ISCTE University Institute of Lisbon
School of Technology and Architecture
Department of Information Science and Technology

Design of TCH-type Sequences for Communications

A Thesis specially elaborated for obtaining the
Degree of Doctor in Information Science and Technology

By
Engº Alexandre M. C. Passos de Almeida, M.Sc.

August - 2012

To my extended family (Gaby, Lourenço, Matilde, Inês C., Nuno, Inês A.). For their love, encouragement and patience...

Key words and phrases. Group theory, Number theory, Finite fields, Coding theory, TCH codes, SN sequences, PN sequences, DFT, UWB

I would like to thank the Prof. Pedro Silva and Prof. Maria Torres for their guidance on the theory of abstract mathematics. I appreciated our fruitful discussions about the marriage between communications engineering and Group and Number theory. I thank also Prof. Pedro Sebastião, Prof. Rui Dinis and Prof. Francisco Cercas for their support and suggestions in the final revision of this thesis.

JÚRI

Presidente: Doutor Francisco António Bucho Cercas, Prof. Catedrático do ISCTE-IUL

Vogais:
Doutor Rui Miguel Henriques Dias Morgado Dinis, Prof. Associado com Agregação da FCT-UNL
Doutor Paulo Miguel de Araújo Borges Montezuma de Carvalho, Prof. Auxiliar da FCT-UNL
Doutor Marco Alexandre Cravo Gomes, Prof. Auxiliar da Universidade de Coimbra
Doutor Nuno Manuel Branco Souto, Prof. Auxiliar do ISCTE-IUL
Doutor Pedro Joaquim Amaro Sebastião, Prof. Auxiliar do ISCTE-IUL

ABSTRACT

This thesis deals with the design of a class of cyclic codes inspired by TCH codewords. Since TCH codes are linked to finite fields the fundamental concepts and facts about abstract algebra, namely group theory and number theory, constitute the first part of the thesis.

By exploring group geometric properties and identifying an equivalence between some operations on codes and the symmetries of the dihedral group we were able to simplify the generation of codewords thus saving on the necessary number of computations. Moreover, we also presented an algebraic method to obtain binary generalized TCH codewords of length $N = 2^k, k = 1, 2, \ldots, 16$. By exploring Zech logarithm's properties as well as a group theoretic isomorphism we developed a method that is both faster and less complex than what was proposed before. In addition, it is valid for all relevant cases relating the codeword length N and not only those resulting from $N = p_i - 1$ for Fermat primes p_i. The method also derives the maximum set of all the codewords of a certain code bringing clear advantages in terms of code size and minimum distance.

In a further investigation we proposed a new generating procedure focusing mostly on group permutations as an efficient way to generate all codewords of a particular cyclic code. For binary sequences associated to sub-Pythagorean primes this method only requires the repeated application of 3 permutations (two for the time domain and one extra for the frequency domain) and a DFT operation, thus saving memory space and processing time. For general M-ary sequences the procedure may require, at most, M additional permutations.

The performance under a Rayleigh fading channel and the application of these sequences to a variety of communications systems, namely Ultra-Wideband (UWB) and Direct Sequence - Code Division Multi Access (DS-CDMA), were also presented.

As a consequence of our sequence design we can summarize some of the advantages obtained: the sequence period or codeword length is not limited to a power of two (and not restricted to a Fermat prime minus 1); using the same generating procedure we can produce (both in time and frequency domains) a larger number of codewords resulting in better data rates and/or better error correction; the mathematical knowledge of the code structure permits not having a loose collection of codewords, but a codeword list with a cohesive structure opening up a lot of improvements in terms of coding/decoding steps; the generation procedure is not limited to binary but allows M-ary sequences as well.

SUMÁRIO

Esta tese aborda o projeto de uma classe de códigos cíclicos inspirados nas palavras do código TCH. Os conceitos fundamentais e fatos sobre álgebra abstrata, ou seja, teoria de grupos e teoria dos números, constituem a primeira parte da tese.

