

$$AC(45,4,31) \longrightarrow \boxed{DC(35,4,23)_{(23)}} \qquad \alpha' = (MM)_o^2$$

The generator matrix of the AC(4,2,3) consists of the column types $\{1, 3\text{-}5\}$ - then

$$D_{AC}(1+4,2,3) \overset{\alpha|}{+} F_{AC}(8,2,6) \overset{\alpha|\alpha'}{+} S_{AC}(4,4,3,2) \equiv$$

$$AC(44,4,30) \longrightarrow \boxed{DC(36,4,24)_{(24)}} \qquad \alpha' = (MM)_o^2$$

$$D_{AC}(1+13,2,9) \overset{\alpha|}{+} F_{AC}(2,1,2) \overset{\alpha|\alpha'}{+} S_{AC}(13,4,9,1) \equiv$$

$$AC(41,4,28). \qquad \alpha' = (MM)$$

This anticode has min d = 27, so therefore

$$AC(41,4,28) \equiv \boxed{DC(41,4,27)_{(27)}} .$$

If, $\overset{\alpha|\alpha'}{+} S_{AC}(14,4,10,1) \equiv AC(42,4,29)$,

this anticode has min d = 27, so therefore

$$AC(42,4,29) \equiv \boxed{DC(42,4,27)_{(27)}} .$$

If, $\overset{\alpha|\alpha'}{+} S_{AC}(15,4,11,1) \equiv AC(43,4,30)$,

this anticode has d = 27, so therefore

$$AC(43,4,30) \equiv \boxed{DC(43,4,27)_{(27)}} .$$

AC(210,5,142) + any two new columns from PAC(80,4,54) $\equiv$

$$AC(212,5,144) \longrightarrow \boxed{DC(30,5,18)_{(18)}}$$

AC(210,5,142) + any new column from PAC(80,4,54) $\equiv$

$$AC(211,5,143) \longrightarrow \boxed{DC(31,5,19)_{(19)}}$$

$$D_{AC}(1+22,3,16) \overset{\alpha|}{+} F_{AC}(8,2,6) \overset{\alpha|\alpha'}{+} S_{AC}(26,5,18) \equiv$$

$$AC(210,5,142) \longrightarrow \boxed{DC(32,5,20)_{(20)}} \qquad \alpha' = (MM)_o^2$$

$$D_{AC}(1+22,3,16) \overset{\alpha|}{+} F_{AC}(8,2,6) \overset{\alpha|\alpha'}{+} S_{AC}(22,3,16) \equiv$$

$$AC(206,5,140) \longrightarrow \boxed{DC(36,5,22)_{(22\text{-}23)}}$$

$$D_{AC}(1+2,1,2) \overset{\alpha|}{+} F_{AC}(50,4,35) \overset{\alpha|\alpha'}{+} S_{AC}(2,5,2,4) \equiv$$

$$AC(152,5,105) \longrightarrow \boxed{DC(90,5,57)_{(57\text{-}59)}} .$$

$D_{AC}(1+2,1,2) \overset{\alpha}{+} F_{AC}(49,4,34) \overset{\alpha|\alpha'}{+} S_{AC}(2,5,2,4) \equiv$

$AC(149,5,102) \longrightarrow \boxed{DC(93,5,60)_{(60-61)}}$

$D_{AC}(1+2,1,2) \overset{\alpha}{+} F_{AC}(48,4,33) \overset{\alpha|\alpha'}{+} S_{AC}(2,5,2,4) \equiv$

$AC(146,5,99) \longrightarrow \boxed{DC(96,5,63)_{(63)}}$

$AC(138,5,93)$ + any 3 new columns from $PAC(242,5,162) \equiv$

$AC(141,5,96) \longrightarrow \boxed{DC(101,5,66)_{(66)}}$

$AC(138,5,93)$ + any 2 new columns from $PAC(242,5,162) \equiv$

$AC(140,5,95) \longrightarrow \boxed{DC(102,5,67)_{(67)}}$

$AC(138,5,93)$ + any new column from $PAC(242,5,162) \equiv$

$AC(139,5,94) \longrightarrow \boxed{DC(103,5,68)_{(68)}}$

$D_{AC}(1+4,2,3) \overset{\alpha}{+} F_{AC}(26,3,18) \overset{\alpha|\alpha'}{+} S_{AC}(8,5,6,3) \equiv$

$AC(138,5,93) \longrightarrow \boxed{DC(104,5,69)_{(69)}} \qquad \alpha' = (MM)^3_o$

$AC(136,5,92)$ + any column from $PAC(242,5,162) \equiv$

$\Updownarrow \qquad AC(137,5,93) \longrightarrow \boxed{DC(105,5,69)}$

$D_{AC}(1+2,1,2) \overset{\alpha}{+} F_{AC}(45,4,31) \overset{\alpha|\alpha'}{+} S_{AC}(2,5,2,4) \equiv$

$AC(137,5,93) \longrightarrow DC(105,5,69)_{(69)}$

If $\overset{\alpha|\alpha'}{+} S_{AC}(6,5,5,3) \equiv AC(136,5,92)$

$\longrightarrow \boxed{DC(106,5,70)_{(70)}} \qquad \alpha' = (MM)^3_o$

If $\overset{\alpha|\alpha'}{+} S_{AC}(5,5,4,3) \equiv AC(135,5,91)$

$\longrightarrow \boxed{DC(107,5,71)_{(71)}} \qquad \alpha' = (MM)^3_o$

$D_{AC}(1+2,1,2) \overset{\alpha}{+} F_{AC}(44,4,30) \overset{\alpha|\alpha'}{+} S_{AC}(2,5,2,4) \equiv$

$AC(134,5,90) \longrightarrow \boxed{DC(108,5,72)_{(72)}}$

If $\underline{\alpha|\alpha'}\ S_{AC}(4,2,3,3) \equiv AC(134,5,90)$

$\longrightarrow \boxed{DC(108,5,72)_{(72)}} \qquad \alpha' = (MM)_o^3$

$D_{AC}(1+2,2,1) \underline{\alpha|}\ F_{AC}(43,4,30) \underline{\alpha|\alpha'}\ S_{AC}(2,5,2,4) \equiv$

$AC(131,5,90) \longrightarrow \boxed{DC(111,5,72)_{(72-73)}}$

$D_{AC}(1+2,1,2) \underline{\alpha|}\ F_{AC}(42,4,29) \underline{\alpha|\alpha'}\ S_{AC}(2,5,2,4) \equiv$

$AC(128,5,87) \longrightarrow \boxed{DC(114,5,75)_{(75)}}$

$AC(125,5,84)$ + any 2 new column from $PAC(242,5,162) \equiv$

$AC(127,5,86) \longrightarrow \boxed{DC(115,5,76)_{(76)}}$

$AC(125,5,84)$ + any new column from $PAC(242,5,162) \equiv$

$AC(126,5,85) \longrightarrow \boxed{DC(116,5,77)_{(77)}}$

$D_{AC}(1+2,1,2) \underline{\alpha|}\ F_{AC}(41,4,28) \underline{\alpha|\alpha'}\ S_{AC}(2,5,2,4) \equiv$

$AC(125,5,8,4) \longrightarrow \boxed{DC(117,5,78)_{(78)}}$

$D_{AC}(1+13,3,9) \underline{\alpha|} F_{AC}(8,2,6) \underline{\alpha|\alpha'}\ S_{AC}(26,5,18,2) \equiv$

$AC(138,5,93), \quad \alpha' = (MM)_o^2$

The $AC(138,5,93)$ has $d_{min} = 90$, thus

$AC(138,5,93) \longrightarrow \boxed{DC(138,5,90)_{(90)}}$

$D_{AC}(1+4,2,3) \underline{\alpha|} F_{AC}(13,3,9) \underline{\alpha|\alpha'}\ S_{AC}(8,5,6,3) \equiv$

