1

of/de

1

PM-1 3½"x4" PHOTOGRAPHIC MICROCOPY TARGET
NBS 1010a ANSI/ISO #2 EQUIVALENT

PRECISIONSM RESOLUTION TARGETS

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file - Votre référence

Our file - Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

**GENERALISED LINEAR ANTICODES
AND OPTIMUM
ERROR-CORRECTING CODES**

by

WINNIE LIN, B.Sc

A thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfilment
of the requirements for the degree of

Master of Science

Information and Systems Science Committee
Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario, Canada
December 1995

copyright© W. Lin 1995

National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-612-09026-4

The undersigned recommend to
the Faculty of graduate Studies and Research
acceptance of the thesis

**GENERALISED LINEAR ANTICODES
AND OPTIMUM ERROR-CORRECTING CODES**

submitted by **WINNIE LIN**
in partial fulfilment of the requirments for
the degree of Master of Science

ACTING

Chair of Department

Thesis Supervisor

CARLETON UNIVERSITY

Date

Name WILLIAM LIA

Dissertation Abstracts International is arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation. Enter the corresponding four-digit code in the spaces provided.

COMPUTER SCIENCE

SUBJECT TERM

0000

SUBJECT CODE

U·M·I

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS

Architecture 0729
Art History 0377
Cinema 0900
Dance 0378
Fine Arts 0357
Information Science 0723
Journalism 0391
Library Science 0399
Mass Communications 0708
Music 0413
Speech Communication 0459
Theater 0465

EDUCATION

General 0515
Administration 0514
Adult and Continuing 0516
Agricultural 0517
Art 0518
Bilingual and Multicultural 0282
Business 0688
Community College 0275
Curriculum and Instruction 0727
Early Childhood 0518
Elementary 0524
Finance 0277
Guidance and Counseling 0519
Health 0680
Higher 0745
History of 0520
Home Economics 0278
Industrial 0521
Language and Literature 0279
Mathematics 0280
Music 0522
Philosophy of 0998
Physical 0523

Psychology 0525
Reading 0535
Religious 0527
Sciences 0714
Secondary 0533
Social Sciences 0534
Social Science of 0340
Special 0529
Teacher Training 0530
Technology 0710
Tests and Measurements 0288
Vocational 0747

LANGUAGE, LITERATURE AND LINGUISTICS

Language 0679
General 0289
Ancient 0290
Linguistics 0291
Modern 0401
Literature 0294
Classical 0295
Comparative 0297
Medieval 0298
Modern 0316
African 0591
American 0305
Asian 0352
Canadian (English) 0355
Canadian (French) 0593
English 0311
Germanic 0312
Latin American 0315
Middle Eastern 0313
Romance 0314
Slavic and East European

PHILOSOPHY, RELIGION AND THEOLOGY

Philosophy 0422
Religion 0318
General 0321
Biblical Studies 0319
Clergy 0320
History of 0322
Philosophy of 0469
Theology

SOCIAL SCIENCES

American Studies 0323
Anthropology 0324
Archaeology 0326
Cultural 0327
Physical 0310
Business Administration 0272
General 0770
Accounting 0454
Banking 0338
Management 0385
Marketing 0501
Canadian Studies 0503
Economics 0505
General 0508
Agricultural 0509
Commerce-Business 0510
Finance 0511
History 0358
Labor 0366
Theory 0351
Folklore 0357
Geography 0351
Gerontology 0578
History 0578
General

Ancient 0579
Medieval 0581
Modern 0582
Black 0326
African 0331
Asia, Australia & Oceania 0332
Canadian 0334
European 0335
Latin American 0336
Middle Eastern 0333
United States 0337
History of Science 0585
Law 0398
Political Science 0615
General 0616
International Law and Relations 0617
Public Administration 0814
Recreation 0452
Social Work 0626
Sociology 0627
General 0938
Criminology and Penology 0631
Demography 0628
Ethnic and Racial Studies 0629
Individual and Family Studies 0630
Industrial and Labor Relations 0700
Public and Social Welfare 0344
Social Structure and Development 0709
Theory and Methods 0999
Transportation 0453
Urban and Regional Planning 0453
Women's Studies

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES

Agriculture 0473
General 0285
Agronomy 0475
Animal Culture and Nutrition 0476
Animal Pathology 0359
Food Science and Technology 0478
Forestry and Wildlife 0479
Plant Culture 0480
Plant Pathology 0817
Plant Physiology 0777
Range Management 0746
Wood Technology 0306
Biology 0287
General 0308
Anatomy 0309
Biotistics 0379
Botany 0329
Cell 0353
Ecology 0369
Entomology 0793
Genetics 0410
Limnology 0307
Microbiology 0317
Molecular 0416
Neuroscience 0433
Oceanography 0821
Physiology 0778
Radiation 0472
Veterinary Science 0786
Zoology 0760
Biophysics 0425
General 0996
Medical

Geodesy 0370
Geology 0372
Geophysics 0373
Hydrology 0388
Mineralogy 0411
Paleobotany 0345
Paleoecology 0426
Paleontology 0418
Paleozoology 0985
Palynology 0427
Physical Geography 0368
Physical Oceanography 0415

HEALTH AND ENVIRONMENTAL SCIENCES

Environmental Sciences 0768
Health Sciences 0566
General 0300
Audiology 0992
Chemotherapy 0567
Dentistry 0350
Education 0769
Hospital Management 0758
Human Development 0982
Immunology 0564
Medicine and Surgery 0347
Mental Health 0569
Nursing 0570
Nutrition 0380
Obstetrics and Gynecology 0354
Occupational Health and Therapy 0381
Ophthalmology 0571
Pathology 0419
Pharmacology 0572
Pharmacy 0382
Physical Therapy 0573
Public Health 0574
Radiology 0575
Recreation

Speech Pathology 0460
Toxicology 0383
Home Economics 0386

PHYSICAL SCIENCES

Pure Sciences 0485
Chemistry 0749
General 0486
Agricultural 0487
Analytical 0488
Biochemistry 0738
Inorganic 0490
Nuclear 0491
Organic 0494
Pharmaceutical 0495
Physical 0754
Polymer 0405
Radiation 0605
Mathematics 0986
Physics 0606
General 0608
Acoustics 0748
Astronomy and Astrophysics 0607
Atmospheric Science 0798
Atomic 0759
Electronics and Electricity 0609
Elementary Particles and High Energy 0610
Fluid and Plasma 0752
Molecular 0756
Nuclear 0611
Optics 0463
Radiation 0346
Solid State 0984
Statistics 0984
Applied Sciences 0984
Applied Mechanics 0984
Computer Science

Engineering 0537
General 0538
Aerospace 0539
Agricultural 0540
Automotive 0541
Biomedical 0542
Chemical 0543
Civil 0544
Electronics and Electrical Heat and Thermodynamics 0348
Hydraulic 0545
Industrial 0546
Marine 0547
Materials Science 0794
Mechanical 0548
Metallurgy 0743
Mining 0551
Nuclear 0552
Packaging 0549
Petroleum 0765
Sanitary and Municipal System Science 0790
Geotechnology 0428
Operations Research 0796
Plastics Technology 0795
Textile Technology 0994

PSYCHOLOGY

General 0621
Behavioral 0384
Clinical 0622
Developmental 0620
Experimental 0623
Industrial 0624
Personality 0625
Physiological 0989
Psychobiology 0349
Psychometrics 0632
Social 0451

EARTH SCIENCES

Biogeochemistry 0425
Geochemistry 0996

ABSTRACT

In this thesis, a method for constructing optimum generalised linear block error-correcting codes is studied. This method is based on anticodes, which are codes constructed based on minimum maximum distance. When an anticode is deleted from the corresponding parent M-sequence code, a linear block code is obtained. If the anticode is optimum, then the code resulting from the deletion process will also be optimum. Furthermore, a large number of optimum codes can be obtained from one anticode.

This work focuses on generalised linear anticodes, particularly ternary and quaternary codes. Methodologies for generalised linear anticode construction are developed. A large number of optimum anticodes are constructed, which are obtained by manual scripting and computer program searching. From these, the codes constructed yield numerous known optimum linear block codes for $k \leq 6$ in $GF(3)$ and $k \leq 5$ in $GF(4)$. These results prove that constructing anticodes is an efficient and simple method for obtaining optimum and near-optimum error-correcting codes.

To:

My parents,

who gave me life, ambition and dreams

My husband,

who always stands by me

for better or for worse

My daughters,

who brighten my life

and will carry on my dreams

ACKNOWLEDGEMENT

I would like to thank my supervisor, Dr. T. A. Gulliver, for suggesting this interesting topic. His continual advice, guidance, understanding and encouragement during my entire graduate program are greatly appreciated.

I also want to thank my fellow graduate students in particular, Olivier Guieu and Craig Scratchley for their help and enlightening discussions and suggestions.

Special appreciation and thanks to my husband, John Shen, for his unwavering support, understanding and encouragement.

TABLE OF CONTENTS

Abstract.....	ii
Dedication.....	iii
Acknowledgments.....	iv
Table of Contents.....	v
 Chapter 1 Introduction.....	 1
1.1 General Introduction.....	1
1.2 Problem Definition and Thesis Objectives.....	3
1.3 Thesis Outline.....	4
 Chapter 2 Error-Correcting Codes.....	 6
2.1 Related Mathematical Concepts.....	7
2.1.1 Rings.....	7
2.1.2 Fields.....	8
2.2 Linear Block Codes.....	10
2.2.1 Minimum Distance.....	11
2.2.2 Generator Matrix.....	12
2.2.3 Parity Check Matrix.....	13
2.2.4 Encoding and Decoding.....	14
2.3 Optimum Codes.....	17
2.4 The Griesmer Bound.....	17
2.5 M-Sequence Codes.....	18
 Chapter 3 Linear Anticodes.....	 22
3.1 Anticode Concept.....	22
3.2 Linear Anticodes.....	24
3.3 Some Basic Properties of Linear Anticodes.....	27
3.4 Optimum Anticodes.....	30
3.5 Linear Anticode Construction Methods.....	30
3.5.1 Adjacency Column.....	30
3.5.2 Simple Stacking.....	32
3.5.3 Inversion Stacking.....	34

3.5.4 Combined Stacking.....	36
Chapter 4 Results.....	43
4.1 Anticode Computer Search.....	43
4.2 Linear Anticodes.....	46
4.2.1 Optimum Ternary Linear Anticode.....	49
4.2.2 Optimum Quaternary Linear Anticodes.....	52
4.2.3 Limitations of Generalised Anticode Construction.....	57
Chapter 5 Discussion and Conclusions.....	61
References.....	63

CHAPTER 1

INTRODUCTION

1.1 GENERAL INTRODUCTION

In recent years, there has been an increasing demand for efficient and reliable digital data transmission and storage systems. A merging of communications and computer technology is often required in the design of these systems. A major concern of the designer is the control of errors for reliable data reproduction at the receiver.

In 1948, Shannon [8] demonstrated in a landmark paper that, by proper encoding of the information, errors induced by a noisy channel or storage medium can be reduced to any desired level as long as data is not transmitted at a rate higher than the channel capacity. This entails adding some predetermined redundancy to the transmitted message so as to enhance the possibility of correctly interpreting it at the receiver. Here, the process of adding redundancy to the original message is called encoding, and the process of interpreting it is termed decoding. These processes are normally digital, so that error-control coding is usually considered as a branch of digital communications.