Ao explorar as propriedades geométricas dos grupos e a identificação de uma equivalência entre algumas operações sobre as palavras de código e as simetrias do grupo diedral simplificou-se a geração de palavras de código, economizando no número de cálculos necessários. Além disso, desenvolveu-se um método algébrico para obter palavras binárias generalizadas (do tipo TCH) de comprimento $N = 2^k$, $k = 1, 2, \ldots, 16$. Ao explorar as propriedades dos logaritmos de Zech, bem como um determinado isomorfismo desenvolveu-se um método que é mais rápido e menos complexo do que o que foi proposto anteriormente. Além disso, o método é válido para todos os casos relevantes relativos à palavra de código de comprimento N e não só as resultantes de $N = p_i - 1$ para primos de Fermat p_i. O método também deriva o conjunto máximo de todas as palavras de um determinado código trazendo vantagens claras em termos de tamanho do código e distância mínima.

Numa investigação mais aprofundada propôs-se um novo procedimento de geração, focalizado principalmente em grupos de permutações, permitindo uma forma eficiente de gerar todas as sequências de um código cíclico particular. Para sequências binárias associados a primos sub-pitagóricos este método requer apenas a aplicação repetida de 3 permutações (duas para o domínio do tempo e uma extra para o domínio da frequência) e uma operação de DFT, poupando assim espaço de memória e tempo de processamento. Para sequências M-árias mais gerais o procedimento pode exigir, no máximo, M permutações adicionais.

Nesta tese apresentaram-se também exemplos de aplicação destas sequências no contexto de sistemas de comunicação, como por exemplo, UWB e DS-CDMA.

Pode-se resumir algumas das vantagens resultantes do trabalho apresentado:

• o período da sequência (ou comprimento da palavra de código) não é limitada a uma potência de dois (e o seu valor não fica restrito a um primo de Fermat menos uma unidade),

• utilizando o mesmo procedimento de geração pode-se produzir um maior número de palavras de código, resultando em melhores rácios de transmissão de dados e/ou melhorar a capacidade de correcção de erros,

• o conhecimento matemático da estrutura do código permite passar de um conjunto disperso de palavras de código para uma lista de palavras com uma estrutura coesa abrindo uma série de melhorias em termos de codificação/decodificação,

• o procedimento de geração não é limitado às palavras binárias, mas permite sequências M-árias também.

NOMENCLATURE

$(a_1, a_2, \ldots, a_k)$	cycle of length k
$[E : F]$	dimension of a field extension of E over F
$[G : H]$	index of a subgroup H in a group G
$\operatorname{cis}\theta$	$\cos\theta + j\sin\theta$
$\deg p(x)$	degree of $p(x)$
δ_{ij}	Kronecker delta
$\dim V$	dimension of a vector space V
$\emptyset$	the empty set
$\gcd(m,n)$	greatest common divisor of m and n. Alternatively represented also as (m,n)
$\langle a \rangle$	cyclic subgroup generated by a
$A \cap B$	intersection of sets A and B
$A \cup B$	union of sets A and B
$a \in A$	a is in the set A
$a \preceq b$	a preceeds b
$A \setminus B$	difference between sets A and B
$A \subset B$	A is a subset of B
$A \times B$	Cartesian product of sets A and B
$a \vee b$	join of a and b
$a \wedge b$	meet of a and b
A'	complement of the set A
A^n	$A \times \cdots \times A$ (n times)
A_n	alternating group on n letters
a_{ij}	the (i,j) element of the matrix A
$d(\mathbf{x}, \mathbf{y})$	Hamming distance between $\mathbf{x}$ and $\mathbf{y}$