$AC(73)5,51) \longrightarrow \boxed{DC(169,5,111)_{(111)}}$

$AC(72,4,50) \xrightarrow{(MM)_o^1} AC(72,5,50,1)$

$\longrightarrow \boxed{DC(170,5,112)_{(112)}}$

$AC(50,4,35) \xrightarrow{(MM)_o^1} AC(50,5,35,1)$

$\longrightarrow \boxed{DC(192,5,127)_{(127)}}$

$$AC(49,4,34) \xrightarrow{(MM)_o^1} AC(49,5,34,1)$$

$$\longrightarrow \boxed{DC(193,5,128)}_{(128)}$$

$$AC(48,4,33) \xrightarrow{(MM)_o^1} AC(48,5,33,1)$$

$$\longrightarrow \boxed{DC(194,5,129)}_{(129)}$$

$$AC(47,4,33) \xrightarrow{(MM)_o^1} AC(47,5,33,1)$$

$$\longrightarrow \boxed{DC(195,5,129)}_{(129)}$$

$$AC(46,4,32) \xrightarrow{(MM)_o^1} AC(46,5,32,1)$$

$$\longrightarrow \boxed{DC(196,5,130)}_{(130)}$$

$$AC(45,4,31) \xrightarrow{(MM)_o^1} AC(45,5,31,1)$$

$$\longrightarrow \boxed{DC(197,5,131)}_{(131)}$$

$$AC(44,4,30) \xrightarrow{(MM)_o^1} AC(44,5,30,1)$$

$$\longrightarrow \boxed{DC(198,5,132)}_{(132)}$$

$$AC(43,4,30) \xrightarrow{(MM)_o^1} AC(43,5,30,1)$$

$$\longrightarrow \boxed{DC(199,5,132)}_{(132)}$$

$$AC(42,4,29) \xrightarrow{(MM)_o^1} AC(42,5,29,1)$$

$$\longrightarrow \boxed{DC(200,4,133)}_{(133)}$$

$$D_{AC}(1+2,1,2) \overset{\alpha}{\dagger} F_{AC}(138,5,93) \overset{\alpha|\alpha'}{\dagger} S_{AC}(2,6,2,5) \equiv$$

$$AC(416,6,279) \longrightarrow \boxed{DC(312,6,207)}_{(207)}$$

$$AC(408),6,273) + \text{any 3 new columns from } PAC(728,6,486) \equiv$$

$$AC(411,276) \longrightarrow \boxed{DC(317,6,210)}_{(210)}$$

$D_{AC}(1+2,1,2) \overset{\alpha}{+} F_{AC}(136,5,92) \overset{\alpha|\alpha'}{+} S_{AC}(2,6,2,5) \equiv$

$\Updownarrow$ $\quad AC(410,6,276) \longrightarrow \boxed{DC(318,6,210)_{(210-211)}}$

$AC(408,6,273)$ + any 2 new columns from $Q_{AC}(728,6,486) \equiv$

$AC(410,6,275) \longrightarrow \boxed{DC(318,6,211)_{(211)}}$

$AC(408,6,273)$ + any new column from $PAC(728,6,486) \equiv$

$AC(409,6,274) \longrightarrow \boxed{DC(319,6,212)_{(212)}}$

$D_{AC}(1+4,2,3) \overset{\alpha}{+} F_{AC}(80,4,54) \overset{\alpha|\alpha'}{+} S_{AC}(8,6,6,4) \equiv$

$AC(408,6,273) \longrightarrow \boxed{DC(320,6,213)_{(213)}} \qquad \alpha' = (MM)_{o}^{4}$

$D_{AC}(1+2,\ 1,2) \overset{\alpha}{+} F_{AC}(135,5,91) \overset{\alpha|\alpha'}{+} S_{AC}(2,6,2,5) \equiv$

$AC(407,6,276) \longrightarrow \boxed{DC(321,6,213)_{(213)}}$

$AC(390,6,261)$ + any 3 new columns from $PAC(728,6,486) \equiv$

$AC(393,6,264) \longrightarrow \boxed{DC(335,6,222)_{(222)}}$

$AC(390,6,261)$ + any 2 new columns from $PAC(728,6,486) \equiv$

$AC(392,6,262) \longrightarrow \boxed{DC(336,6,223)_{(223)}}$

$AC(390,6,261)$ + any new column from $PAC(728,6,486) \equiv$

$AC(391,6,262) \longrightarrow \boxed{DC(337,6,224)_{(224)}}$

$D_{AC}(1+13,3,4) \overset{\alpha}{+} F_{AC}(2,6,3,18) \overset{\alpha|\alpha'}{+} S_{AC}(26,6,18,3) \equiv$

$AC(390,6,261) \longrightarrow \boxed{DC(338,6,225)_{(225)}} \qquad \alpha' = (MM)_{o}^{3}$

$AC(386,6,259)$ + any two new columns from $PAC(728,6,486) \equiv$

$AC(388,6,261) \longrightarrow \boxed{DC(340,6,225)_{(225)}}$

$AC(386,6,259)$ + any new column from $PAC(728,6,486) \equiv$

$AC(387,6,260) \longrightarrow \boxed{DC(341,6,226)_{(226)}}$

If $\alpha \mid \alpha' S_{AC}(22,6,16,3) \equiv AC(386,6,259)$

$\longrightarrow \boxed{DC(342,6,227)}_{(227)} \quad \alpha' = (MM)_o^3$

If $\alpha \mid \alpha' S_{AC}(20,6,15,3) \equiv AC(384),6,258)$

$\longrightarrow \boxed{DC(344,6,228)}_{(228)} \quad \alpha' = (MM)_o^3$

If $\alpha \mid \alpha' S_{AC}(19,6,14,3) \equiv AC(383,6,257)$

$\longrightarrow \boxed{DC(345,6,229)}_{(229)} \quad \alpha' = (MM)_o^3$

If $\alpha \mid \alpha' S_{AC}(18,6,13,3) \equiv AC(382,6,256)$

$\longrightarrow \boxed{DC(346,6,230)}_{(230)} \quad \alpha' = (MM)_o^3$

If $\alpha \mid \alpha' S_{AC}(17,6,12,3) \equiv AC(381,6,255)$

$\longrightarrow \boxed{DC(347,6,231)}_{(231)} \quad \alpha' = (MM)_o^3$

If $\alpha \mid \alpha' S_{AC}(16,6,12,3) \equiv AC(380,6,255)$

$\longrightarrow \boxed{DC(348,6,231)}_{(231)} \quad \alpha' = (MM)_o^3$

If $\alpha \mid \alpha' S_{AC}(15,6,11,3) \equiv AC(379,6,254)$

$\longrightarrow \boxed{DC(349,6,232)}_{(232)} \quad \alpha' = (MM)_o^3$

If $\alpha \mid \alpha' S_{AC}(14,6,10,3) \equiv AC(378,6,253)$

$\longrightarrow \boxed{DC(350,6,233)}_{(233)} \quad \alpha' = (MM)_o^3$

If $\alpha \mid \alpha' S_{AC}(13,6,9,3) \equiv AC(377,6,252)$

$\longrightarrow \boxed{DC(351,6,234)}_{(234)} \quad \alpha' = (MM)_o^3$

$AC(212,5,143) \xrightarrow{(MM)_1^o} AC(212,6,143,1)$

$\longrightarrow \boxed{DC(516,6,342)}_{(342)}$

$AC(211,5,143) \xrightarrow{(MM)_1^o} AC(211,6,143,1)$

$\longrightarrow \boxed{DC(517,6,343)}_{(343)}$

$$AC(210,5,142) \xrightarrow{(MM)_1^o} AC(210,6,142,1)$$

$$\longrightarrow \boxed{DC(518,6,344)}_{(344)}$$

$$AC(141,5,96) \xrightarrow{(MM)_1^o} AC(141,6,96,1)$$

$$\longrightarrow \boxed{DC(587,6,390)}_{(390)}$$

$$AC(138,5,93) \xrightarrow{(MM)_1^o} AC(138,6,93,1)$$

$$\longrightarrow \boxed{DC(590,6,393)}_{(393)}$$

$$AC(137,5,93) \xrightarrow{(MM)_1^o} AC(137,6,93,1)$$

$$\longrightarrow \boxed{DC(591,6,293)}_{(393)}$$

$$AC(136,5,92) \xrightarrow{(MM)_1^o} AC(136,6,92,1)$$

$$\longrightarrow \boxed{DC(592,6,394)}_{(394)}$$

$$AC(134,6,90,1) + \text{any new column from } PAC(728,6,486) \equiv$$

$$AC(135,6,91,1) \longrightarrow \boxed{DC(593,6,395)}_{(395)}$$

$$AC(134,5,90) \xrightarrow{(MM)_1^o} AC(134,6,90,1)$$

$$\longrightarrow \boxed{DC(594,6,396)}_{(396)}$$

$$AC(125,5,84) \xrightarrow{(MM)_1^o} AC(125,6,84,1)$$

$$\longrightarrow \boxed{DC(603,6,402)}_{(402)}$$

The 3-ary anticodes, constructed by the iterative stacking process have produced linear 3-ary block codes. These codes meet the Griesmer bound exactly, thus they are optimum codes. However, the resultant codes have $n \leqslant q^{k-1} - 1$ (except for those resulting from the $(MM)_b^a$ process only). Therefore, for the same k, the 3-ary codes resulting from the

iterated anticode stacking process have higher rates than the construction method suggested by Griesmer which have $n \geq q^{k-1} - 1$. This is due to the restriction imposed as $\sum_i k_i \leq k$, (see Section 5.2) by Griesmer.

CHAPTER IX

DISCUSSION AND CONCLUSION

The main objective of the research described in this thesis was the extension and development of the anticode method for the construction of linear block error-control codes, as a means of increasing the security of the transmitted data in a communication system.

In Chapters III and IV, the development necessarily involved the review and analysis of the mathematical and practical construction of some of the best known coding techniques, which in turn involved a brief review of the related mathematical background and concepts of error-control coding. The anticodes originate from parent anticodes. Because of the equivalent relationship between the parent anticode and maximal length code, the Griesmer bound and the Solomon and Stiffler generalization of this bound by means of a simpler approach, were reviewed.