Since Shannon's work, a great deal of effort has been expended

on the problem of devising efficient encoding and decoding methods. Recent developments have contributed towards achieving the reliability required by today's high-speed digital systems, and the use of coding for error control has become an integral part of modern communication systems and digital computers. The study of error-correcting codes has led to the development of the now well established field of coding theory, which is concerned with the synthesis, encoding, decoding and performance of error detecting and correcting codes.

The set of all possible encoded messages is known as a code, or code book, and each encoded message is called a codeword [1]. If the codewords of a code are very different from each other, then the probability that the channel noise will change a transmitted codeword into another one is reduced. Thus, the error correcting capability of a code is directly related to the maximum difference, or *distance*, between any pair of codewords.

A *linear code* is defined by the following three basic parameters:

the number of information symbols in each codeword, k ;

the length of each codeword, n ;

the minimum distance between pairs of codewords, d .

The number of codewords is $N = q^k$, where q is the size of the symbol alphabet. The symbol alphabet and related parameters are defined in the next chapter. For efficient use of the communication

a code should ideally have as small a redundancy as possible, i.e. $n-k$ should be minimized; and the error-correcting capability of the code should be as large as possible, i.e. d should be maximised. From these considerations, an optimum code can be defined as one which has:

- i. the minimum n , for a given k and d ;
- or ii. the maximum d , for a given k and n .

These two criteria turn out to be equivalent, so that if one is achieved, then both are [1].

1.2 PROBLEM DEFINITION AND THESIS OBJECTIVES

Finding optimum codes for error control purposes is an important issue in coding theory. It is a very difficult problem which has defied general solution so far. The aim of this thesis is to construct optimum codes. This study concentrates on the synthesis of optimum codes and so is not concerned with implementation or performance.

In this thesis, anticode construction is presented as a means of finding optimum error-correcting codes. An anticode, which is an array of m columns and N rows, with a minimum maximum-distance δ between any pair of rows, has optimum parameters opposite to those of a code. An optimum anticode has a maximum m , for a given N and δ , or a minimum δ , for a given m and N .

By deleting or puncturing anticodes from the class of optimum parent codes (M-sequence codes), new codes are obtained. If the anticode used is optimum then the new codes obtained from the deletion process will also be optimum. Furthermore, a large number of optimum codes with different parameters can be obtained from one optimum anticode. This thesis is devoted to the study of anticode structures and their use in obtaining optimum codes.

1.3 THESIS OUTLINE

In chapter 2, the fundamental theory of error-correcting codes is presented. In Section 2.4, the Griesmer bound is presented. The relationship between M-sequence codes and linear error-correcting codes is presented in Section 2.5.

Chapter 3 is devoted to the study of anticodes. The anticode concept is described in detail in the first part of this chapter. Several anticode construction methods are presented and their roles in obtaining linear error-correcting codes are discussed.

In Chapter 4, the results obtained are presented. These include detailed constructions for many of the anticodes which have been obtained and the discussion about the limitations of generalised anticode construction. This chapter also includes a description of the algorithms used in the computer programs which

search for optimum anticodes.

Chapter 5 summarizes the results of the thesis, and lists several recommendations for further work on this topic.

CHAPTER 2

ERROR-CORRECTING CODES

The subject of error-correcting codes arose originally in response to practical problems in the reliable communication of digitally encoded information. Claude Shannon's paper "a Mathematical Theory of Communication" [8], written in 1948, started the discipline in electrical engineering called information theory. The branch of information theory called error-correcting codes was begun by Hamming in 1947 [20].

Coding theory deals with the problem of detecting and correcting transmission errors caused by noise in the channel. The following diagram depicts a general information transmission system:

Suppose a message is transmitted over a communications channel that is subject to some amount of noise. The object of an error-correcting code is to add redundancy to the message so that the original message can be recovered even if it has been corrupted by noise.

A brief review of the fundamental concepts of error-correcting codes is now presented.

2.1 RELATED MATHEMATICAL CONCEPTS

Rings, fields and their derivative mathematical structures are extensively used in the study of linear codes. In this section, the basic properties of rings and fields are reviewed.

2.1.1 RINGS

Let R be a non-empty set and define two operations, denoted by $+$ and $*$ (which may be quite different from ordinary addition and multiplication we are accustomed to). Then $(R, +, *)$ is a *ring* if for all a, b, c belonging to R , the following conditions are satisfied:

a) $a+b=b+a$	Commutative Law of $+$
b) $a+(b+c)=(a+b)+c$	Associative Law of $+$
c) There exists $z \in R$ such that
$a+z=z+a=a$ for every $a \in R$	Existence of an identity for $+$
d) For each $a \in R$ there is an
element $b \in R$ with $a+b=b+a=z$	Existence of inverses under $+$
e) $a*(b*c)=(a*b)*c$	Associative Law of $*$
f) $a*(b+c)=a*b+a*c$
$(b+c)*a=b*a+c*a$	Distributive laws of $*$ over $+$

A ring, R is called a commutative ring if its multiplicative operation is commutative. That is $a*b=b*a$, for any elements a and b belonging to R .

If there is an element $u \in R$ such that $u \neq 0$ and $a*u = u*a = a$ for all $a \in R$, then u is called the *unity*, or *multiplicative identity*, of R . R is then called a *ring with unity*.

Let R be a ring with unity u . If $a, b \in R$ and $a*b = b*a = u$, then b is called a *multiplicative inverse* of a and a is called a *unit* of R . (The element b is also a unit of R .)

2.1.2 FIELDS

A *field* is a commutative ring with unity in which every non-zero element is a unit. The *order* of a field is the number of elements contained in the field.

Finite fields, which have a finite order, are also called Galois fields, in memory of the French mathematician, Evariste Galois (1811-1832). Finite fields are denoted by $GF(q)$, where $q = p^m$ is the order of the field, p is a prime integer, and m is any integer > 0 .

If $p=2$, the elements of $GF(2)$ are "0" and "1". A code with symbols from this field is called a binary code. If the number of

elements in the field is larger than two, the code is called a *generalised* or *q-ary code*.

When the number of elements in the field is a prime, the elements are natural numbers and the operations of addition and multiplication obey the rules of arithmetic modulo the prime in question. For example, the elements of $GF(2)$ are 0 and 1 with addition and multiplication modulo 2 and the elements of $GF(11)$ are the numbers 0 to 10 with arithmetic modulo 11. This does not apply when the number of elements is a power of a prime. For example, the elements of $GF(4)$ may be denoted by 0, 1, 2, 3 with their arithmetic properties defined by the following two tables [19]:

Table 2.1 Addition and Multiplication Tables for $GF(4)$

+	0	1	2	3
0	0	1	2	3
1	1	0	3	2
2	2	3	0	1
3	3	2	1	0

*	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	3	1
3	0	3	1	2

The above tables are determined in the following way:
 Let $0 \leftrightarrow 0$, $1 \leftrightarrow 1$, $2 \leftrightarrow \alpha$, $3 \leftrightarrow \alpha^2$, and α is a root of the polynomial $x^2 + x + 1$ over $GF(2)$. Then $\alpha^2 = \alpha + 1$, and $\alpha^3 = \alpha^2 + \alpha = 1$.

2.2 LINEAR BLOCK CODES

There are two types of linear codes in common use today, block codes and convolutional codes. In this thesis, only block codes are considered.

In block coding, the information sequence is divided into blocks of k information symbols called a *message*. A message is represented by the k -tuple $u = (u_1, u_2, \dots, u_k)$. In a q -ary code, there are a total of q^k possible messages. The encoder transforms each message, which is the information k -tuple, independently into an n -tuple $v = (v_1, v_2, \dots, v_n)$ of symbols called a *codeword*. Therefore, corresponding to the q^k possible messages, there are q^k possible codewords at the encoder output. This set of q^k codewords of length n is called an (n, k) *block code* C [3]. For a block code to be useful, the q^k codewords must be distinct. Therefore, there should be a one-to-one correspondence between a message u and a codeword v .

A block code of length n with q^k codewords is called a linear (n, k) code if and only if its q^k codewords form a k -dimensional subspace of the vector space of all n -tuples over the Galois field, $GF(q)$ [3].

The ratio $R=k/n$ is called the *information or code rate* [1]. For a code to be useful (i.e., to have a different codeword

assigned to each message), $k \leq n$ or $R \leq 1$.

2.2.1 MINIMUM DISTANCE

Let $\mathbf{v} = (v_1, v_2, \dots, v_n)$ be an n -tuple. The *Hamming weight* (or simply *weight*) of $\mathbf{v}$ is defined as the number of non-zero symbols in $\mathbf{v}$ [1]. Let $\mathbf{v}$ and $\mathbf{w}$ be two n -tuples. The *Hamming distance* (or simply *distance*) between $\mathbf{v}$ and $\mathbf{w}$, denoted by $d(\mathbf{v}, \mathbf{w})$, is defined as the number of positions in which $\mathbf{v}$ and $\mathbf{w}$ differ [6] (containing "0" or not).

Given a block code C , the *minimum distance* of C , denoted by d , is defined as the minimum value of the distances between all pairs of codewords [1]. The minimum distance of a linear block code is equal to the minimum weight of its nonzero codewords [3]. Therefore, for a linear block code, determining the minimum distance of the code is equivalent to determining its minimum weight.

For a block code to be able to correct e errors in a codeword, the code must have minimum distance $d \geq 2e + 1$. To detect e errors, a minimum distance of at least $e + 1$ is required [3]. This is further explained in Section 2.3.4.

2.2.2 GENERATOR MATRIX

Since any (n, k, d) linear block code is a k -dimensional subspace of the vector space V_n over $GF(q)$, it can be generated by a set of k independent n -tuples. These n -tuples can be arranged in the form of a $k * n$ matrix called the *generator matrix*, denoted by G . The codewords are then linear combinations of the k rows of G . An (n, k, d) linear code is completely specified by the k rows of G .

A desirable property of a linear block code is a *systematic* structure, that is any codeword has k information symbols (message) and $n-k$ parity check symbols which are a linear sum of the information symbols. In this case the generator matrix has the following form [1]:

$$G = [I_k \mid g]$$

where I_k is the $k * k$ identity matrix and g is a $k * (n-k)$ matrix which depends on the particular code.

If $u = (u_1, u_2, \dots, u_k)$ is the k -tuple message block to be encoded, the corresponding n -tuple codeword, $v = (v_1, v_2, \dots, v_n)$, can be given as follows [3]:

$$v = u \cdot G$$

where u is an $1 * k$ row vector, G is the $k * n$ generator matrix of the code, and v is an $1 * n$ row vector. The symbols of u and v and the entries of G are elements of $GF(q)$, and the addition and

multiplication operations are defined over the elements of $GF(q)$.

The generator matrix of an (n, k, d) linear code is not unique. Any two generator matrices are equivalent if one can be obtained from the other by a combination of elementary row operations and column permutations. In this way, any generator matrix can be transformed to systematic form (SF), i.e., the form $[I|P]$, where I is a $k \times k$ identity matrix.

2.2.3 PARITY CHECK MATRIX

For any $k \times n$ matrix G over $GF(q)$ with k linear independent rows, there exists an $(n - k) \times n$ matrix H over $GF(q)$ with $n - k$ linear independent rows such that any vector in the row space of G is orthogonal to the rows of H and any vector that is orthogonal to the rows of H is in the row space of G [3]. Hence, the (n, k, d) linear code generated by G can be described in an alternative way as follows : An n -tuple v is a codeword of the code C generated by G if and only if $v^T \cdot H = 0$. This matrix H is called a *parity check matrix* of the code, and in systematic form is given by [1]:

$$H = [h \mid I_{n-k}]$$

where I_{n-k} is an $(n - k) \times (n - k)$ identity matrix, and h is an $(n - k) \times k$ matrix which depends on the structure of the code.