D_n	dihedral group of order 2n
$d_{\min}$	minimum (Hamming) distance of a code
F^*	multiplicative group of a field F
f^{-1}	inverse of the function f
$G \cong H$	G is isomorphic to H
$G(E/F)$	Galois group of E over F
G/N	factor group of G mod N
i_g	$i_g(x) = gxg^{-1}$
id	identity mapping
$m \mid n$	m divides n
$R[x]$	ring of polynomials over R
S_n	symmetric group on n letters
$U \oplus V$	direct sum of vector spaces U and V
$U(n)$	group of units in $\mathbb{Z}_n$
$w(\mathbf{x})$	weight of $\mathbf{x}$
$Z(G)$	center of a group G
$\mathbb{C}$	the set of complex numbers
$\mathbb{N}$	the set of natural numbers
$\mathbb{Q}$	the set of rational numbers
$\mathbb{R}$	the set of real numbers
$\mathbb{Z}$	the set of integers
$\mathbb{Z}_n$	the integers modulo n
$\mathrm{Aut}(G)$	automorphism group of G
$\mathrm{GF}(p^n)$	Galois field of order p^n
α	a primitive element
$\mathbb{F}_p$	The finite field of order p
$\mathrm{lcm}\,(m,n)$	least common multiple of m and n
$a \equiv b \bmod n$	a is congruent to b modulo n
E_b	Bit Energy
N_0	Spectral Noise Density
R_c	Code Rate
Z(x)	The Zech logarithm of x.

Acronyms

AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BPSK	Binary Phase Shift Keying

CDMA	Code Division Multiple Access
CRC	Cyclic Redundancy Check
DFT	Discrete Fourier Transform
DS-SS	Direct Sequence Spread Spectrum
ESD	Energy Spectral Density
FDMA	Frequency Division Multiple Access
FEC	Forward Error Correction
FFT	Fast Fourier Transform
FH	Frequency Hopping
FH-CDMA	Frequency Hopping Code Division Multiple Access
FH-SS	Frequency Hopping Spread Spectrum
HD	Hard Decision
IDFT	Inverse Discrete Fourier Transform
IFF	(or iff) If and only if
IFFT	Inverse Fast Fourier Transform
LoS	Line of Sight
LSE	Least Square Error
MB-OFDM	Multi Band Orthogonal Frequency Division Multiplexing
MC-CDMA	Multi Carrier Code Division Multiple Access
MUI	Multi User Interference
NLoS	Non Line of Sight
OFDM	Orthogonal Frequency Division Multiplexing
ORD	(ord) Order of an element
PAM	Pulse Amplitude Modulation
PDF	Probability Density Function
PN	Pseudo Noise
PPM	Pulse Position Modulation
QAM	Quadrature Amplitude Modulation
PSD	Power Spectral Density
R_c	The code rate, k/n
RS	Reed-Solomon
QPSK	Quadrature Phase Shift Keying
SD	Soft Decision
SN	Sidel'nikov
SNR	Signal to Noise Ratio
TCH	Tomlinson, Cercas, Hughes

TH	Time Hopping
UWB	Ultra Wide Band

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1. Context

The basic model for communication has remained unchanged for several decades, with the central role being played by the newspaper, the telephone and television. As a society, we are shifting our usage of technology from the management and distribution of information towards the creation of symbolic social linkages. Communication is becoming an end in itself, rather than a means for handling information.

The planetary hypermedia available today offers a world each time more flexible and accessible to an individual taking advantage of the privilege of weight absence, ubiquity and interactivity. Moreover the data transfer still uses text, sound and image, but now in a complete imbricate manner. The new media is no longer the mass media. We have moved from the broadcast to the point-cast and this brings the self-centered, portable or mobile context to most communication devices. What was previously described as the information technology is now better described as the information and communication technology.

The widespread introduction of the Internet in the 90's has spawned many innovations and services that stem from its interactive character. The process of adding mobility to interactivity transformed the role of the Internet and other forms of communication and paved the way for yet another set of innovations and services. The convergence of computing and communication is a process that has turned phones into smart mobile terminals with powerful multimedia capabilities.