Since anticodes originate from parent anticodes, the mathematical analysis of the parent anticodes was established. From this, the properties of the Q-I systematic form of anticodes, and the partitioning of the generator matrix of the parent anticode and hence the generator matrix of any anticode into the smaller generator matrices (subsections), was introduced. Furthermore, these two properties lead to the additional properties:

(a) the explicit derivation of the anticode weight distribution from the Q-I property; and

(b) predetermination of the weight attribute of any column or columns related to the row space of the subsections of the generator matrix of the anticode in the code book.

Considering these properties, various anticode constructions from smaller dimension unit basic anticodes resulted. Using these methods,

a large number of optimum and near-optimum anticode, and as a result, a large number of optimum and near-optimum codes, were constructed.

Once an optimum unit basic anticode was constructed, it could be used as the basis of construction (domain anticode) in the iterated image stacking process, and other various construction processes, to obtain larger dimension anticodes. Thus the construction of optimum long unit basic anticodes was important as part of the process of obtaining short optimum linear codes of high efficiency. On the other hand, short unit basic anticodes soon lead to short, low efficiency anticodes. If the iterative image stacking process of anticode construction resulted in a non-optimum anticode, the resulting anticode could be used as a basis for various complementary computer searches for the construction of the optimum or near optimum anticodes (refinement process). However, if the anticode was optimum, the same computer methods could be used to obtain longer or shorter optimum anticodes.

The linear codes derived from anticodes, constructed by the various processes, for a fixed dimension k and minimum distance d have block length

$$n \leq q^{k-1} - 1,$$

except for those anticodes constructed by the processes of mapped-map stacking. However, the linear codes suggested by Solomon and Stiffler for the same parameters k and d have bounded block length

$$n \geq q^{k-1} - 1.$$

Therefore the anticode construction methods described here result in higher rate codes. This difference arises from the weak sufficient condition imposed on the suggested construction method. The Solomon and Stiffler algorithm consists of the deletion of the subgroups (subspace) of the additive group (vector space) of the columns of a maximal length sequence code. For a $(n_o = q^k - 1,\ k,\ d = (q-1)\ q^{k-1})$

maximal length code, the number of deletion columns are:

$$m = \sum_i q^{\ell_i} - 1$$

where

$$\begin{cases} \sum_i \ell_i \leq k & (1) \\ \ell_i \neq \ell_j & (2) \end{cases}$$

The maximum value of m is achieved if $\ell_1 = k - 1$ and $\ell_2 = 1$. This means that the constructed, punctured set have been formed by the first $\ell_1 = k - 1$ information column of the maximal length code, namely, $I_1, I_2, \ldots I_{k-1}$ and all linear combination of these columns, plus the column $\ell_2 = 1$, namely, the last information column. As an example, if k = 7, the maximum value of m is achieved when $\ell_1 = k-1 = 6$ and $\ell_2 = 1$, thus

$$m = (2^{k-1} - 1) + (2^1 - 1) = 63 + 1 = 64$$

$$n = n_o - m = 127 - 64 = 63$$

which is the block length of the shortest possible code. On the contrary, construction by means of the simple stack of BAC(63,6,32) gives

$$\text{BAC}(63,6,32) \xrightarrow{(MM)_1^o} \text{BAC}(63,7,32,1)$$

The columns of this anticode are exactly the same as the above 63 columns of the Solomon & Stiffler construction formed from linear combinations of information columns $I_1, I_2, \ldots I_6$. For the Solomon & Stiffler construction, from the conditions

$$\begin{cases} \sum_i \ell_i \leq k \\ \ell_i \neq \ell_j \end{cases}$$

it was possible to add only one column to these 63 columns to construct

a set of punctured columns of length $m = 63 + 1 = 64$. However, with the anticode construction methods described, it is possible to add 48 columns, by concatenating the simple inverted form of the AC(48,6,26) to the AC(63,7,32,1) to get AC(111,7,58), as follows:

$$AC(48,6,26) \xrightarrow{(IM)_0^1} AC(48,7,48)$$

$$AC(63,6,32) + AC(48,7,48) \equiv AC(111,7,58) \quad .$$

Deleting this anticode from the PAC(127,7,64) gives DC(16,7,6) which is an optimum linear code and has exactly the same parameter as that in the table of Stinaff and Helgert. This counter example shows how weak the conditions imposed by Solomon & Stiffler are, which makes the codes obey the conditions

$$k/n \longrightarrow 0$$

when

$$n \longrightarrow \infty$$

which are not imposed on codes derived from anticodes as long as the optimum long unit basic anticode can be constructed.

The concept of binary anticodes in Chapter VIII is extended to the construction of anticodes whose elements are chosen from the finite field GF(q). The same methods of construction as in the binary case are applicable also for q-ary anticodes, with operations defined over the elements of GF(q). The anticode construction method over GF(q) was applied to construct 3-ary anticodes, and a large number of optimum and a few near-optimum anticodes, and as a result, a large number of linear 3-ary codes, were found. However, a complementary computer study could be done to extend the results as easily as in the binary case. The optimality and long block length of the unit basic anticodes are also as important in the construction of anticodes/codes over GF(q) as in the binary case.

Recommendations for Further Research

From the above discussion, a number of possibilities for further development and extension of the anticode concepts arise as follows:

(a) the Q-I systematic property of a column and a group of columns provide a predetermination of the weight attribute of the column or a set of columns of the parent anticode, and any related anticode. As an example, if a column of the generator matrix of an anticode starts with two zero's, then the 4 elements of this column in the parent anticode or anticode from the $(2^i + 1)^{th}$ element are solely determined by the i^{th} element of the generator matrix, e.g. if $i^{th} = 5^{th}$ element in the generator matrix is a zero, the 4 elements starting from the $(2^5 + 1)^{th} = 33^{rd}$ element are zero's in the parent anticode or code; if the $i^{th} = 5^{th}$ element in the generator matrix is a one, the 4 elements starting from the $(2^5 + 1)^{th} = 33^{rd}$ elements are ones. This also applies for a group of columns. Thus, using this concept as a basis, a computer search program may be speeded up considerably. Instead of dealing with individual elements, the group (subsections) of the generator matrix may be processed in one step.

(b) the iterative image stacking process of anticode construction may be directly applied to linear optimum block codes of smaller dimension to construct optimum or near optimum larger dimension linear block codes. Intuitively, in constructing optimum anticodes from optimum small dimension anticodes over GF(3), the resulting anticodes turned out to have the maximum minimum distance for their parameters. Thus the resulting anticode turned out to be an optimum code.

(c) The same anticode construction processes may be used also in conjunction with some of the already existing optimum anticodes related to existing optimum block codes from other sources; that is,

found from known isolated optimum or near-optimum codes.

(d) a complementary computer search parallel to that of (a) and Section 6.4.8 may be used also for anticodes over GF(3) to extend the results of the anticodes over GF(q). Also, it would be interesting to apply the anticode construction methods to the anticode over the elements of the finite fields of GF(4) and GF(5), and so on.

Appendix I

Encoder and decoder for the codes derived from parent anticodes

The equivalence of the maximal length shift register code book [D] of Section 5.1 and the parent anticode [C] can be used to encode and decode codes derived from the parent anticode, taking advantage of the maximal length linear feedback shift register code (M-sequence code) properties.

The code book [D] of an (n_o, k_o, d_o), M-sequence code forms an Abelian group (subspace of the vector space of all n_o-tuples), thus it can be generated by any k_o linearly independent vectors (rows) in [D]. Following the example of Section 5.1, the code book [D] of the 3-bit shift register of Figure (I-a) generated by the primitive feedback polynomial $p(x) = x^3+x+1$ is:

$$[D] = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

The generator matrix of [D] can be formed from the first $k_o = 3$ rows of [D] as

$$[G] = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

The row space of the generator matrix [G] which is the matrix [D] is formed as $[L] \cdot [G] = [D]$, so then

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

where $[L]$ is a $(2^k - 1).K$ matrix.