Since the row space of G is the null space of H and vice versa, $G \cdot H^T = 0$, where H^T is the transpose of H . Hence the sub-

matrix g contained in G is equal to $-h^T$, where h^T is the transpose of the sub-matrix h contained in H .

2.2.4 ENCODING AND DECODING

In this section, an example [7] is given to demonstrate the process of encoding and decoding. The roles of the generator matrix G and parity check matrix H , as well as the relation between the minimum distance d and the error-correcting capability of a code, are explained.

Suppose we have a binary information sequence:

(0 0 0 1 0 0 0 1 0 0 0 1 1 1 0 1 0 1 0 1 1 1 1 1).

First, divide this sequence into 6 messages, each of which has 3 information symbols. The set of messages are:

$P = \{ 000, 100, 010, 001, 110, 101, 011, 111 \}$.

Let

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

be a 3×6 generator matrix which defines the encoding function $E: P \rightarrow C$, i.e., $E(P) = P * G = C$. After encoding, each 3-tuple message is transformed into an 6-tuple codeword. For example,

$$E(110) = [110] G = [110] \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{array} \right] = [110101],$$

and

$$E(010) = [010] G = [010] \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} = [010011].$$

The set of codeword is

$$C = \{000000, 100110, 010011, 001101, 110101, 101011, 011110, 111000\}.$$

Note that the minimum distance between codewords is 3, so we can detect errors of weight ≤ 2 and correct all single errors.

The $(6-3)*6$ parity check matrix H is

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

We can identify r as a codeword if $r^T * H = 0$. Suppose $r = 110110$ is received. The product $r^T * H$, is called the *syndrome* of r . For this example,

$$r^T * H = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix},$$

so that r is not a codeword. comparing r to the list of codewords, we see that $d(100110, r) = 1$. For all other codewords $c \in C$, $d(r, c) \geq 2$. Writing $r = c + e = 100110 + 010000$, we find that the transmission error (of weight 1) occurs in the second component of

r. Changing the second component of r, we get c, and the message comprises the first three components of c.

Let $r = c + e$, where c is a codeword and e is an error pattern of weight 1. Suppose that 1 is in the i th component of e, $1 \leq i \leq 6$, and all other components are 0. Then

$$r^T * H = (c + e)^T * H = (c^T + e^T) * H = c^T * H + e^T * H.$$

If c is a codeword, it follows that $c^T * H = 0$, so $= e^T * H$ ith column of matrix H.

Suppose that we receive $r = 000111$. Computing the syndrome

$$r^T * H = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix},$$

we obtain a result that is not one of the columns of H. Yet $e^T * H$ can be obtained as the sum of two columns from H. If $e^T * H$ came from the first and sixth columns of H, correcting these components in v results in the codeword 100110. If we sum the third and fifth columns of H to get this syndrome, upon changing the third and fifth components of r we get a second codeword, 001101. So we can not expect H to correct multiple errors. This is not surprising since the minimum distance between codewords is 3.

2.3 OPTIMUM CODES

An optimum code has maximum Hamming distance d for a given block length n and dimension k . Since there are three defining parameters for any code, namely n , k and d , an optimum code can be described in the following three ways:

- i. The code has the maximum d , for the given n and k ;
- ii. The code has the minimum n , for the given d and k ;
- iii. The code has the maximum k , for the given n and d .

There is no general method for constructing an optimum code given a set of parameters. Numerous methods have been devised for constructing either certain classes of optimum codes with a limited parameter range, or special classes of codes with predetermined parameters. The anticode method considered in this thesis is one means of obtaining optimum codes.

2.4 THE GRIESMER BOUND

A lower bound on n for the case of high redundancy or low rate codes was presented by Griesmer in 1960. Solomon and Stiffler generalised the Griesmer bound over $GF(q)$, and described a class of low rate codes which meet the Griesmer bound [11].

The generalised Griesmer bound shows that the minimum value of

n for which it is possible to obtain an (n, k, d_0) linear code is given by

$$n \geq d_0 + d_1 + \dots + d_{k-1}$$

where

$$d_i = \left\lceil \frac{d_{i-1} + q - 1}{q} \right\rceil$$

and $|x|$ denoted the largest integer in x .

Codes which meet the Grisemer bound are optimum.

2.5 THE M-SEQUENCE CODES

The $N=q^k$ codewords of a linear block error-correcting code can be tabulated as the rows of an $N \times n$ matrix, called the *code book* of the code. Each element of this matrix belongs to the Galois field, $GF(q)$. The N rows of the code book are the codewords; the n columns of the code book are called the *code columns*. There are k information columns, and $n-k$ redundant columns which are linear combinations of the k information columns if the code is in SF [1].

The linear code that results from a linear code book consisting of all possible non-zero columns for a given k has the following parameters:

$$n = q^k - 1$$

$$d = (q-1)q^{k-1}$$

and $R = k/n = k/(q^k-1).$

This code is called a maximum-length-sequence or *M-sequence code*

[2]. Its generator matrix has all possible distinct k -tuples as columns (except the all zero column). The M-sequence codes satisfy the Griesmer bound with equality, and so are optimum codes.

For example, consider the $(n, k, d) = (7, 3, 4)$ binary M-sequence code, which has the code book shown below:

	Column						
	1	2	3	4	5	6	7
$k = 3$	0	0	0	0	0	0	0
$n = 2^k - 1 = 7$	0	0	1	0	1	1	1
$d = 2^{k-1} = 4$	0	1	0	1	0	1	1
$N = 2^k = 8$	0	1	1	1	1	1	0
	1	0	0	1	1	0	1
	1	0	1	1	0	1	0
	1	1	0	0	1	1	0
	1	1	1	0	0	0	1

This binary code has eight codewords, the all zero codeword and seven of weight four. It has three information columns: 1, 2 and 3, and four redundancy columns: 4 to 7, which consist of all possible linear combinations of the three information columns:

$$\text{Column 4} = \text{Column 1} +_2 \text{Column 2}$$

$$\text{Column 5} = \text{Column 1} +_2 \text{Column 3}$$

$$\text{Column 6} = \text{Column 2} +_2 \text{Column 3}$$

$$\text{Column 7} = \text{Column 1} +_2 \text{Column 2} +_2 \text{Column 3}$$

The notation $+_q$ denotes the operation of symbol-by-symbol addition in $\text{GF}(q)$. The notation $*_q$ denotes the operation of symbol-by-symbol multiplication in $\text{GF}(q)$.

Now consider the quaternary M-sequence code with parameters

$(n, k, d) = (15, 2, 12)$, which has the following code book:

		Column														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$k = 2$ $n = 4^k - 1 = 15$ $d = (4-1)4^{k-1} = 12$ $N = 4^k = 16$		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		0	1	1	2	3	1	2	3	1	2	3	0	2	0	3
		0	2	2	3	1	2	3	1	2	3	1	0	3	0	1
		0	3	3	1	2	3	1	2	3	1	2	0	1	0	2
		1	0	1	1	1	2	2	2	3	3	3	2	0	3	0
		1	1	0	3	2	3	0	1	2	1	0	2	2	3	3
		1	2	3	2	0	0	1	3	1	0	2	2	3	3	1
		1	3	2	0	3	1	3	0	0	2	1	2	1	3	2
		2	0	2	2	2	3	3	3	1	1	1	3	0	1	0
		2	1	3	0	1	2	1	0	0	3	2	3	2	1	3
		2	2	0	1	3	1	0	2	3	2	0	3	3	1	1
		2	3	1	3	0	0	2	1	2	0	3	3	1	1	2
		3	0	3	3	3	1	1	1	2	2	1	0	2	0	0
		3	1	2	1	0	0	3	2	3	0	1	1	2	2	3
		3	2	1	0	2	3	2	0	0	1	3	1	3	2	1
		3	3	0	2	1	2	0	3	1	3	0	1	1	2	2

This $(15, 2, 12)$ quaternary M-sequence code has sixteen codewords with weight twelve. Columns 1 and 2 are information columns; columns 3 to 15 are redundancy columns, which are all possible linear combinations of columns 1 and 2:

$$\text{Column 3} = \text{Column 1} +_4 \text{Column 2}$$

$$\text{Column 4} = \text{Column 1} +_4 2 *_2 \text{Column 2}$$

$$\text{Column 5} = \text{Column 1} +_4 3 *_2 \text{Column 2}$$

$$\text{Column 6} = 2 *_4 \text{Column 1} +_4 \text{Column 2}$$

$$\text{Column 7} = 2 *_4 \text{Column 1} +_4 2 *_4 \text{Column 2}$$

$$\text{Column 8} = 2 *_4 \text{Column 1} +_4 3 *_4 \text{Column 2}$$

$$\text{Column 9} = 3 *_4 \text{Column 1} +_4 \text{Column 2}$$

$$\text{Column 10} = 3 *_4 \text{Column 1} +_4 2 *_4 \text{Column 2}$$

$$\text{Column 11} = 3 *_4 \text{Column 1} +_4 3 *_4 \text{Column 2}$$

$$\text{Column 12} = 2 *_4 \text{Column 1}$$

Column 13= $3 *_4$ Column 1

Column 14= $2 *_4$ Column 2

Column 15= $3 *_4$ Column 2.

Addition $+_4$ and multiplication $*_4$ in $GF(4)$ are defined in Section 2.1.2.

A convenient way of obtaining linear block codes without repeated columns is by puncturing (or deleting) columns from the M-sequence code, without altering the number of codewords. Any columns may be removed, provided at least k columns remain. If an information column is removed, then the order of the codewords can be re-arranged to restore the code book to systematic form. This idea was first suggested by Solomon and Stiffler in 1964 [10]. The following chapter presents a construction method based on this idea, called the anticode method.

CHAPTER 3

LINEAR ANTICODES

3.1 ANTICODE CONCEPT

In the previous chapter, M-sequence codes were defined. They are optimum linear codes with code books made up of all possible non-zero column (without any repeated columns). The idea of puncturing columns from an M-sequence code book was also introduced. By partitioning off the punctured columns from an M-sequence code book, another code is found with reduced block length and minimum distance, but having the same number of codewords as the "parent" M-sequence code.

The M-sequence codes are optimum, and all the codewords have weight q^{k-1} , except the all-zero word. The ultimate intention of puncturing or deleting columns is to construct an optimum code with shorter length. Thus, if m columns are to be punctured, it is desirable that the maximum weight, δ , of the rows of the punctured columns should be as small as possible, as this maximum value of row weight will determine the minimum weight, and hence the minimum

distance, of the resulting code. If a set of m columns, with maximum row weight δ , is punctured from an M-sequence code with k information digits, then a code with block length $q^{k-1}-m$ and minimum distance $q^{k-1}-\delta$ results [2].

The set of punctured columns should have a minimum value of δ for a given m ; alternatively, for a given value of δ , a maximum value of m is sought. These specifications are exactly the opposite of those of a code, which are a maximum value of d for a given n , or a minimum value of n for a given d . Hence, the set of punctured columns is called an *anticode*.