The Global System for Mobile communication (GSM) and its extended version, the Digital Communication System (DCS), initially developed for Europe has been adopted in more than 80 countries worldwide. The requirements of multimedia applications drove the research for a 3rd generation system, which started around 1990, the significant outcome being the development

of the Universal Mobile Telecommunications System (UMTS). Currently, the evolution in terms of increasing the capacity and speed using new modulation techniques was made in effect by the 3GPP Long Term Evolution (LTE) that were deployed in the end of 2011. The future is already under development and being marketed as 4G LTE-A (A for Advanced), a standard for wireless communication of high-speed data for mobile smartphones and data terminals.

New generations of wireless mobile radio systems [**2**] aim to provide flexible data rates (including high, medium, and low data rates) and a wide variety of applications (like video, data, ranging, localization etc.) to the mobile users while serving as many users as possible. The limited available resources like spectrum and power, however, constraint this goal. In fact, the demand for higher data rate is leading to utilization of wider transmission bandwidth, BW. We went from 200 kHz for GSM, 1.25 MHz for IS-95, and 5 MHz for UMTS to 20 MHz for LTE and projected 100 MHz to LTE-A. As the transmission bandwidth gets wider more challenges we need to face due the characteristics of the wireless channel. Among these challenges we find a) diverse channel multipath (resulting from wave reflections and refraction), with implications in Inter-Symbol Interference (ISI) and fading in the time domain as well as frequency selectivity in the frequency domain, b) time variability of the channel gain and Doppler shift in the frequency domain as a direct result of user mobility, c) the need for resource management to compensate for limitations in spectrum access and power availability. As more and more devices go wireless, future technologies will face spectral crowding and coexistence of wireless devices will be a major issue. Therefore, considering the limited bandwidth availability, accommodating the demand for higher capacity and data rates is a challenging task, requiring innovative technologies that can coexist with devices operating at various frequency bands.

To address these issues Orthogonal Frequency Division Multiplexing (OFDM) has been proposed as an enabling technology. In OFDM a frequency band is divided into overlapping subcarriers that are orthogonal (when one is at its peak all the others cross zero). Since the BW of each subcarrier is small (much less than the coherence bandwidth of the channel) each subcarrier sees flat fading. This is one advantage of this system. OFDM is however not without design issues. Due to the usage of multiple subcarriers it can suffer from high Peak-to-Average Power Ratio (PAPR). Subcarrier orthogonality can be broken by carrier frequency offsets. To overcome deep spectral valleys a strong channel coding is advised. Being similar to OFDM, Single-Carrier with Frequency Domain Equalization (SC-FDE) gathered some interest and eventually got extended to SC-FDMA to accommodate multi-user access. It uses SC modulation (typically QPSK or 16QAM), Digital Fourier Transform or DFT-spread orthogonal frequency multiplexing and frequency domain equalization. It is currently adopted for the uplink in 3GPP LTE. To estimate the channel a reference signal composed of a symbol by symbol product of an orthogonal sequence and a pseudo-random sequence is used.

1.2. Motivation

All these aspects involving frequency based equalization, Digital Fourier Transforms (DFT), pseudo-random sequences provide us with the motivation to pursue an investigation into methods of either generalizing or optimizing previously known sequences. In our case we had been using TCH sequences so those binary cyclic sequences were the starting point of the work developed in this thesis.

1.2.1. Sequences and coding. In a communications system we are interested in a reliable and at the same time cost effective way of transmitting data from a source to a receiver. The branch of science dealing with these issues is referred to coding theory and it involves the study of codes and their properties. According to the specific discipline (namely mathematics, information theory, electrical engineering etc.) where codes are studied we find their use in applications like network coding, cryptography, data compression and error correction. These last two applications are perhaps the most known and are also referred as *source coding*, for data compression, and *channel coding*, for error detection/correction.

Source encoding deals primarily with removing the redundancy of the data to transmit it more efficiently. This happens, for example, in Zip data compression to make computer files smaller or in facsimile transmission where the use of a run length code removes all superfluous data (like sweeping empty lines) and this way decreasing the transmission bandwidth.