Multiplying the matrix $[L]$ on the left by a non-singular row permutation matrix $[A]$ such that,

$$[A] \cdot [L] = [M]^T$$

then

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

The matrix $[M]^T$ is identical to that of Section 6.2.1 and is the k_o variable truth table, excluding the all zero row, and can be generated by a k_o-bit binary counter, as in Figure (I-b). Thus,

$$[A] \cdot [L] \cdot [G] = [M]^T \cdot [G] = [D_p] \simeq [D]$$

Alternatively $[D_p]$ is the same as the M-seq. code book $[D]$, only the rows have been permuted with respect to $[A]$. However, $[D_p]$ has been spanned by $[A]$ $[L] = [M]^T$, therefore the columns of $[D_p] \simeq [D]$ are in Q-I systematic form, thus any k_o dimension anticode can be deleted from $[D_p]$. This means that the equivalence can be used to encode and decode

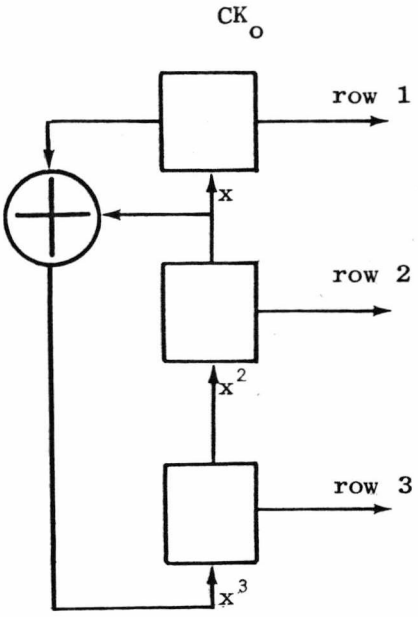


Figure (I-a) : M-sequence and
[G] matrix generator

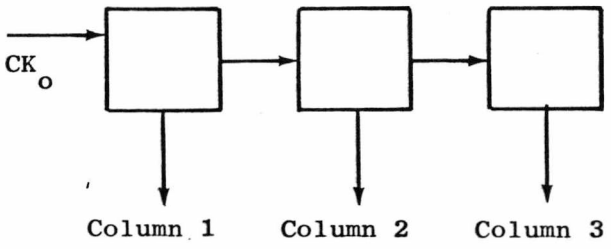


Figure (I-b) : [M] matrix generator

the codes derived from anticodes, deleting the anticode column positions from the m-sequences generated by a k_o-bit feedback shift register. The encoding and decoding procedure is the same as the encoder-decoder suggested by Solomon and Stiffler (1964), shown in Figure (I-2), and Figure (I-3).

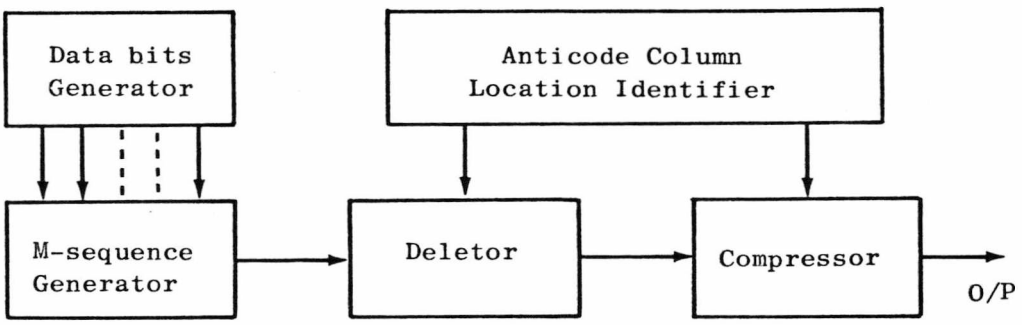


Figure (I-2) : Solomon and Stiffler Type Encoder

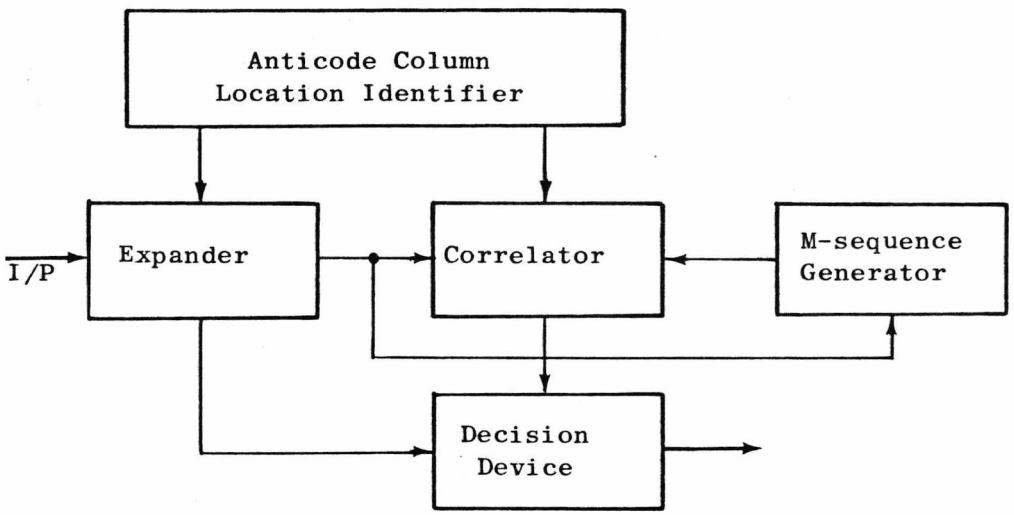


Figure (I-3) : Solomon & Stiffler Type Decoder

However, from the similarity between the parent anticode [C] and the Walsh-Hadamard matrix, it might be possible to take advantage of the latter's fast transform algorithm to decode the derived code from the parent anticode in a simpler way.

Appendix II

Approximate performance of the linear binary codes derived from parent anticodes.

The performance of a linear binary DC(n, k, d), over a binary symmetric channel, can be approximated by using the minimum distance of the code.

However, in general this is very approximate (Slepian 1956). A better approximation can be achieved by the use of the weight distribution of the linear block code (Cohen et al. 1976, Huntoon & Michelson 1977). Using the explicit derivation of the anticode-related code weight distribution, the decoded block error probability (regardless of the post decoder errors) of some linear binary block codes have been examined. The procedure uses the weight distribution of the code to approximate the number α_i of the error patterns of weight i which are correctable by the code. Having α_i, the probability of incorrect decoding

$$P_{ICD} = 1 - \sum_{i=o}^{n} \alpha_i \, p^i (1-p)^{n-i}$$

can be evaluated, where p is the crossover probability of the channel.

The α_i can be approximated from the code weight distribution from the equation,

$$\alpha_i = Q_i \, \beta_i$$

given by Hobbs (1965), where

$$\beta_i = \binom{n}{i} = \frac{n!}{i!(n-i)!}$$

A_j being the number of code words of weight j. Then

$$Q_i = \Pi_j (1 - p(j/i))^{A_j}$$

where the conditional probability is

$$P(j/i) = \frac{1}{\beta_i}\left\{\left[\sum_{r=j/2+1}^{j} c(i, j, r)\right] + \frac{1}{2}\, c(i, j, \tfrac{1}{2})\right\} ,$$

and $\quad c(i, j, r) = \binom{j}{r}\binom{n-j}{i-r}$.

The entires α_i and β_i in the following tables are the number of correctable error patterns α_i out of the β_i possible error pattern of weight i. These have been computed by a simple computer program, given in this appendix, on an ICL-2960 machine.

DC(9,5,3) has weight polynomial,

$$A(Z) = 1 + 6Z^3 + 10Z^4 + 8Z^5 + 4Z^6 + 2Z^7 + Z^8$$

i	0	t=1	2	3	4	5	6	7	8	9
β_i	1	9	36	-	-	-	-	-	-	-
α_i	1	9	11	-	-	-	-	-	-	-

DC(9,5,1) has weight polynomial,

$$A(Z) = 1 + Z + 14Z^4 + 14Z^5 + Z^8 + Z^9$$

i	0	1	2	3	4	5	6	7	8	9
β_i	1	9	36	-	-	-	-	-	-	-
α_i	1	8	9	-	-	-	-	-	-	-

DC(23,6,10) has weight polynomial,

$$A(Z) = 1 + 31Z^{10} + 19Z^{12} + 5Z^{14} + 8Z^{16}$$

i	0	1	2	3	t=4	t+1=5	6
β_i	1	23	253	1771	8855	33649	100947
α_i	1	23	253	1771	8855	29954	52178

i	7	8	9	10	11 23
β_i	245157	490314	817190	-	- ..... -
α_i	28787	2429	12	-	- ..... -

DC(23,6,7) has weight polynomial,

$$A(Z) = 1 + Z^7 + 15Z^{10} + 15Z^{11} + 15Z^{12} + 15Z^{13} + Z^{17} + Z^{22},$$

i	0	1	2	t=3	t+1=4	5	6
β_i	1	23	253	1771	8855	33649	100947
α_i	1	23	253	1771	8837	31521	67341

i	7	8	9	10	11	.....	23
β_i	245157	490314	817190	-	-	.....	-
α_i	57168	10393	189	-	-	.....	-