An anticode is an array of m columns and $N=q^k$ rows, in which every row is composed of m digits taken from $GF(q)$. The q^k rows of the anticode are the *codewords*; its m columns are the *anticode columns*. In the anticode, the distance between any pair of rows is equal to or less than a certain value, δ , which is the *maximum distance* of the anticode. An anticode is optimum if it has the *minimum* value of δ for a given m and k .

3.2 LINEAR ANTICODES

An M-sequence code is a linear block code which has no repeated columns. The columns of the M-sequence code consist of k independent information columns and all possible distinct linear combinations of these k columns. Since the anticode is punctured from the M-sequence code, typically the anticode has no repeated columns. All the columns of the anticode are distinct linear combinations of the k information columns of the parent M-sequence code. Thus an anticode with parameters (m, k, δ) punctured from an M-sequence code is called a *linear anticode*.

The words of an anticode form a group, like those of a linear code. So it follows that the maximum weight of the anticode is equal to its maximum distance, δ . A linear anticode is defined to be optimum if it has minimum δ , that is the minimum maximum distance, for a given k and m . In other words, a linear anticode is optimum if, when punctured from its parent M-sequence code, it will produce an optimum linear code. This optimum code will have a maximum minimum distance d for a given k and n .

Since the parent M-sequence code is optimum, if the punctured anticode is optimum, then the resulting code is also optimum. Any anticode with parameters (m, k, δ) will be denoted by $AC(m, k, \delta)$ in this thesis, and the corresponding code by $RC(n, k, d)$.

Consider the (7,3,4) binary M-sequence code given in Chapter 2, which has the code book shown below:

	1	2	3	4	5	6	7
$k = 3$	0	0	0	0	0	0	0
	0	0	1	0	1	1	1
$n = 2^k - 1 = 7$	0	1	0	1	0	1	1
	0	1	1	1	1	1	0
$d = 2^{k-1} = 4$	1	0	0	1	1	0	1
	1	0	1	1	0	1	0
$N = 2^k = 8$	1	1	0	0	1	1	0
	1	1	1	0	0	0	1

columns 4, 5 and 6 produce an anticode with parameters $(m, k, \delta) = (3, 3, 2)$, which is optimum. If this anticode is deleted from the above parent M-sequence code then an optimum linear code with parameters (4,3,2) is obtained:

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

$$AC(m, k, \delta) = (3, 3, 2)$$

$$RC(n, k, d) = (4, 3, 2)$$

As a second example, consider the (15,2,12) quaternary M-sequence code in Chapter 2:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$k = 2$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	1	1	2	3	1	2	3	1	2	3	0	2	0	3
$n = 4^k - 1 = 15$	0	2	2	3	1	2	3	1	2	3	1	0	3	0	1
	0	3	3	1	2	3	1	2	3	1	2	0	1	0	2
$d = (4-1)4^{k-1} = 12$	1	0	1	1	1	2	2	2	3	3	3	2	0	3	0
	1	1	0	3	2	3	0	1	2	1	0	2	2	3	3
$N = 4^k = 16$	1	2	3	2	0	0	1	3	1	0	2	2	3	3	1
	1	3	2	0	3	1	3	0	0	2	1	2	1	3	2
	2	0	2	2	2	3	3	3	1	1	1	3	0	1	0
	2	1	3	0	1	2	1	0	0	3	2	3	2	1	3
	2	2	0	1	3	1	0	2	3	2	0	3	3	1	1
	2	3	1	3	0	0	2	1	2	0	3	3	1	1	2
	3	0	3	3	3	1	1	1	2	2	2	1	0	2	0
	3	1	2	1	0	0	3	2	3	0	1	1	2	2	3
	3	2	1	0	2	3	2	0	0	1	3	1	3	2	1
	3	3	0	2	1	2	0	3	1	3	0	1	1	2	2

If the anticode $AC(10,2,8)$, which is optimum (because the corresponding $(5,2,4)$ code meets the Griesmer bound), is deleted from the above M-sequence code then an optimum linear code with parameters $(5,2,4)$ results. In this case, the $(10,2,8)$ anticode consists of columns 1 to 8, 14 and 15, and the $(5,2,4)$ code consists of columns 9 to 13:

```

0 0 0 0 0 0 0 0 0 0
0 1 1 2 3 1 2 3 0 3
0 2 2 3 1 2 3 1 0 1
0 3 3 1 2 3 1 2 0 2
1 0 1 1 1 2 2 2 3 0
1 1 0 3 2 3 0 1 3 3
1 2 3 2 0 0 1 3 3 1
1 3 2 0 3 1 3 0 3 2
2 0 2 2 2 3 3 3 1 0
2 1 3 0 1 2 1 0 1 3
2 2 0 1 3 1 0 2 1 1
2 3 1 3 0 0 2 1 1 2
3 0 3 3 3 1 1 1 2 0
3 1 2 1 0 0 3 2 2 3
3 2 1 0 2 3 2 0 2 1
3 3 0 2 1 2 0 3 2 2

```

$AC(k,m,\delta)=(10,2,8)$

```

0 0 0 0 0
1 2 3 0 2
2 3 1 0 3
3 1 2 0 1
3 3 3 2 0
2 1 0 2 2
1 0 2 2 3
0 2 1 2 1
1 1 1 3 0
0 3 2 3 2
3 2 0 3 3
2 0 3 3 1
2 2 2 1 0
3 0 1 1 2
0 1 3 1 3
1 3 0 1 1

```

$RC(n,k,d)=(5,2,4)$

Consider the following example, which results in another optimum linear code by the anticode method. If the quaternary anticode $(80,4,62)$, which has been constructed by the author, is deleted from its parent M-sequence code with parameters $(255,4,192)$, a linear code with parameters $(175,4,130)$ is obtained. The Griesmer bound shows that a quaternary code with $k = 4$ and maximum-minimum distance equal to 130, might exist with block length $n \geq 175$. This means that the linear quaternary code $(175,4,130)$, produced by the anticode method, is optimum.

3.3 SOME BASIC PROPERTIES OF LINEAR ANTICODES

All q^k codewords of an M-sequence code are different. In the case of an anticode, it may consist of only q^j distinct words, where $1 \leq j \leq k$, and then each distinct word will appear q^{k-j} times in the anticode [2]. For example, this $q=2$, $k=3$, $m=3$, $\delta=2$ anticode has $j=2$, i.e. each word appears twice:

000	
110	$q=2$
101	$k=3$
011	$m=3$
011	$\delta=2$
101	$j=2$
110	
000	

This rather startling property of anticodes is in direct opposition to that of codes, which should not have indistinct words.

A linear anticode is a subspace of the vector space V_m . Hence a set of basic vectors (codewords) can be found to describe the subspace, that is, a linear anticode can be described by means of a generator matrix in exactly the same way as a linear code. Each column in the parent M-sequence code book corresponds to a column in the M-sequence code generator matrix. The generator matrix of the anticode is formed from the columns of the M-sequence code generator matrix corresponding to those columns punctured from the M-sequence code book. The anticode generator matrix will not necessarily be in systematic form (SF), but it can be transformed to SF form. If the anticode has repeated columns (that is, if $j < k$),

then the SF generator matrix has $k-j$ all-zero rows, and a $j \times j$ identity matrix instead of a $k \times k$ identity matrix [2].

For example, the $q=2$, $k=3$ M-sequence code generator matrix is:

$$[G_{MS}] = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{array} \right]$$

The three right-hand columns of this matrix form the generator matrix of a $q=2$, $k=3$, $m=3$, $\delta=2$ anticode:

$$[G_{AC}] = \left[\begin{array}{ccc} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{array} \right]$$

Converting to SF:

$$[G_{AC-SF}] = \left[\begin{array}{ccc|c} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 \end{array} \right]$$

The all-zero row and 2×2 identity matrix arise from the fact that $j=2$ in this case.

The four right-hand columns of G_{MS} form the generator matrix of a $q=2$, $k=3$, $m=4$, $\delta=3$ anticode:

$$[G_{AC}] = \left[\begin{array}{cccc} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{array} \right], \quad [G_{AC-SF}] = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{array} \right]$$

In this case $j=3$, so there are no all-zero rows. If the generator matrix of an anticode can be converted, by elementary row operations and column permutation, into that of another anticode, then these two anticodes have the same δ and are said to be *equivalent*. This result parallels that for equivalent codes. The properties of codes and anticodes are compared in Table 3.1 [2].

Table 3.1 A Comparison Of The Properties
Of Codes And Anticodes

CODE	ANTICODE
- block length n	- block length m
- minimum distance d : $d = \min(d_{ij}), i \neq j,$ $= \min(w_i)$ for a linear code	- maximum distance δ $\delta = \max(d_{ij}), i \neq j$ $= \max(w_i)$ for a linear anticode
- optimum if has the greatest d for a given n and N (or k)	- optimum if has the least δ for a given m and N (or k)
- deleting a column in general reduces d by one	- deleting a column in general does not reduce δ
- adding a column does not in general increase d	- adding a column in general does increase δ
- the words of a code are all distinct; hence the $[G]$ matrix of a code has no all-zero rows	- an anticode may have repeated rows; hence the $[G]$ matrix of an anticode may have all-zero rows

3.4 OPTIMUM ANTICODES

An anticode is considered to be optimum in this thesis for one of the following reasons:

- i. it produces a linear code which has already been found and is known to be optimum;
- ii. it produces a linear code which has parameters which meet a known bound. i.e., the Griesmer bound.

3.5 LINEAR ANTICODE CODE CONSTRUCTION METHODS

In this section, various methods of anticode construction are presented, which in turn lead to code constructions. Any anticode with parameters (m, k, δ) , without repeated rows and columns, will be called a *basic anticode* if used as a unit in another structure to produce a new anticode. Each of the anticode construction methods will be studied and their properties, advantages and disadvantages will be discussed.

3.5.1 ADJACENCY COLUMN

Before presenting the anticode constructions, the concept of an adjacency column is presented, because this plays an important role in anticode construction.

A *basic* adjacency column is a column which contains q^{k-1} of each element in the Galois field $GF(q)$ and these elements appear in their natural order, which is:

$$[0 \dots 0 \mid 1 \dots 1 \mid \dots \mid q-1 \dots q-1]^T$$

Multiply this basic column by one element of $GF(q)$ other than 0 or 1, results in another adjacency column. There are $q-2$ elements beside 0 and 1 in $GF(q)$, so there are $q-1$ possible adjacency columns.