On the other hand, channel encoding tries to add redundancy in a way that makes the data more robust against the channel disturbances. These obstacles to a clean transmission can be dust or scratches in a typical music Compact Disk (CD) or fading and noise in a cellular wireless network. In these instances codes like Reed-Solomon (RS) and Low Density Parity Check (LDPC) are used to add extra bits to the source data in order for the receiver to have more information to decode correctly. One objective of channel coding theory is to find codes which a) transmit quickly, i.e. have a good code ratio, b) have a reasonable size, i.e. have many valid codewords, and c) perform adequately at correcting or at least detecting errors. The performance in these (not mutually exclusive) areas generally involves a trade off. Therefore there are codes that are more appropriate, i.e. optimal, than others according to the specific application to which they are targeted.

A sub-field of coding theory is algebraic coding theory dealing with the expression and research in algebraic terms of the properties of the codes. Three common properties are codeword length (or period for cyclic codes), code size (total number of valid codewords) and the minimum distance between two valid codewords (e.g. Hamming distance for binary or Lee distance for q-ary alphabet).

The two most researched types of codes are block codes and convolutional codes. In the first type, the encoding of the source data is done in blocks. If the sum of any two codewords is also a valid codeword the the block code is a linear one. Examples include, Reed-Solomon, Golay, Hadamard or Hamming codes [**3**]. To study these codes in a unified way researchers have found a way to relate parameters of different block codes in a form called code bounds. The lower or upper bounds are the result of exploring the relationship between the code rate and distance. The second type of code, i.e. convolutional, takes the fundamental ideal of representing a codeword by the weighted sum of the source symbols. The output of the encoder uses the states of the convolution encoder to convolve with each input bit. In terms of noise protection, in general, a convolutional code does not perform better than an equivalent block code. What it does offer is a greater simplicity of implementation. The encoding process is usually supported by a simple circuit having state memory and feedback logic (normally XOR gates). The decoding process can be implemented in firmware or software. The Viterbi algorithm [**4**] and related variants is the most used algorithm in decoding convolutional codes.

Two other concerns of coding theory revolve around designing codes to achieve synchronization or to better differentiate users in what is called a Code-Division Multiple Access (CDMA) system. In these applications codewords are also referred as sequences. Codes for synchronization purposes are developed so that a phase shift can be readably be detected or corrected allowing multiple signals to be sent on the same channel using the same codeword. In CDMA applications each user is associated with a different code sequence and by evaluating the correlation between different sequences, as a way to differentiate them, several users can share the same channel at the same time.

1.2.2. UWB applications. The convergence of data, entertainment, and mobile communications within the home has created the need for economical technologies and architectures capable of integrating both legacy and new personal area networks. Ultra wideband (UWB) technology is uniquely qualified to address that requirement and was designed specifically to support high data-rate (hundreds of Mbit/s), short range (up to tens of meters), point-to-point wireless communications. UWB, which is an underlay (or sometimes referred as shared unlicensed) system, coexists with other licensed and unlicensed narrowband systems. The transmitted power of UWB devices is controlled by the regulatory agencies, such as the Federal Communications Commission (FCC) in the United States and by the European Conference of Postal and Telecommunications Administrations (CEPT) in Europe, so that narrowband systems are affected from UWB signals only at a negligible level. UWB systems, therefore, are allowed to coexist with other technologies only under stringent power constraints. In spite of this, UWB offers attractive solutions for many wireless communication areas, including wireless personal area networks (WPANs), wireless telemetry and telemedicine, and wireless sensors networks.

With its wide bandwidth, UWB has a potential to offer a capacity much higher than the current narrowband systems for short-range applications.

UWB systems present numerous and unique advantages. First of all it introduces unlicensed usage of an extremely wideband spectrum, as mentioned above. The underlay usage of spectrum greatly increases spectrum efficiency and opens new doors for wireless applications. Excellent time resolution is another key benefit of UWB signals for ranging applications. Due to the extremely short duration of transmitted pulses, sub-decimeter ranging is possible. Robustness against eavesdropping (since UWB signals look like noise) and low power transmission are other benefits of UWB.