DC(48,7,22) has weight polynomial,

$$A(Z) = 1 + 48Z^{22} + 30Z^{24} + 48Z^{26} + Z^{48},$$

i	0	1	2	3	4	5
β_i	1	48	1128	17296	194580	1712304
α_i	1	48	1128	17296	194580	1712304

i	6	7	8	9
β_i	12271512	73629072	377348994	167710664
α_i	12271512	73629072	377348994	167710664

i	t=10	t+1=11	12
β_i	6540715896	22595200368	69668534468
α_i	6540715896	22578276209	69158585434

i	13	14
β_i	192928249296	482306232400
α_i	185443773610	413871745605

i	15	16
β_i	1093260079344	2254848913647
α_i	682089472348	662685956238

i	17	18
β_i	4244421484512	7309837001104
α_i	265369038250	23703131413

i	19	20
β_i	11541847896480	16735679449896
α_i	366942233	347078

i	21	22		48
β_i	22314239266528	-	.....	-
α_i	11	-	.....	-

DC(48,7,16) has weight polynomial

$$A(Z) = 1 + 3Z^{16} + 120Z^{24} + 3Z^{32} + Z^{48}$$

i	0	1	2	3	4	5
β_i	1	48	1128	17296	194580	1712304
α_i	1	48	1128	17296	194580	1712304

i	6	t = 7	t + 1 = 8
β_i	12271512	73629072	377348994
α_i	12271512	73629072	377342559

i	t+1=9	10	11
β_i	167710664	6540715896	22595200368
α_i	167688928	6537150048	22557347904

i	12	13
β_i	69668534468	192928249296
α_i	69214893604	187068034527

i	14	15
β_i	482320623240	1093260079344
α_i	425215758392	723133164517

i	16	17
α_i	2234848913647	4244421484512
β_i	734829498817	312343262689

i	18	19
β_i	7309837001104	11541847896480
α_i	33989748302	517825302

i	20	21
β_i	16735679449896	22314239266528
α_i	560079	21

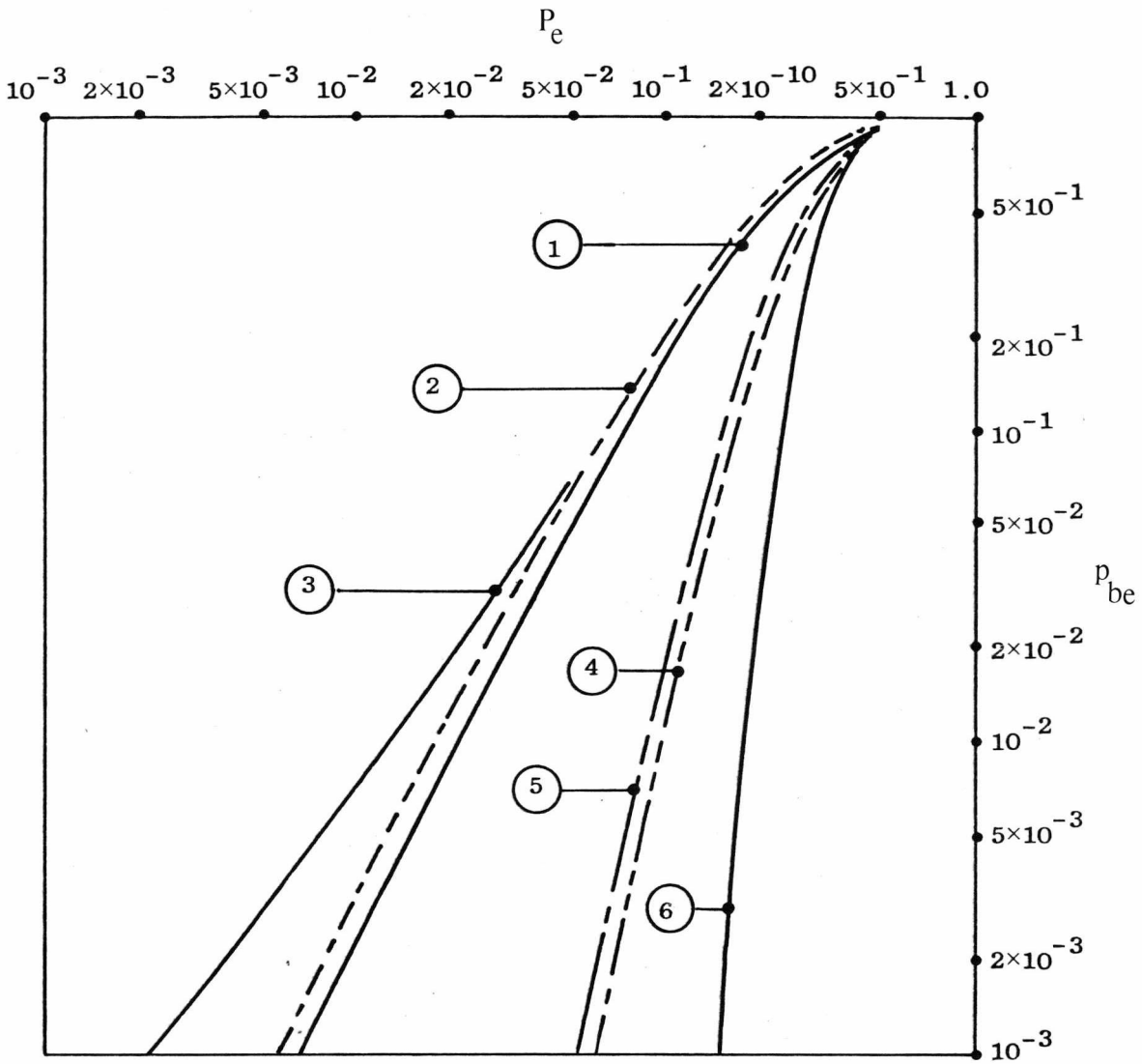
i	22 48
β_i	- ..... -
α_i	- ..... -

The block error probability of the above linear block codes has been computed, using the entries of the tables, and is plotted in Figure (II-1). In addition, the approximate block error probability of DC(9,5,3) has been compared with the computed block error probability based on the minimum distance consideration, given by:

$$P_{BE(min.dist.)} = 1 - \sum_{i=o}^{[(d-1)/2]} \binom{n}{i} p^i (1-p)^{n-i} \quad .$$

The results of the two computations are parallel straight lines for $p \leqslant 10^{-1}$ separated by about one tenth of an order of magnitude in error rate. The conclusion is that the difference between the two is almost entirely due to the error pattern of weight (t+1) which is not taken into account in the approximation of the minimum distance calculation.

Although DC(9,5,1) has minimum distance d = 1, the difference between the theoretical result and the computed one for $p \leqslant 3 \times 10^{-2}$ is very small. In spite of the difference between the Hamming distance of DC(23,6,10) and DC(23,6,7), the performance of the two codes has



① DC(9,5,3), estimate
② DC(9,5,3), theoretical (min. distance)
③ DC(9,5,1), estimate
④ DC(23,6,10), estimate
⑤ DC(23,6,7), estimate
⑥ DC(48,7,16) & DC(48,7,22) estimate

Figure (II-1) : Decoded Block Error Probability

only a small difference. The conclusion to be drawn from this is that although the total number of correctable error patterns

$$\sum_{i=o}^{n} \alpha_i$$

by DC(23,6,7) for all entires of the related table except for i = 4 is greater than those by DC(23,6,10); the slightly poorer performance of DC(23,6,7) is due to the difference in the number of more likely correctable errors of weight i = 4 in DC(23,6,10). Interestingly, this is the only case for which the α_i of DC(23,6,10) is bigger than the α_i of DC(23,6,7). Thus this makes the curve for DC(23,6,7), in spite of having a smaller Hamming distance, almost parallel to that for DC(23,6,10) over the range of error rate of interest. To obtain block error probabilities (bit-error probabilities) Olanigan and Turner (1976) employed combined systematic search and min.-distance decoding (Hybrid decoding), for several linear binary codes. The above results are similar to those results. However, the computation time for Hybrid decoding for fairly big k is rather excessive.