For example, a basic quaternary adjacency column for $k=3$ is a set of 4^* zeros, 4^* ones, 4^* twos and 4^* threes:

$$[0000000000000000111111111111111122222222222222223333333333333333]^T$$

Multiplying this basic column by 2 and 3 (according the multiplication table for $GF(4)$ given in Section 2.1.2), results in the other two adjacency columns:

$$[0000000000000000222222222222222233333333333333331111111111111111]^T$$

$$[0000000000000000333333333333333311111111111111112222222222222222]^T$$

There are two ($3-1=2$) adjacency columns in $GF(3)$:

$$[0 \dots 0 \mid 1 \dots 1 \mid 2 \dots 2]^T$$

$$[0 \dots 0 \mid 2 \dots 2 \mid 1 \dots 1]^T$$

There are three ($4-1=3$) adjacency columns in $GF(4)$:

$$[0 \dots 0 \mid 1 \dots 1 \mid 2 \dots 2 \mid 3 \dots 3]^T$$

$$[0 \dots 0 \mid 2 \dots 2 \mid 3 \dots 3 \mid 1 \dots 1]^T$$

$$[0 \dots 0 \mid 3 \dots 3 \mid 1 \dots 1 \mid 2 \dots 2]^T$$

3.5.2 SIMPLE STACKING

The fact that an anticode may have repeated rows was discussed in Section 3.3. Thus a legitimate way of constructing an anticode is to repeat or stack the anticode on itself several times. This process is called *simple stacking*. For example,

$$\begin{array}{ccc}
 \begin{array}{c} 0\ 0\ 0 \\ 0\ 1\ 1 \\ 1\ 0\ 1 \\ 1\ 1\ 0 \\ AC(3,2,2) \end{array} & \xrightarrow{\text{-----}} & \begin{array}{c} 0\ 0\ 0 \\ 0\ 1\ 1 \\ 1\ 0\ 1 \\ 1\ 1\ 0 \\ \text{-----} \\ 0\ 0\ 0 \\ 0\ 1\ 1 \\ 1\ 0\ 1 \\ 1\ 1\ 0 \\ AC(3,3,2) \end{array}
 \end{array}$$

This $(3,3,2)$ binary anticode is a simple stack of the anticode $(3,2,2)$, which is stacked onto itself twice. It is obvious that anticodes with parameters $(3,k,2)$ will exist and can be constructed for any $k \geq 2$ by the process of simply stacking the anticode $(3,2,2)$, 2^{k-2} times on itself. Thus for all $k \geq 2$, binary linear codes with parameters $n = (2^k - 1) - 3$ and $d = 2^{k-1} - 2$ exist.

As can be seen from the construction of the $(3,k,2)$ anticodes, simple stacking does not increase the maximum distance of the anticode, since no new rows are generated in the process. Thus in general for any $k_1 > k$, simple stacking of an (m,k,δ) anticode q^{k_1-k} times will produce an anticode with parameters (m,k_1,δ) , and this in turn results in block codes for any $k_1 > k$ with the

parameters $n = (q^{k^i} - 1) - m$, $d = q^{k^{i-1}} - \delta$, and $N = q^{k^i}$.

For example, consider stacking the $(10, 2, 8)$ quaternary anticode which was presented in Section 3.1.2. To produce a $(10, 3, 8)$ anticode, $AC(10, 2, 8)$ is stacked on itself $4^{3-2} = 4$ times. To construct an $(10, 4, 8)$ anticode, $AC(10, 2, 8)$ is stacked on itself $4^{4-2} = 16$ times.

Simple stacking is a simple and powerful method for anticode construction. It can be used on its own or in conjunction with other methods to produce optimum anticodes and hence optimum linear codes.

It is convenient to represent simple stacking by the following notation:

$$AC(m, k, \delta) \equiv \boxed{0} \text{ -----> } AC(m, k+1, \delta) \equiv \begin{array}{|c|} \hline 0 \\ \hline 0 \\ \hline \end{array}$$

The above notation represents the binary case. For quaternary anticodes, the notation is defined as:

$$AC(m, k, \delta) \equiv \boxed{0} \text{ -----> } AC(m, k+1, \delta) \equiv \begin{array}{|c|} \hline 0 \\ \hline 0 \\ \hline 0 \\ \hline 0 \\ \hline \end{array}$$

To denote stacking $AC(m, k, \delta)$ 4 times.

3.5.3 INVERSION STACKING

Inversion stacking is another way of using a given anticode to produce anticodes with a higher value of k .

In the binary case, a new anticode is constructed simply by stacking the original anticode upon the complement of itself, i.e., adding 1 to every bit in the code. It is represented by the following notation:

$$AC(m, k, \delta) \equiv \boxed{0} \text{ -----> } AC(m, k+1, m) \equiv \begin{array}{|c|} \hline 0 \\ \hline 1 \\ \hline \end{array}$$

where 1 denotes the anticode with words which are the inversion of those of the basic anticode symbolised by 0 .

For example,

$$\begin{array}{ccc} \begin{array}{c} 0\ 0\ 0 \\ 1\ 0\ 1 \\ 0\ 1\ 1 \\ 1\ 1\ 0 \\ AC(3, 2, 2) \end{array} & \text{----->} & \begin{array}{c} 0\ 0\ 0 \\ 1\ 0\ 1 \\ 0\ 1\ 1 \\ 1\ 1\ 0 \\ \hline 1\ 1\ 1 \\ 0\ 1\ 0 \\ 1\ 0\ 0 \\ 0\ 0\ 1 \\ AC(3, 3, 3) \end{array} \end{array}$$

For generalised anticodes, to produce an anticode with parameter $k+1$, the basic anticode is stacked on itself q times. Then symbols are added to the stacked blocks according to the order that these symbols appear in an adjacency column. For example, to inversion stack a quaternary anticode, the adjacency column will be

$$[0...0 \mid 1...1 \mid 2...2 \mid 3...3]^r$$

so symbol-by-symbol, we add 0 to block 1, add 1 to block 2, add 2 to block 3, add 3 to block 4.

The ternary anticode inversion stacking can be represented by the following notation:

$$AC(m, k, \delta) \equiv \boxed{0} \quad \text{---->} \quad AC(m, k+1, m) \equiv \begin{array}{|c|} \hline 0 \\ \hline I_1 \\ \hline I_2 \\ \hline \end{array} \quad \text{or} \quad \begin{array}{|c|} \hline 0 \\ \hline I_2 \\ \hline I_1 \\ \hline \end{array}$$

where 0 represents the basic anticode, and I_1 denotes 1 added to 0 symbol-by-symbol.

The quaternary anticode inversion stacking can be denoted by the following notation:

$$AC(m, k, \delta) \equiv \boxed{0} \quad \text{---->} \quad AC(m, k+1, m) \equiv \begin{array}{|c|} \hline 0 \\ \hline I_1 \\ \hline I_2 \\ \hline I_3 \\ \hline \end{array} \quad \text{or} \quad \begin{array}{|c|} \hline 0 \\ \hline I_2 \\ \hline I_3 \\ \hline I_1 \\ \hline \end{array} \quad \text{or} \quad \begin{array}{|c|} \hline 0 \\ \hline I_3 \\ \hline I_1 \\ \hline I_2 \\ \hline \end{array}$$

where 0 represents the basic anticode, and I_1 denotes 1 added to 0 symbol-by-symbol.

Inversion stacking alters the value of δ , making it equal to m , because of the all-zero codeword in the original linear anticode. Thus inversion stacking on its own is not an effective way of generating optimum or near optimum anticodes. Used in

combination with simple stacking, however, it becomes a powerful method for generating good anticodes.

The following is a simple example of quaternary anticode inversion stacking:

$$\begin{array}{rcccl}
 & & & \begin{array}{r} 0\ 0 \\ 1\ 2 \\ 2\ 3 \\ 3\ 1 \\ \hline 1\ 1 \\ 0\ 3 \\ 3\ 2 \\ 2\ 0 \\ \hline 2\ 2 \\ 3\ 0 \\ 0\ 1 \\ 1\ 3 \\ \hline 3\ 3 \\ 2\ 1 \\ 1\ 0 \\ 0\ 2 \end{array} & & \\
 \begin{array}{r} 0\ 0 \\ 1\ 2 \\ 2\ 3 \\ 2\ 1 \\ AC(2,1,2) \end{array} & \xrightarrow{\quad\quad\quad} & & & \begin{array}{l} AC(2,1,2) +_4 1 \\ \\ AC(2,1,2) +_4 2 \\ \\ AC(2,1,2) +_4 3 \end{array} \\
 & & AC(2,2,2) & &
 \end{array}$$

3.5.4 COMBINED STACKING

Linear anticodes can be produced by various combinations of simple stacking, inversion stacking and adding an adjacency column. A condition for these combinations is that the results must be linear, i.e. they should produce a linear anticode. This follows the rules for inversion stacking, since simple stacking always produce a linear code.

There are basically two types of combined stacking. One is

single combined stacking, in which the simple stacked anticode and the inversion stacked anticode, are derived from the same basic anticode. The other method is mixed combined stacking, in which each of the component anticodes are derived from different basic anticodes with the same k .

Combined stacking is very simple to apply for constructing binary anticodes. Single combined stacking is denoted by:

$$AC(m, k, \delta) \quad \boxed{0} \quad \text{-----} \rightarrow \quad \begin{array}{|c|c|} \hline 0 & 0 \\ \hline I & 0 \\ \hline \end{array} \quad AC(2m, k+1, 2\delta)$$

The maximum distance in the lower half of this anticode is m (all the words in this part have weight m). Since all linear anticodes without repeated columns have $m/2 < \delta$, the overall maximum weight is 2δ [1]. Because $m/2 < \delta$, it is always possible to add the $[0 \dots 0 \mid 1 \dots 1]^T$ adjacency column to this anticode, and produce an anticode with the same maximum distance but longer length:

$$AC(m, k, \delta) \quad \boxed{0} \quad \text{-----} \rightarrow \quad \begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 1 & I & 0 \\ \hline \end{array} \quad AC(2m+1, k+1, 2\delta)$$

where 0 represents the basic anticode and $[0 \mid 1]^T$ denotes the set of q^k zeros and q^k ones. This is an important and useful method of combined stacking, as it permits construction of many new anticodes from previously known ones.

Mixed combined stacking in $GF(2)$ is restricted by the

following condition:

$$\begin{array}{ccc}
 \boxed{0_1} & \& \boxed{0_2} & \text{-----}> & \begin{array}{|c|c|} \hline 0_2 & 0_1 \\ \hline I_2 & 0_1 \\ \hline \end{array} \\
 AC(m_1, k, \delta_1) & & AC(m_2, k, \delta_2) & & AC(m_1 + m_2, k+1, \delta_1 + \delta_2) \\
 & & & & \text{provided } m_2 - w_{2-\min} \leq \delta_2 \text{ and} \\
 & & & & m_2 \leq \delta_1 + \delta_2 \text{ where } w_{2-\min} = \min(w_2)
 \end{array}$$

The minimum weight, $w_{2-\min}$, of the basic anticode (m_2, k, δ_2) should be such that $m_2 - w_{2-\min} \leq \delta_2$, and $m_2 \leq \delta_1 + \delta_2$. If $m_2 - w_{2-\min} \leq \delta_2$, and $m_2 \leq \delta_1 + \delta_2 - 1$, then an adjacency column may also be added to the constructed anticode without increasing the maximum distance.

The application of combined stacking in generalised anticode construction is more complicated. Consider single combined stacking in GF(4):

$$\begin{array}{ccc}
 AC(m, k, \delta) & \boxed{0} & \text{-----}> & \begin{array}{|c|c|} \hline 0 & 0 \\ \hline I_1 & 0 \\ \hline I_2 & 0 \\ \hline I_3 & 0 \\ \hline \end{array} & AC(2m, k+1, 2\delta) \\
 \text{with } m - \tau \leq \delta & & & &
 \end{array}$$

0 represents the basic anticode, I_i is constructed by adding i to 0 , so every digit i in 0 becomes 0 in I_i , and every digit 0 in 0 becomes i in I_i . Therefore, the maximum distance of the new anticode is related to the minimum number of τ s in each row of 0 , which is represented by the following notation :

$$\tau_i = \min(\text{number of } \tau\text{s in each row of } 0)$$

and $\tau = \min(\tau_i)$.