The computer program for calculation of the entires for the table is given below.

```
      DIMENSION PJ(23,23),JAW(23),Q(23),A(23),B(23),PC(23),TSUM(24)
      DIMENSION PE(24),TSUM2(24)
      INTEGER N
      DATA N/23/
      READ (7,110)(JAW(IC),IC=1,N)
110   FORMAT(1614/714)
      READ (7,100)(PE(I),I=1,24)
100   FORMAT (3F20.10)
      DO 2 J=1,N
      IF (JAW(J).EQ.0) GO TO 22
      IF (J.EQ.N) GO TO 111
      JR=(J+1)/2
      IF (JR.EQ.0) JR=1
      DO 4 I=1,N
      IF (I.LT.JR) GO TO 222
      SUM Q=0.0
      DO 6 JK=JR,J
      IF (JK.GT.I) GO TO 33
      A1=FACT(J)/(FACT(JK)*FACT(J-JK))
```

```
      B1=FACT(N-J)/(FACT(I-JK)*FACT(N-J-I+JK)
      C1=FACT(N)/(FACT(I)*FACT(N-I))
      W=(A1*B1)/C1
      IF(JK.NE.JR) GO TO 90
      W=W/2.0
   90 SUM Q= SUM Q+W
    6 CONTINUE
   33 PJ(J,I)=SUM Q
      IF(PJ(J,I).GT.1)PJ(J,I)=1.0
      GO TO 4
  222 PJ(J,I)=0.0
    4 CONTINUE
      GO TO 2
   22 DO 8 IN=1,N
    8 PJ(J,IN)=0.0
      GO TO 2
  111 IN=N/2
      DO 14 IJ=1,IN
   14 PJ(J,IJ)=0.0
      KN=(N/2)+1
      DO 18 IK=KN,N
   18 PJ(J,IK)=1.0
    2 CONTINUE
      DO 10 I-1,N
      AD=0.0
      DO 12 J=1,N
      IF(JAW(J).EQ.0)GO TO 12
      IF(PJ(J,I).EQ.1.0) GO TO 28
      AC=FLOAT(JAW(J))*ALOG10(1.0-PJ(J,I))
      AD=AD+AC
   12 CONTINUE
      Q(I)=10.0**AD
      GO TO 10
   28 Q(I)=0.0
   10 CONTINUE
      DO 16 I=1,N
      B(I)=FACT(N)/(FACT(I)*FACT(N-I))
      WRITE(2,160)I,b(I)
  160 FORMAT(1HO,2HB(,12,2H)=,E26.16)
      A(I)=B(I)*Q(I)
      IF (A(I).LT.1.0)A(I)=0.0
      WRITE(2,150)I,a(I)
  150 FORMAT(1HO,2HA(,12,2H)=,E26.16
   16 CONTINUE
      DO 3 IR=1,24
      WRITE(2,120)IR,PE(IR)
  120 FORMAT(1HO,10HCH. PROB.(,12,2H)=,E16.8)
      DO 24 JP=1,N
      PK=A(JP)
      IF(PK)41,41,5
    5 AA1=ALOG10(PK)+FLOAT(JP)*ALOG10(PE(IR))
      1+FLOAT(N-JP)*ALOG10(1.0-PE(IR))
      ASUMJ=AA1+10.0
      IF(ASUMJ)41,31,31
   41 PC(JP)=0.0
      GO TO 24
   31 PC(JP)=10**AA1
   24 CONTINUE
      TSUM(IR)=0.0
```

```
      DO 26 MM = 1,N
   26 TSUM(IR)-TSUM(IR)+PC(MM)
      WRITE(2,130)IR,TSUM(IR)
  130 FORMAT(1HO,23HCORRECT DECODING PROB.(,12,2H)=,E16.8)
      TS=FLOAT(N)*ALOG10(1.0-PE(IR))
      TTS=TS+10.0
      IF(TTS)9,7,7
    7 ST=10.0**TS
      TSUM(IR)=ST+TSUM(IR)
    9 TSUM2(IR)=1.0-TSUM(IR)
      WRITE(2,170)IR,TSUM2(IR)
  170 FORMAT(1HO,18HBLOCK ERROR PROB.(,12,2H)=,E16.8)
    3 CONTINUE
      STOP
      END



      FUNCTION FACT(N)
      FACT=1.0
      DO 1 I=1,N
      IX=I
    1 FACT=FACT*IX
      RETURN
      END
```

Appendix III

Anticode refinement computer program

This computer program was developed to perform the refinement of the anticodes constructed by the processes explained in Chapter VI. The anticode being refined is the AC(210,8,10) which is constructed as:

$$DAC(1+48,6,26) \overset{\alpha|}{+} F_{AC}(3,2,2) \overset{\alpha|\alpha'}{+} S_{AC}(63,8,32,2)$$

$$\equiv AC(210,8,10) .$$

Applying this program, the AC(213,8,11) results. The computer program is as follows below:

```
      DIMENSION GDAC(6,49), GIM(8,210),GSAC(6,63),IM(256,210),WT(256
      DIMENSION FAC(4,3),W(256),GDL(8,45),CHAR(45),DL(256,45),WSA(256)
      DIMENSION GFAC(2,3),WDL(256)
      INTEGER GDAC,GIM,GSAC,IM,FAC,W,Q,P,GDL,CHAR,DL,WSA,WT,DEL
      INTEGER GFAC,WDL
      DATA W,I1,I2,I3,I4,J1,J2,J3,J4/256*0,6,8,256,4,49,210,45,63/
      DATA 16,WDL/148,256*0/
      LSUM(Q,P)=Q+P-2*Q*P
      READ(7,100)((GFAC(I,J),J=1,3),I=1,2)
      READ(7,101)((GDAC(I,J),j=1,J1),I=1,I1)
      READ(7,102)((GSAC(I,J),J=1,J4),I=1,I1)
      READ(7,103)((GDL(I,J),J=1,J3),I=1,I2)
      READ(7,104)((FAC(I,J),J=1,3),I=1,I4)
  100 FORMAT(6I1)
  101 FORMAT(8OI1)
  102 FORMAT(8OI1)
  103 FORMAT(8OI1)
  104 FORMAT(12I1)
      WRITE(2,225)(GDL(I,J),J=1,J3),I=1,I2)
  225 FORMAT(8(3X,45I1/))
      K=0
      DO 2 I=1,J1
      DO 2 J=1,3
      K=K+1
      GIM(1,K)=GFAC(1,J)
    2 GIM(2,K)=GFAC(2,J)
      DO 4 I=1,6
      JK=0
      IK=I+2
      DO 4 J=1,J1
```

```
      IF(GDAC(I,J).EQ.1)GO TO 11
      DO 6 JJ=1,3
      JK=JK+1
    6 GIM(IK,JK)=O
      GO TO 4
   11 DO 8 JJ=1,3
      JK=JK+1
    8 GIM(IK,JK)=1
    4 CONTINUE
      DO 10 I=1,2
      DO 10 JS=I6,J2
   10 GIM(I,JS?=O
      DO 12 I=3,8
      IK=I-2
      DO 12 JS=I6,J2
      JK=JS-147
   12 GIM(I,JS)=GSAC(IK,JK)
      K=O
      DO 16 JS=1,J1
      DO 16 J=1,3
      K=K+1
      DO 16 I=1,4
   16 IM(I,K)=FAC(I,J)
      DO 18 I=1,4
      DO 18 JS=I6,J2
   18 IM(I,JS)=O
      WRITE(2,200)((GIM(I,J),J=1,110),I=1,8)
  200 FORMAT(8(3X,110I1/))
      WRITE(2,201)((GIM(I,J),J=111,210),I=1,8)
  201 FORMAT(8(3X,100I1/))
      DO 20 J=1,J2
      DO 22 I=3,I2
      K=(2**(I-1))+1
      KK=2**I
      IF(GIM(I,J).EQ.1)GO TO 15
      L=O
      DO 24 II=K,KK
      L=L+1
   26 IM(II,J)=IM(L,J)
      GO TO 22
   15 L=O
      DO 26 II=K,KK
      L=L+1
   26 IM(II,J)=LSUM(IM(L,J),1)
   22 CONTINUE
      DO 28 TJ=1,I3
   28 W(IJ)=W(IJ)+IM(IJ,J)
   20 CONTINUE
      WRITE(2,210)(W(I),I=1,I3)
  210 FORMAT(16(16I8/))
      DO 32 J=1,J3
      DL(1,J)=O
      DL(2,J)=GDL(1,J)
      DL(3,J)=GDL(2,J)
   32 DGL(4,J)=LSUM(GDL(1,J),GDL(2,J))
      DO 138 J=1,J3
      DO 38 I=3,12
      K=(2**(I-1))+1
```

```
      KK=2**I
      IF(GDL(I,J).EQ.1)GO TO 19
      L=0
      DO 40  II=K,KK
      L=L+1
 40   DL(II,J)=DL(L,J)
      GO TO 38
 19   L=0
      DO 42 II=K,KK
      L=L+1
 42   DL(II,J)=LSUM(DL(L,J),1)
 38   CONTINUE
      DO 72 N=1,I3
 72   WDL(N)=WDL(N)+DL(N,J)
138   CONTINUE
      WRITE(2,235)(WDL(N),N=1,I3)
235   FORMAT(16(16I8/))
      DEC=112
      IC=J3
      CHAR(1)=1
      CHAR(2)=1
      CHAR(3)=1
      DO 44 I=4,J3
 44   CHAR(I)=0
 52   DO 46 I-1,I3
 46   WSA(I)=0
      DO 49 J=1,J3
      IF(CHAR(J).EQ.0)GO TO 49
      DO 48 I=1,I3
      WSA(I)=WSA(I)+DL(I,J)
      WT(I)=WSA(I)+W(I)
      IF(WT(I).GT.DEL)GO TO 7
 48   CONTINUE
 49   CONTINUE
      GO TO 50
  7   CALL NEXCOM(CHAR,IC,IFIN)
      IF(IFIN.EQ.1)GO TO 25
      GO TO 52
 50   WRITE(2,220)(CHAR(I),I=1,J3)
220   FORMAT(3X,32HOPTIMAZATION BY COLUMN ADJACENCY,/,3X,45I1)
      WRITE(2,240)(WT(I),I=1,I3)
240   FORMAT(3X,23HNEW CODE W-DISTRIBUTION,/,(16(16I8/)))
      GO TO 56
 25   WRITE(2,215)(CHAR(I),I=1,J3)
215   FORMAT(3X,15HNO CONSTRUCTION,/,45I1)
 56   STOP
      END


      SUBROUTINE NEXCOM(CHAR,IC,IFIN)
      INTEGER CHAR(IC),P,P1
      IFIN=0
      DO 1 N=1,IO
      IF(CHAR(N).EQ.1)GO TO 2
  1   CONTINUE
  2   P=0
      N1=N+1
      IF(N1.GT.IO)GO TO 4
```

```
      DO 3 M=N1,IC
      IF(CHAR(M) .NE.1)GO TO 4
3     P=P+1
4     IF(P+N,EQ.10)GO TO 7
      N1=N1+P
      CHAR(N1)=1
      N1=N+P+1
      P1=P+1
      DO 5 M=1,P1
      N2=N1-M
5     CHAR(N2)=0
      IF(P.EQ.0)RETURN
      DO 6 M=1,P
5     CHAR(M)=1
      RETURN
7     IFIN=1
      RETURN
      END
```

REFERENCES

The following abbreviations, which are standard in electronics literature, are used in these references:-

AFCRC	:	Air Force Cambridge Research Center, Bedford, Mass.
BAMS	:	Bulletin of the American Mathematical Society.
BSTJ	:	Bell System Technical Journal.
CJM	:	Canadian Journal of Mathematics.
IBM	:	International Business Machines.
IBMJ	:	IBM Journal of Research and Development.
IEE	:	Institution of Electrical Engineers.
IEEE	:	Institute of Electrical and Electronics Engineers.
IC	:	Information and Control.
IRE	:	Institute of Radio Engineers.
IT	:	Information Theory
JCT	:	Journal of Combinatorial Theory.
MIT	:	Massachusetts Institute of Technology
SIAM	:	Society of Industrial and Applied Mathematics
SIAMJ	:	SIAM Journal on Applied Mathematics.

N. M. Abramson — A Class of Systematic Codes for Non Independent Errors - IRE Trans., IT-5, No. 4, p.150, Dec. 1959.

N. M. Abramson — Error Correcting Codes from Linear Sequential Circuits - The 4th London Symposium on Information Theory, Editor C. Cherry, Butterworths, p.26, 1960.

T. C. Ancheta, Jr. — Syndrome-Source-Coding and its Universal Generalization - IEEE Trans. IT-22, p.432, 1976

W. E. Barnes — Introduction to Abstract Algebra, D.C. Company, Boston. 1963

L. D. Baumert & R. J. McEliece — A Note on the Griesmer Bound - Corresp. of IEEE Trans., Info. Theo., p. 134, 1973.

E. R. Berlekamp — Algebraic Coding Theory - McGraw-Hill, New York, 1968.

E. R. Berlekamp — A Survey of Algebraic Coding Theory - Springer Verlag, New York, 1970.

E. R. Berlekamp — The Weight Enumerators for Certain Subcodes of the Second Order Binary Reed-Muller Codes - IC-17, p.485, 1970.

E. R. Berlekamp — Long Primitive Binary BCH Codes have $d \sim \frac{2n \ln R^{-1}}{\lg_{10} n}$ - IEEE Trans. IT-18, p.415, 1972.

S. D. Berman — Semi-simple Cyclic and Abelian Codes II, Cybernetic 3(3), p. 25, 1967.

G. Birkhoff & T. G. Bartee — Modern Applied Algebra - McGraw Hill Book Company, New York, 1970.

I. F. Blacke	Algebraic Coding Theory & History and Development - Dowden Hutchinson & Ross Inc., 1973.
R. C. Bose & D. K. Ray-Chaudhuri	On a Class of Error Correcting Binary Group Codes, IC-2, p.183, 1960a.
R. C. Bose & D. K. Ray-Chaudhuri	Further Results on Error Correcting Binary Group Codes - IC-3, p. 279 , 1960
D. M. Burton	Introduction to Modern Abstract Algebra, Addison-Wesley Company, 1967.
C. L. Chen & W. W. Peterson & E. J. Weldon Jr.	Some Results on Quasi-Cyclic Codes - IC-15, p. 407, 1969_a.
C. L. Chen	The Existence of Arbitrary Long Pseudo Cyclic Codes that meet the Gilbert Bound - Proc. 5^{th} Annual Princeton Conf., Info. Sc., p. 242, 1971.
R. T. Chien	A New Proof of the BCH Bound, IEEE Trans., IT-11, No. 4, p.541, July, 1972.
M. Cohen & A. Lempel	On Fast M-Sequence Transform. - Correspondence of IEEE Trans. on Infor. Theo., January, 1977.
G. Cohen & P. J. Godlewski	Residual Error Rate of Binary Block Codes - IEEE Trans., IT, p.483, July, 1976.
A. B. Cooper III	Algebraic Codes Constructed from Other Algebraic Codes; a Short Survey and Some Results. U.S. Army Material Systems Analysis Activity, Aberdeen providing ground, Maryland, 21005, USA, 1977.
P. Elias	Error Free Coding - IRE Trans., IT-4, p.29, Sept., 1954.
P. Elias	Coding for Noisy Channels - IRE Nat. Convention Record Part 4, p.37, 1955.

R. M. Fano	The Transmission of Information - MIT Press, Wiley, New York, 1961.
P. G. Farrell	Coding for Noisy Data Links, Ph.D. Thesis, University of Cambridge, 1969.
P. G. Farrell	Linear Binary Anticodes - Electronic Letters Vol. 6, No. 13, 25th June, 1970.
P. G. Farrell & A. Farrag	Further Properties of Linear Binary Anticodes - Electron. Letts. Vol. 10, No. 16, Aug. 1974.
P. G. Farrell	An Introduction to Anticodes - Chapter 3 in Algebraic Coding Theory and Application, Ed. G. Longo, Springer-Verlag, 1979.
P. G. Farrell, R. J. Smith & T. Meshkati	Array Codes - IEEE International Symposium on Info. Theo., June, 1979.
A. B. Fontaine & W. W. Peterson	Group Codes Equivalence and Optimum Codes, IRE Trans., IT-5, p.60. 1966
G. D. Forney Jr.	Concatenated Codes - MIT Press, Cambridge, Mass., U.S.A., 1966.
J. B. Fraleigh	A First Course in Abstract Algebra - Addison-Wesley Publishing Co, 1977.
R. G. Gallager	Information Theory and Reliable Communication - John Wiley & Sons, New York, 1968.
R. G. Gallager	Information Theory and Reliable Communication - Courses and Lectures, No. 30, Springer Verlag, Berlin, 1970.
E. N. Gilbert	A comparison of Signalling Alphabets, BSTJ, 31, p.504. 1952
M. J. E. Golay	Notes on Digital Coding - Proc. IRE, 37, p.657, 1949.

M. J. E. Golay — Binary Coding - IEEE Trans. - IT-4, p.23 1954.

M. J. E. Golay — Notes on the Penny Weighting Problem, Lossless Symbol Coding with Nonprimes, etc., IRE Trans., Infor. Theor., IT-4, p. 103, 1958.

H. D. Goldman & M. Klima H. Smola — The Weight Structure of Some Bose-Chaudhuri Codes - IEEE Trans., IT-14, No. 1, p. 16T, 1968.

S. W. Golomb — Shift-Register Sequences - Holden-Day Inc., San Francisco, California.1967

D. Gorenstien & W. W. Peterson & N. Zierler — Two Error-Correcting BCH Codes are Quasi-Perfect - IC-3, p.291, 1960.

D. Gorenstien & N. Zierler — A Class of Cyclic Linear Error-Correcting Codes in p^m Symbols, SIAMJ, Vol. 9, p. 207.

J. H. Griesmer — A Bound for Error Correcting Codes, IBMJ, 4, p. 532, 1960.

R. W. Hamming — Error Detecting and Error Correcting Codes, BSTJ, 29, p.147, 1950.

A. A. Hashim — Methods for Constructing and Decoding Block Error-Correcting Codes, Ph.D. Thesis, University of London.1974

H. J. Helgert & R. D. Stinaff — Minimum Distance Bounds for Binary Linear Codes, IEEE Trans., IT-19, No. 3, p.344, May,1973

C. F. Hobbs — Approximation of Performance of a Binary Group Code - Correspondence of IEE Trans. Infor. Theor., p.142, January 1965.

A. Hocquenghem — Codes Correcteurs d'Erreurs, Chiffres, 2, p. 147, 1959.

D. A. Huffman — A Method for the Construction of Minimum Redundance Codes, Proc. IRE, Vol. 40, No. 9, p.1098, Sept. 1952.

C. W. Hoffner & S. M. Reddy — Circulant Bases for Cyclic Codes, IEEE Trans., IT-16, No. 4, p. 511, 1970.

S. M. Johnson — A New Bound for Error-Correcting Codes, IRE Trans. Info. Theo., p. 203, April, 1962.

J. Jestesen — A Class of Constructive Asymptotically and Algebraic Codes, IEEE Trans., IT-18, p. 652, 1972.

M. Karlin — New Binary Coding Results by Circulants, IEEE Trans., IT-15, No. 1, Pt.I, p.81, 1969.

M. Karlin — Decoding of Circulant Codes - IEEE Trans., IT-16, p.797, 1970.

T. Kasami & S. Lin & W. W. Peterson — Some Results on Weight Distribution of BCH Codes, IEEE Trans., IT-12, p.274.1966

T. Kasami — An Upper Bound on k/n for Affine Invariant Codes with fixed d/n - IEEE Trans. IT-15, p.175, 1969.

T. Kasami & N. Tokura — Some Remarks on BCH Bounds and Minimum Weights of Binary Primitive BCH Codes, IEEE Trans., IT-15, p.408, 1969.