Thus, the restriction for generalised anticode single combined stacking is:

$$m - \tau \leq \delta.$$

As an example, a ternary optimum anticode $AC(6,2,5)$ can be used to construct an optimum anticode $AC(20,3,15)$ as follows [1]:

```

0 0 0 0 0 0
0 0 1 2 1 2
0 0 2 1 2 1
2 1 1 1 0 0
2 1 2 0 1 2
2 1 0 2 2 1
1 2 2 2 0 0
1 2 0 1 1 2
1 2 1 0 2 1

```

$AC(6,2,5)$

----->

0	0	0	0	0
2	1	I_2	I_1	0
1	2	I_1	I_2	0

$AC(20,3,15)$

where $O = AC(6,2,5)$

$$I_1 = AC(6,2,5) +_3 1$$

$$I_2 = AC(6,2,5) +_3 2$$

and the two adjacency columns are composed of sets of 3^* zeros, 3^* ones and 3^* twos.

We can see that $\tau = 1$ for $AC(6,2,5)$, so the restriction $m - \tau \leq \delta$, or $6 - 1 \leq 5$ here, is satisfied.

The necessary condition for single combined stacking can be modified to apply to mixed combined stacking:

where $\tau_x = \min(\tau_{x_i})$

and $\tau_{x_i} = \min(\text{number of 1s in each row of } O_{x_i})$

It is difficult to obtain the value of τ , especially when k is large. So a convenient way to avoid this problem is to use an M-sequence code for the inversion stacking. An M-sequence code has the same number of non-zero elements in each non-zero word, which is:

$$\tau = \frac{d}{q-1} = \frac{(q-1)q^{k-1}}{q-1} = q^{k-1}$$

Since the length of the M-sequence code is $m = q^k - 1$, we have $m - \tau = (q^k - 1) - q^{k-1} = (q-1)q^{k-1} - 1$. The maximum weight of the M-sequence code is $\delta = (q-1)q^{k-1}$. Therefore, the condition $m - \tau < \delta$ is always satisfied. Because $\delta = m - \tau + 1$, it is always possible to add one adjacency column to an anticode, which contains inversion stacking of a M-sequence code, to produce an anticode with the same maximum distance but longer length.

Another important reason to use the above technique is that M-sequence codes are basic anticodes. It is crucial that the anticode which is used in inversion stacking is basic. If the anticode is not basic, then it will have repeated codewords. An anticode with repeated rows has more than one all-zero codewords (for example the binary (3,3,2) anticode in Section 3.3). When i is added to it, the resulting anticode block $\mathbf{r}_i$ will have several rows with weight m . This will produce an anticode with maximum distance larger than $\Sigma \delta_i$, which is not optimum.

Using an M-sequence code for inversion stacking is very useful in combined stacking for generalised anticodes. Many of the anticodes which are listed in the next chapter have been constructed by applying this method.

For example, in $GF(4)$, the adjacency of $AC(5,2,4)$ simple stacked to $AC(5,3,4)$, and the M-sequence code $(15,2,12)$, which has $\tau = 4$, inversion stacked to $(15,3,15)$, gives optimum $AC(20, 3, 16)$. Adding one adjacency column to $AC(20,3,16)$ results in the optimum anticode $AC(21,3,16)$.

Applying combined stacking, many optimum or near optimum anticodes can be constructed from basic anticodes with small k . This makes combined stacking a very powerful tool for constructing linear anticodes.

In the binary field, combined stacking can be extended to a variety of combinations of simple and inversion stacking, with or without column adjacency. As an example, an optimum binary $AC(3, 2, 2)$ can be used to construct an optimum $AC(10, 4, 6)$ as follows [2]:

This construction includes simple stacking of $AC(3,2,2)$, inversion stacking of $AC(3,2,2)$, simple stacking and inversion stacking of $AC(3,3,3)$ which is the inversion stacking of $AC(3,2,2)$, and adjacency column of 2^n zeros and 2^n ones.

There are some restrictions of using combined stacking for generalized anticodes construction. These will be discussed in the next chapter.

All of the anticode construction methods presented in this chapter can be extended to any field according to the same principles and conditions.

CHAPTER 4

RESULTS

The results of applying the various anticode construction methods described in the previous chapter, and also those obtained from a computer search, are given in this chapter.

4.1 ANTICODE COMPUTER SEARCH

In order to complement the theoretical study of the generalised linear anticodes, several programs were written and developed in the C language and used on SUN workstations. These programs are software realisations of algorithms developed by the author to search for anticodes with predetermined parameters.

ALGORITHM 1(Basic search program)

- Step 1. Input the predetermined parameters (m , k , δ) for the anticode which is to be constructed.
 - Step 2. According to the value of k , build M-sequence code book.
 - Step 3. Input the columns which the user has chosen to construct the anticode.
 - Step 4. Test whether the maximum distance of the constructed anticode meets the predetermined δ and output the result.
- END.

This program is very basic, and the result is completely dependent on which columns the user chooses to construct the anticode. This program works well for small k , eg. $k = 2$, because it is possible to construct optimum anticodes by randomly selecting columns from the M-sequence code book. When k gets large, it is very difficult to construct optimum anticodes by random selection since the number of columns in the M-sequence code book is large. This program was used primarily to test whether an anticode constructed using the methods presented in Chapter 3, is optimum.

When an optimum anticode is constructed, it can be used to construct other anticodes. MacWilliams and Sloane describe six methods for constructing new codes from old ones [4]. Based on these methods and according to the properties of generalised linear anticodes, two algorithms were developed by the author to construct new anticodes: the deletion method and the heuristic method. The delete and heuristic search programs are very powerful, as many optimum anticodes were constructed.

ALGORITHM 2 (Deletion search program)

- Step 1. Input parameters m and k of the known optimum anticode.
- Step 2. Input the predetermined parameter δ of the anticode $(m-1, k, \delta)$ which is to be constructed.
- Step 3. According to the value of k , build M-sequence code book.
- Step 4. Input columns of the known optimum anticode. Exhaustive search through these columns: deleting one column, test if

the new anticode meets the predetermined δ ; if "yes", record the number of the deleted column.

Step 5. Output all suitable columns or state nothing is found.

END.

ALGORITHM 3 (Heuristic search program)

Step 1. Input parameters m and k of the known optimum anticode.

Step 1. Input the predetermined parameter δ for the anticode $(m+1, k, \delta)$ which is to be constructed.

Step 2. According to the value of k , build M-sequence code book.

Step 3. Input the columns of the known optimum anticode. Search through the columns which do not belong to the known optimum anticode: add one column, test if the new anticode meets the predetermined δ ; if "yes", record the number of the added column.

Step 4. Output all suitable columns or state nothing is found.

END.

4.2 LINEAR ANTICODES

In this section, the constructions of ternary and quaternary anticodes with $k = 2$ to 6 are given. The constructions included have been chosen according to the simplest anticode construction. If an anticode is a result of the computer search, then the first anticode found by this search has been listed. Typically the anticode with small k have been used to construct anticode with larger k .

The information columns of the M-sequence code are defined as follows:

Column 1: [q^{k-1} of 0 | q^{k-1} of 1 | q^{k-1} of 2 | ... | q^{k-1} of q]^r

Column 2: [q^{k-2} of 0 | q^{k-2} of 1 | q^{k-2} of 2 | ... | q^{k-2} of q]^r

simple stack q times

Column 3: [q^{k-3} of 0 | q^{k-3} of 1 | q^{k-3} of 2 | ... | q^{k-3} of q]^r

simple stack q^2 times

.....

Column k : [q^{k-k} of 0 | q^{k-k} of 1 | q^{k-k} of 2 | ... | q^{k-k} of q]^r

simple stack q^{k-1} times

For example, the information columns for M-sequence code with $k = 3$ in GF(3) are:

Column 1: [$3^{3-1} = 9$ of 0 | $3^{3-1} = 9$ of 1 | $3^{3-1} = 9$ of 2]^r

[000 000 000 | 111 111 111 | 222 222 222]^r

Column 2: [$3^{3-2} = 3$ of 0 | $3^{3-2} = 3$ of 1 | $4^{3-2} = 3$ of 2]^r

simple stack $3^{3-2} = 3$ times

[000 111 222 | 000 111 222 | 000 111 222]'

Column 3: [$3^{3-3} = 1$ of 0 | $3^{3-3} = 1$ of 1 | $3^{3-3} = 1$ of 2]'

simple stack $3^{3-1} = 9$ times

[012 012 012 | 012 012 012 | 012 012 012]'

In order to show how the information columns are combined to construct the redundancy columns of the M-sequence code, consider the following example. The three information columns in above example, denote Column 1 as C, Column 2 as B and Column 3 as A. The linear combinations of A, B, C are listed below:

Columns 4 to 7: $i*C+j*B$, $i=1$ to 2, $j=1$ to 2, j changes first.

Columns 8 to 11: $i*C+j*A$, $i=1$ to 2, $j=1$ to 2, j changes first.

Columns 12 to 15: $i*B+j*A$, $i=1$ to 2, $j=1$ to 2, j changes first.

Columns 16 to 23: $i*C+j*B+h*A$, $i=1$ to 2, $j=1$ to 2, $h=1$ to 2, h changes first.

Columns 24 to 26: $2*C$, $2*B$, $2*A$, $3*C$, $3*B$, $3*A$.

Please notice that Column 1 actually is the basic adjacency column, Column 58 and 61 are the other two adjacency column. Please see the M-sequence code book below.

From the above example, we can see that the M-sequence codes in the computer search programs are constructed in the following order: first the information columns, then combinations of two information columns, three information columns, ..., k information columns, finally the information columns multiplied by the elements

in $GF(q)$ (besides 0 and 1).

Ternary M-sequence code book for $k=3$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
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	0	1	0	0	0	0	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	0	0	2
0	0	2	0	0	0	0	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2	1	0	0	1
0	1	0	1	2	1	2	0	0	0	0	1	1	2	2	1	1	2	2	1	1	2	2	0	2	0
0	1	1	1	2	1	2	1	2	1	2	2	0	0	1	2	0	0	1	2	0	0	1	0	2	2
0	1	2	1	2	1	2	2	1	2	1	0	2	1	0	0	2	1	0	0	2	1	0	0	2	1
0	2	0	2	1	1	1	0	0	0	0	2	2	1	1	2	2	1	1	2	2	1	1	0	1	0
0	2	1	2	1	2	1	1	2	1	2	0	1	2	0	0	1	2	0	0	1	2	0	0	1	2
0	2	2	2	1	2	1	2	1	2	1	1	0	0	2	1	0	0	2	1	0	0	2	0	1	1
1	0	0	1	1	2	2	1	1	2	2	0	0	0	0	1	1	1	1	2	2	2	2	2	0	0
1	0	1	1	1	2	2	2	0	0	1	1	2	1	2	2	0	2	0	0	1	0	1	2	0	2
1	0	2	1	1	2	2	0	2	1	0	2	1	2	1	0	2	0	2	1	0	1	0	2	0	1
1	1	0	2	0	0	1	1	1	2	2	1	1	2	2	2	2	0	0	0	0	1	1	2	2	0
1	1	1	2	0	0	1	2	0	0	1	2	0	0	1	0	1	1	2	1	2	2	0	2	2	2
1	1	2	2	0	0	1	0	2	1	0	0	2	1	0	1	0	2	1	2	1	0	2	2	2	1
1	2	0	0	2	1	0	1	1	2	2	2	2	1	1	0	0	2	2	1	1	0	0	2	1	0
1	2	1	0	2	1	0	2	0	0	1	0	1	2	0	1	2	0	1	2	0	1	2	2	1	2
1	2	2	0	2	1	0	0	2	1	0	1	0	0	2	2	1	1	0	0	2	2	1	2	1	1
2	0	0	2	2	1	1	2	2	1	1	0	0	0	0	2	2	2	2	1	1	1	1	1	0	0
2	0	1	2	2	1	1	0	1	2	0	1	2	1	2	0	1	0	1	2	0	2	0	1	0	2
2	0	2	2	2	1	1	1	0	0	2	2	1	2	1	1	0	1	0	0	2	0	2	1	0	1
2	1	0	0	1	2	0	2	2	1	1	1	1	2	2	0	0	1	1	2	2	0	0	1	2	0
2	1	1	0	1	2	0	0	1	2	0	2	0	0	1	1	2	2	0	0	1	1	2	1	2	2
2	1	2	0	1	2	0	1	0	0	2	0	2	1	0	2	1	0	2	1	0	2	1	1	2	1
2	2	0	1	0	2	2	2	2	1	1	2	2	1	1	1	1	0	0	0	0	2	2	1	1	0
2	2	1	1	0	0	2	0	1	2	0	0	1	2	0	2	0	1	2	1	2	0	1	1	1	2
2	2	2	1	0	0	2	1	0	0	2	1	0	0	2	0	2	2	1	2	1	1	0	1	1	1

Many optimum anticodes were constructed by the methods discussed in Chapter 3. Then the corresponding columns in the M-sequence code book were found. This information was input to the computer search programs to find other optimum anticodes with smaller or larger m (in a small range).

4.2.1 TERNARY LINEAR ANTICODES

Many optimum anticodes for $4 \leq k \leq 6$ have been constructed by the author (ternary anticodes for $1 \leq k \leq 3$ have been constructed by Farrag [1]). The resulting codes yield numerous known optimum linear block codes. Typical examples of anticodes construction for $4 \leq k \leq 6$ are presented below.

$$k = 4$$

$AC(27,4,19) = (\text{Heuristic search})$ Composed columns: 2-5, 17-28, 53-60, 78-80.

$$\Rightarrow RC(53,4,35)$$

$AC(28,4,20) = (\text{Heuristic search})$ $AC(27,4,19) + \text{Column 6.}$

$$\Rightarrow RC(52,4,34)$$

$AC(29,4,21) = (\text{Heuristic search})$ $AC(28,4,20) + \text{Column 7.}$

$$\Rightarrow RC(51,4,33)$$

$AC(36,4,26) = AC(10,3,8)$ simple stacked adjoined to the inversion stacking of $AC(26,3,18)$.

$$O_1 = AC(47,3,36)$$

$$O_2 = AC(63,3,48)$$

O_2	O_1
I_{21}	O_1
I_{22}	O_1

Composed columns in its parent M-sequence code:

2-6, 9-10, 13-14, 25-32, 37-40, 45-48, 61-68, 78-80.

$$\Rightarrow RC(44,4,28)$$

$AC(35,4,25) = (\text{Deletion search})$ $AC(36,4,26) - \text{Column 6.}$

Composed columns: 2-5, 9-10, 13-14, 25-32, 37-40,

45-48, 61-68, 78-80.

=> RC(45,4,29)

AC(34,4,24) = (Deletion search) AC(35,4,25) - Column 5.

Composed columns: 2-4, 9-10, 13-14, 25-32, 37-40,
45-48, 61-68, 78-80.

=> RC(46,4,30)

AC(33,4,24) = (Deletion search) AC(34,4,24) - Column 4.

Composed columns: 2-3, 9-10, 13-14, 25-32, 37-40,
45-48, 61-68, 78-80.

=> RC(47,4,30)

AC(46,4,32) = $O_1 = AC(47,3,36)$

$O_2 = AC(63,3,48)$

adjacency column:

[0 | 1 | 2]

0	O_2	O_1
1	I_{21}	O_1
2	I_{22}	O_1

Composed columns in its parent M-sequence code:

1-6, 9-10, 13-14, 17-22, 25-32, 37-40, 45-48, 61-68,
78-80.

=> RC(34,4,22)

AC(47,4,33) = (Heuristic search) AC(46,4,32) + Column 7.

=> RC(33,4,21)

AC(45,4,31) = (Deletion search) AC(46,4,32) - Column 80.

=> RC(35,4,23)

AC(44,4,30) = (Deletion search) AC(45,4,31) - Column 18.

=> RC(36,4,24)

AC(43,4,30) = (Deletion search) AC(44,4,39) - Column 79.

=> RC(37,4,24)

$$k = 5$$

$AC(92,5,63) = AC(12,3,9)$ simple stacked adjoined to the inversion stacking of $AC(80,4,54)$.

$$\Rightarrow RC(150,5,99)$$

$AC(93,5,63) = AC(92,5,63) +$ Adjacency column of $3^* 0s$, $3^* 1s$ and $3^* 2s$.

$$\Rightarrow RC(149,5,99)$$

$AC(114,5,78) = AC(34,4,24)$ simple stacked adjoined to the inversion stacking of $AC(80,4,54)$.

$$\Rightarrow RC(128,5,84)$$

$AC(116,5,80) = AC(36,4,26)$ simple stacked adjoined to the inversion stacking of $AC(80,4,54)$.

$$\Rightarrow RC(150,5,99)$$

$AC(160,5,108) = AC(80,4,54)$ simple stacked adjoined to its inversion stacking.

$$\Rightarrow RC(82,5,54)$$

$AC(161,5,108) = AC(160,5,108) +$ Adjacency column of $3^* 0s$, $3^* 1s$ and $3^* 2s$.

$$\Rightarrow RC(81,5,54)$$

$$k = 6$$

$AC(334,6,225) = AC(92,5,63)$ simple stacked adjoined to the inversion stacking of $AC(242,5,162)$.

$$\Rightarrow RC(394,6,261)$$

$AC(335,6,225) = AC(334,6,225) +$ Adjacency column of $3^* 0s$, $3^* 1s$ and $3^* 2s$.

$$\Rightarrow RC(393, 6, 261)$$

$AC(403, 6, 270) = AC(161, 5, 108)$ simple stacked adjoined to the inversion stacking of $AC(242, 5, 162)$.

$$\Rightarrow RC(325, 6, 216)$$

$AC(404, 6, 270) = AC(161, 6, 108) + \text{Adjacency column of } 3^* 0s, 3^* 1s \text{ and } 3^* 2s.$

$$\Rightarrow RC(324, 6, 216)$$

$$AC(484, 6, 324) =$$

$$O = AC(242, 5, 162)$$

0	0	0
I_2	I_1	0
I_1	I_2	0

$$\Rightarrow RC(244, 6, 162)$$

$AC(485, 6, 324) = AC(484, 6, 324) + \text{Adjacency column of } 3^* 0s, 3^* 1s \text{ and } 3^* 2s.$

$$\Rightarrow RC(243, 6, 162)$$

4.2.1 QUATERNARY LINEAR ANTICODES

Many optimum anticodes for $1 \leq k \leq 5$ have been constructed by the author. The resulting codes yield numerous know optimum linear block codes. Typical examples of anticodes construction for $3 \leq k \leq 5$ are presented below.

$$k = 3$$

$AC(20, 3, 16) = AC(5, 2, 4)$ simple stacked adjoined to the inversion stacking of $AC(15, 2, 12)$.

$$\Rightarrow RC(43, 3, 32)$$

$AC(21,3,16) = AC(20,3,16) + \text{adjacency column of } 4^* 0s, 4^* 1s, * 2s$
and $4^* 3s$.

$\Rightarrow RC(42,3,32)$

$AC(40,3,32) =$

$0 = AC(10,2,8)$

0	0	0	0
I_3	I_2	I_1	0
I_1	I_3	I_2	0
I_2	I_1	I_3	0

$\Rightarrow RC(23,3,16)$

$AC(41,3,32) = AC(40,3,32) + \text{Adjacency column of } 4^* 0s, 4^* 1s, 4^* 2s$
and $4^* 3s$.

$\Rightarrow RC(22,3,16)$

$AC(42,3,32) = AC(41,3,32) + \text{Adjacency column of } 4^* 0s, 4^* 2s, 4^* 3s$
and $4^* 1s$.

$\Rightarrow RC(21,3,16)$

$AC(48,3,37) = AC(15,2,12)$ has $\tau = 4$, and $m-\tau = 11$, then by inversion stacking of $AC(15,2,12)$, code (15,3,11) is produced. By deleting this code from its parent M-sequence code, anticode (48,3,37) is obtained.

Composed columns in its parent M-sequence code:

1-3, 7-12, 16-30, 40-63.

$\Rightarrow RC(15,3,11)$

$AC(49,3,38) = (\text{Heuristic search}) AC(48,3,37) + \text{Column } 39$.

$\Rightarrow RC(14,3,10)$

$AC(50,3,39) = (\text{Heuristic search}) AC(49,3,38) + \text{Column } 38$.

$\Rightarrow RC(13,3,9)$

AC(59,3,46) = (Basic search) Composed columns: 1-56, 60-62.

=> RC(4,3,2)

AC(58,3,45) = (Deletion search) Composed columns: 1-35, 37-56,
60-62.

=> RC(5,3,3)

AC(57,3,44) = (Deletion search) Composed columns: 1-35, 37, 39-56,
60-62.

=> RC(6,3,4)

$k = 4$

AC(89,4,68) = AC(26,3,20) simple stacked adjoined to the inversion
stacking of AC(63,3,48).

Composed columns in its parent M-sequence code :

2-7, 14-16, 23-25, 32-34, 41-43, 50-55, 59-67, 86-94,
113-121, 140-148, 167-193, 254-255.

=> RC(166,4,124)

AC(90,4,68) = AC(89,4,68) + adjacency column of 4^s 0s, 4^s 1s, 4^s 2s
and 4^s 3s.

=> RC(165,3,124)

AC(105,4,80) = AC(42,3,32) simple stacked adjoined to the
inversion stacking of AC(63,3,48).

Composed columns in its parent M-sequence code :

2-7, 14-16, 23-25, 32, 34-35, 37-38, 40-41, 43-44,
46-47, 49-55, 59-67, 86-94, 113-121, 140-145, 149-
154, 158-163, 167-193, 249, 254-255.

=> RC(150,4,112)

AC(175,4,132) =

$O_1 = AC(47,3,36)$

$O_2 = AC(63,3,48)$

adjacency columns:

[0 | 1 | 2 | 3]

[0 | 2 | 3 | 1]

0	0	O_2	O_2	O_1
1	2	I_{21}	I_{22}	O_1
2	3	I_{22}	I_{23}	O_1
3	1	I_{23}	I_{21}	O_1

=> RC(80,4,60)

AC(192,4,145) = AC(63,3,48) has $\tau = 12$, and $m-\tau = 47$, then by inversion stacking of AC(63,3,48), code (63,4,47) is produced. By deleting this code from its parent M-sequence code, anticode (192,4,145) is obtained. Composed columns in its parent M-sequence code: 1-4, 8-13, 17-22, 26-58, 68-85, 95-112, 122-166, 194-255.

=> RC(63,4,47)

AC(193,4,146) = (Heuristic search) AC(192,4,145) + Column 5.

=> RC(62,4,46)

AC(194,4,147) = (Heuristic search) AC(193,4,146) + Column 6.

=> RC(61,4,45)

AC(251,4,191) = (Basic search) Composed columns: 2-252.