T. Kasami — A Gilbert-Varshumov Bound for Quasi-Cyclic Codes of Rate 1/2, IEEE Trans., IT-20, No.5, p.679, 1974.

J. Leech — Some Sphere Packing in Higher Space - Can. J. Math., 16, p.657, 1964.

J. Leech & N. J. A. Sloane — Sphere Packing and Error Correcting Codes - CJM-23, p.718, 1971.

S. K. Leung-Yan-Cheong & M. E. Hellman — Concerning a Bound on Undetected Error Probability - Corres. of IEEE, Trans. Info. Theo. p. 235, 1976.

V. I. Levenstien — On the Minimum Distance of Binary Error Correcting Codes, IC-28, p.268.1973

S. Lin & E. J. Weldon Jr. — Long BCH Codes are Bad - IC-11, p.445, 1967

S. Lin — An Introduction to Error-Correcting Codes - Prentice-Hall Inc., Englewood Cliffs, N. Jersey, U.S.A., 1970.

F. Loonstra — Introduction to Algebra - McGraw Hill Book Company, New York, 1967

R. W. Lucky & J. Salz & E. J. Weldon — Principles of Data Communication - McGraw Hill Book Comapny, New York, 1968.

J. E. MacDonald — Design Method for Maximum-Minimum Distance Error-Correcting Codes, IBM J., 4,p.43,1970.

F. J. MacWilliams & N. J. A. Sloane — Pseudo-Random Sequences and Arrays, Proc. IEEE, Vol.64, No. 12, p.1715, 1976.

F. J. MacWilliams & N. J. A. Sloane — The Theory of Error-Correcting Codes, North-Holland Publishing Company, 1978.

H. B. Mann — On the Number of Information Symbols in Bose-Chaudhuri Codes, IC-5, p.153, 1962.

J. L. Massey — Threshold Decoding - The MIT Press, Cambridge, Massachusetts, USA, 1963.

J. L. Massey — Shift-Register Synthesis and BCH Decoding, IEEE Trans., IT-15, p.122, 1969.

R. J. McEliece — On the Symmetry of Good Non-Linear Codes - IEEE Trans. IT-16, p.609, 1970.

R. J. McEliece & E. R. Rodeich & H. Rumsey Jr. & J. R. Welch — New Upper Bounds on the Rate of a Code via the Delsarte - MacWilliams Inequality. International Symposium on Info. Theo., Ronneby, Sweden. 1976

R. J. McEliece — The Theory of Information and Coding. Addison-Wesley Publishing Company, 1977.

Z. McC, Hutoon & A. M. Michelson — On the Computation of the Probability of Post Decoding Error Events for Block Codes - Corresp. of IEEE Trans. on Infor. Theory, p.399, 1977.

A. W. Nordstrom & J. P. Robinson — An Optimum Non-linear Code, IC-11, p.613. 1967.

D. O. Olanian & L. F. Turner — On the Error-Correcting Capability of Optimum Linear Block Codes - Proc. IEE, Vol. 123, No. 1, 1976.

J. M. Park Jr. — Inductive Proof of an Important Inequality - IEEE Trans., Info. Theor.,p.618, 1969.

W. W. Peterson & D. T. Brown — Cyclic Codes for Error Detection - Proc. IRE, Vol. 49, No. 1, p.228, 1961.

W. W. Peterson — Error-Correcting Codes. MIT Press, Cambridge, Second Edition, 1972.

J. N. Pierce — Limit Distribution of the Min. Distance of Random Linear Codes, IEEE Trans. Info. Theor., p.595, 1967.

V. S. Pless — On the Uniqueness of the Golay Codes, JCT-5, p.215, 1958.

V. S. Pless — On New Family of Symmetry Codes and Related New Five-Designs - BAMS, 75, p.1339, 1969.

V. S. Pless — Symmetry Codes over GF(3) and New Five-Designs. JCT-12, p.119, 1972.

M. Plotkin — Binary Codes with Specified Minimum Distance, IRE Trans. IT-6, p.445, 1960.

E. Prang — Cyclic Error-Correcting Codes in Two Symbols. AFCRC, TN-58-156, 1957.

E. Prang — Some Cyclic Error-Correcting Codes with Simple Decoding Algorithms. AFCRC, TN-57, 103, 1958.

F. M. Reza — An Introduction to Information Theory - McGraw Hill Book Company, 1961.

A. M. Rosie — Information and Communication Theory, Van Nostrand Reinhold, London, 1973.

I. S. Reed — A Class of Multiple-Error-Correcting Codes and Decoding Scheme. IRE Trans., Info.Theory, Vol. PG IT-4, p.38, 1954.

I. S. Reed & G. Solomon — Polynomial Codes over Certain Finite Fields, SIAMJ, Vol-8, p.300, 1960

G. E. Sacks — Multiple Error-Correction by means of Parity Checks - IRE Trans. IT-4, p.145.1954

J. Schonheim — On Linear and Non-Linear Single-Error-Correcting q-ary Perfect Codes, IC-12, p.23 1968.

C. E. Shannon — A Mathematical Theory of Communication, BSTJ, Vol-27, p.379.

C. E. Shannon — Communication in the Presence of Noise, Proc. IRE, Vol.37, No.1, p.10, 1949.

C. E. Shannon	Certain Results in Coding Theory for Noisy Channels - IC-1, No. 1, p.6, 1957.
C. E. Shannon	Probability of Error for Optimal Codes in the Gaussian Channel. BSTJ, Vol.38, No.3, p.611, 1959.
C. E. Shannon & R. G. Gallager & E. K. Berlekamp	Lower Bounds to Error Probability for Coding on Discrete Memoryless Channels - IC-10, p.522, 1967.
H. S. Shapiro & D. L. Slotnick	On the Mathematical Theory of Error-Correcting Codes. IBMJ, 3, p.25, 1959.
D. Slepian	A Class of Binary Signalling Alphabet, BSTJ, 35, p.23, 1956a.
D. Slepian	A Note on Two Binary Signalling Alphabets. IRE Trans., IT-2, p.84, 1956b.
D. Slepian	Some Further Theory of Group Codes. BSTJ, 39, p.1219, 1960..
D. Slepian	Key Paper in the Development of Information Theory, IEEE Press, 1973.
N. J. A. Sloane	A Short Course on Error-Correcting Codes, No. 188, Springer Verlag, 1975.
G. Solomon & J. J. Stiffler	Punctured Systematic Cyclic Coder, IEEE International Convention Record, Vol. 12, Pt. I, p.128, 1964.
G. Solomon & J. J. Stiffler	Algebraically Punctured Cyclic Codes, IC-8, p.170, 1965.
J. J. Stiffler	The Theory of Synchronous Communications, Prentice Hall, New Jersey, USA, 1971.
A. Tietavainen	On the non-existence of Perfect Codes over Finite Fields. SIAMJ. Appl. Math., Vol.24, p.88, 1973.

R. L. Townsend & E. J. Weldon Jr. — Self-Orthogonal Quasi Cyclic Codes - IEEE Trans., Vol. 13, p.183.1967

J. H. Van Lint — Coding Theory - Springer Verlag, Berlin, 1970.

R. R. Varshamov — Existence of the Number of Signals in Error-Correcting Codes - Doklay A.N.S.S.S.R. 117, No. 5, p.789 (I. F. Black, 1973, p.68), 1957.

Y. L. Vasilyev — On Non-Group Closed-Packed Codes - Prob. Cybernetic, p.337, (see also Black 1973), 1962.

T. J. Wagner — A Remark Concerning the Existence of Binary Quasi-Perfect Codes, IEEE Trans., IT-12, p.401. 1966a.

T. J. Wagner — A Search Technique for Quasi-Perfect Codes - IC-9, p.94, 1966b.

T. J. Wagner — Some Additional Quasi-Perfect Codes - IC-10, p.334, 1967.

N. Wax — On Upper Bounds for Error-Detecting and Error-Correcting Codes of Finite Length - IRE Trans., IT-5, p.168, 1959.

L. R. Welch & R. J. McEliece & H. Ramsey Jr. — A Low-Rate Improvement on the Elias Bound, IEEE Trans., IT-20, p.676. 1974

D. Wiggert — Error-Control Coding and Application, McLean Virginia Artech House, 1978.

J. K. Wolf — Adding Two Information Symbols to Certain Non-Binary BCH Codes and Some Applications - BSTJ, No.48, p.2405, 1969.

J. K. Wolf — Non-Binary Random Error Correcting Codes, IEEE Trans. IT-9, p.143, 1970.

J. K. Wolf — A Survey of Coding Theory: 1967-1972, IEEE Trans., IT-19, No. 4, p.381, 1973.

J. M. Wozencraft & I. M. Jacobs — Principals of Communication, John Wiley & Sons, New York, 1965.