=> RC(4,4,1)

AC(250,4,190) = (Deletion search) Composed columns: 2-247, 249-252.

=> RC(5,4,2)

AC(249,4,190) = (Deletion search) Composed columns: 2-166, 168-247, 249-252.

=> RC(6,4,2)

$$k = 5$$

$$AC(600, 5, 452) = O_1 = AC(89, 4, 68)$$

$$O_2 = AC(255, 4, 192)$$

adjacency columns:

$$[0 \mid 1 \mid 2 \mid 3]$$

$$\Rightarrow RC(423, 5, 316)$$

$$AC(601, 5, 542) = AC(600, 5, 542) + \text{adjacency column of } 4^* 0s, 4^* 2s, 4^* 3s \text{ and } 4^* 1s.$$

$$\Rightarrow RC(422, 5, 316)$$

$$AC(615, 5, 464) =$$

$$O_1 = AC(105, 4, 80)$$

$$O_2 = AC(255, 4, 192)$$

$$\Rightarrow RC(408, 5, 304)$$

$$AC(616, 5, 464) = AC(615, 5, 464) + \text{adjacency column of } 4^* 0s, 4^* 1s, 4^* 2s \text{ and } 4^* 3s.$$

$$\Rightarrow RC(407, 5, 304)$$

$$AC(617, 5, 464) = AC(616, 5, 464) + \text{adjacency column of } 4^* 0s, 4^* 2s, 4^* 3s \text{ and } 4^* 1s.$$

$$\Rightarrow RC(406, 5, 304)$$

$$AC(765, 5, 576) =$$

$$O = AC(255, 4, 192)$$

$$\Rightarrow RC(258, 5, 192)$$

$$AC(766, 5, 576) = AC(765, 5, 192) + \text{adjacency column of } 4^* 0s, 4^* 1s,$$

0	O_2	O_2	O_1
1	I_{21}	I_{22}	O_1
2	I_{22}	I_{23}	O_1
3	I_{23}	I_{21}	O_1

O_2	O_2	O_1
I_{21}	I_{22}	O_1
I_{22}	I_{23}	O_1
I_{23}	I_{21}	O_1

0	0	0
I_2	I_1	0
I_3	I_2	0
I_1	I_3	0

$4^* 2s$ and $4^* 3s$.

$\Rightarrow RC(257, 5, 192)$

$AC(767, 5, 576) = AC(766, 5, 576) + \text{adjacency column of } 4^* 0s, 4^* 2s,$
 $4^* 3s \text{ and } 4^* 1s.$

$\Rightarrow RC(256, 5, 192)$

4.3 LIMITATIONS OF GENERALISED ANTICODE CONSTRUCTION

To date, there are numerous unknown optimum codes of small length. For example, there are unknown quaternary codes with parameters $(n \leq 298, k=5)$ and ternary codes with parameters $(n \leq 307, k=6)$. Using the anticode method to obtain codes with these length requires the construction of anticodes with very large lengths. Specifically, quaternary anticodes with parameters $(m \geq 725, k=5)$ and ternary anticodes with parameters $(m \geq 421, k=6)$. However, there are some limitations to generalised anticode construction. These limitations make it very difficult to obtain anticodes with large length.

As discussed in Section 3.5.5, the anticode used in inversion stacking must be a basic anticode, otherwise it will not satisfy the conditions for combined stacking. For example, quaternary anticode $AC(105, 4, 80)$ contains simple stacking of $AC(42, 3, 32)$ and inversion stacking of M-sequence code $(63, 3, 48)$. $AC(105, 4, 80)$ is not a basic anticode, if it is used for inversion stacking, then

there will be 6 rows with weight=105. It is then impossible to combine this with any other anticodes to construct long optimum anticodes. If AC(105,4,80) and M-sequence code (1023,5,768) are combined together, the resulting generator matrix is:

$$\begin{aligned} O_1 &= AC(42,3,32) \\ O_2 &= AC(63,3,48) \\ O_1 + O_2 &= AC(105,4,80) \\ O_3 &= AC(255,4,192) \end{aligned}$$

			O_2	O_1
O_3	O_3	O_3	I_{21}	O_1
			I_{22}	O_1
			I_{23}	O_1
I_{32}	I_{33}	O_3	I_{21}	O_1
			O_2	I_{11}
			I_{23}	I_{11}
			I_{22}	I_{11}
I_{33}	I_{31}	O_3	I_{22}	O_1
			I_{23}	I_{12}
			O_2	I_{12}
			I_{21}	I_{12}
I_{31}	I_{32}	O_3	I_{23}	O_1
			I_{22}	I_{13}
			I_{21}	I_{13}
			O_2	I_{13}

There is a maximum length anticodes can be constructed using these method. For example, to construct a long quaternary anticode with $k=5$ (denoted by O), a non-basic anticode with $k=4$ (denoted by O_1) simple stacked conjoined with tree inversion stacking of M-sequence code (255,4,192) (denoted by O_2):

O_2	O_2	O_2	O_1
I_{23}	I_{22}	I_{21}	O_1
I_{21}	I_{23}	I_{22}	O_1
I_{22}	I_{21}	I_{23}	O_1

The first row of blocks $[I_{21} \ I_{21} \ I_{21} \ O_1]$ has weight = $3 \times 255 = 765$. So the new anticode O will have maximum weight = 765. We know that the M-sequence code (1123,5,768) has maximum weight = 768, so obviously O is far from optimum. Therefore, we can only use two inversion stackings of the M-sequence code. This means that the length of the anticode we can obtain is small.

O_2	O_2	O_1
I_{22}	I_{21}	O_1
I_{23}	I_{22}	O_1
I_{21}	I_{23}	O_1

Each O_2 has block length = 255, and the maximum length of O_1 is also 255. So the maximum length of O is $m = 3 \times 255 = 765$.

In $GF(3)$, for the same reason, only one inversion stacking of an M-sequence code can be used in combined stacking. The M-sequence code for $k=5$ has length = 242. So, for $k=6$, the maximum anticode length which can be obtained is $m = 2 \times 242 = 484$.

In brief, for a given k in any $GF(q)$, the maximum length of anticode can be constructed is :

$$m_{\max} = (q-1)q^{k-1} = (q-1)q^{k-1}$$

Because of these limitations, it is very difficult to obtain new optimum generalised codes from the anticode method.

CHAPTER 5

DISCUSSION AND CONCLUSIONS

The work reported in this thesis is the result of a study of generalised anticodes and their application to the problem of finding generalised optimum error-correcting codes. If a linear anticode, which is an array with a given number of columns, m , minimum maximum-distance, δ , and q^k rows, is deleted or punctured from its parent M-sequence code, a linear block code is obtained. Thus the problem of finding error-correcting codes reduces to that of finding suitable anticodes. Once a basic anticode is found, it can be used in various constructions which result in a number of other anticodes. For example,

the simple stacking anticode
the inversion stacking anticode
the single combined stacking anticode
and the mixed combined stacking anticode.

The following is a summary of the results achieved:

1. Many known ternary linear block codes with $4 \leq k \leq 6$, which have already been found by various methods, were obtained and confirmed through the corresponding anticodes.
2. Many known quaternary linear block codes with $1 \leq k \leq 5$, which have already been found by various methods, were obtained and confirmed through the corresponding anticodes.

Note that the application of the anticode method was able to yield results that were produced from various methods and different sources. Therefore, constructing anticodes is an efficient and simple method for obtaining optimum and near-optimum error-correcting codes.

However, because of the limitations of generalised anticode constructions, it is very difficult to construct very large anticodes. And all the unknown optimum codes have small lengths, so no recode breaking code has been found during this study.

This research focused on quaternary and ternary linear anticodes and codes without repeated columns. The anticode construction methods can be extended to any field. The construction method for anticodes with repeated columns and their role in obtaining optimum codes is worth further study. The computer search algorithms can easily be extended to other values of k and q .

REFERENCES

- [1] A.M. Farrag, "Anticodes and optimum error-correcting codes," Ph.D. Thesis, University of Kent at Canterbury, 1976.
- [2] P.G. Farrell, "An introduction to anticodes," CISM, p.179-229, 1978.
- [3] S. Lin and D.J. Costello, Jr., *Error Control Coding*, Prentice-Hall, Englewood Cliffs, New Jersey, 1991.
- [4] F.J. Macwilliams and N.J.A Sloane, *The Theory of Error-Correcting Codes*, North-Holland, New York, 1977.
- [5] V. Pless, *Introduction to the Theory of Error-Correcting Codes*, Wiley, Toronto, 1989.
- [6] D.G. Hoffman, D.A. Leonard, C.C. Lindner, K.T. Phelps, C.A. Rodger and J.R. Wall, *Coding Theory*, Marcel Dekker, New York, 1991.
- [7] R.P. Grimaldi, *Discrete and Combinatorial Mathematics*, 2nd Edition, Addison-Wesley, Don Mills, Ontario, 1989.
- [8] C.E. Shannon, "A Mathematical Theory of Communication," Bell Syst. Tech. J., Vol.27, p.379-423(Part I), 623-656(Part II), July 1948.
- [9] J.H. Griesmer, "A Bound for Error-Correcting Codes," IBM Journal of research and Development, p.532-542, November 1960.
- [10] G. Solomon and J.J. Stiffler, "Punctured Systematic Cyclic Coder," IEEE International Convention Record, Vol.12, Pt.I, p.128-129, 1964.

- [11] G. Solomon and J.J. Stiffler, "Algebraically Punctured Cyclic Codes," Information and Control, Vol.8, p.170-179, 1965.
- [12] P. G. Farrell, "Linear Binary Anticodes," Electronics Letters, Vol.6, No.13, p.419-421, June 1970.
- [13] P.G. Farrell and A. Farrag, "Further Properties of Linear Binary Anticodes," Electronics Letters, Vol.10, No.16, p.340-341, August 1974.
- [14] V.C Da Rocha and M.M. Campello De Souza, "Product and Interleaving of Anticodes," Electronics letters, Vol.27, No.8, p.634-636, April, 1991.
- [15] A.A. Hashim and V.S. Pozdniakov, "Stacking Techniques for Linear Codes," Electronics Letters, Vol.12, No.18, p.481-483, September, 1976.
- [16] A.A Hashim and V.S. Pozdniakov, "Computer Search for Linear Binary Codes," Electronics Letters, Vol.12, No.14, p.350-351, July, 1976.
- [17] A.A. Hashim, A.G. Constantinides, "Some New Results on Binary Linear Block Codes," Electronics letters, Vol.10, No.3, p.31-33, February 1974.
- [18] P.G. Farrel and Z. Al-Bandar, "Multilevel Single-Error-Correcting Codes," Electronics Letters, Vol.10, No.16, p.347-348, August, 1974.
- [19] D.A. Bell, "Multi-Level Codes," The Radio and Electronic Engineer, Vol.42, No.12, p.555-561, December 1972.
- [20] R.W. Hamming, "Error-Correcting and Error-Detecting Codes," Bell Systems Technical Journal, Vol. 26, p.147-160, 1950.

- [21] N. Hamada and Y. Watamori, "The Nonexistence of Some Ternary Linear Codes of Dimension 6 and The Bounds for $n_3(6,d)$, $1 \leq d \leq 243$," September 1995.
- [22] N. Hamada, "The Nonexistence of Some Quaternary Linear Codes Meeting The Griesmer Bound and The Bounds for $n_4(5,d)$, $1 \leq d \leq 256$," September 1995.

END

0 6-0 6-9/6

FIN