UWB has several applications all the way from wireless communications to radar imaging and vehicular radar. The ultra wide bandwidth and hence the wide variety of material penetration capabilities allows UWB to be used for radar imaging systems, including ground penetration radars, wall radar imaging, through-wall radar imaging, surveillance systems, and medical imaging. Images within or behind obstructed objects can be obtained with a high resolution using UWB.

Similarly, the possibility of acute time resolution and associated accurate ranging capability of UWB can be used for vehicular radar systems (including collision avoidance, guided parking, etc.). Positioning location and relative positioning capabilities of UWB systems are other great applications that have recently received significant attention.

Last but not least is the wireless communication application, which is arguably the reason why UWB became part of the wireless world, including wireless home networking, high-density use in office buildings and business cores, UWB wireless mouse, keyboard, wireless speakers, wireless USB, high-speed Wireless Personal Area Networks (WPAN), wireless sensors networks, wireless telemetry, and telemedicine.

1.2.3. State-of-the-Art relating TCH codes. TCH codes were first proposed in [**5**], and further developed in [**6**]. The motivation behind the development of TCH codes was to obtain a sort of pseudo-noise codewords with good correlation properties and length equal to a power of two, i.e., 2^m. The generation of these codes start with the generation of a single codeword, named basic-TCH, through an algebric equation. From then on, the code is extended through several ad-hoc methods (search and trial) in order to find other codewords with similar characteristics (in terms of weight, minimum distance, correlation properties etc.). The performance of these codes was evaluated under All-White Gaussian Noise channels, both alone as well as concatenated with RS codes (with TCH serving as an inner code and a RS code as the outer code).

There have been some investigation work involving TCH codes in the area of satellite communications systems [**7, 8**], in Ultra-wideband systems [**9**], in optimization of the bit-mapping

of codewords [**10, 11**], in synchronization [**12, 13, 14**], in spread-spectrum applications [**15**], in turbo-codes schemes [**16, 17, 18**], and studying the performance in a land mobile satellite channel [**19**], among others.

In view of the above we have tried to develop an efficient receiver that would address, potentially simultaneously, the synchronization, data decoding and channel estimation using fast algorithms and maintaining the same digital frequency processing structure as much as possible.

1.3. Objectives

The main objectives of this thesis are:

To use and evaluate TCH family of codes for UWB, exploiting its circular nature and FFT processing thus shifting signal processing from time to frequency domain.

To extend and generalize TCH codes in order to obtain codewords with length equal to a power of two (besides 16, 256, and 65536 already obtained by a previous method).

To design a receiver, exploiting the algebraic structure of the sequences that correspond to the generalized codewords, using multirate and joint channel decoding and acquisition. A mechanism for data rate adaptation is a very important and desirable characteristic. A receiver may use this mechanism to inform a transmitter of the optimal data rate to increase throughput and/or reduce the frame error rate (FER).

1.4. Thesis overview

In chapter 2 we present and discuss some algebraic facts and core concepts necessary for the theoretical work to be developed in chapters 4 and 5. This includes, sets, relations and mappings, and group theory with emphasis in dihedral and cyclic groups. In chapter 3 we present some background on codes and introduce number theory as a link to finite fields. We then present the generation process behind the development of TCH codes that provided the motivation to pursue further work. In chapter 4 we present a generalized construction of TCH codewords using Zech logarithms. First we use the properties of a dihedral group to reduce by 6 the number of computations for each codeword. Then we generalize the construction of codewords by applying the Zech logarithm to a list of odd integers chosen between 1 and p where p belongs to a specific subclass of primes. For the same codeword length we develop algorithms to obtain one codeword from a different one and for different codeword lengths we develop a permutation matrix to obtain codes with all relevant codewords. In chapter 5 we proceed one step further and generalize the construction of codewords through the use of a small number of permutations. Not only we address the construction in the time domain but we also make it work seamlessly in the frequency domain. This procedure allows for the development of an optimal

frequency domain maximum likelihood receiver that links the theoretical work with practical engineering applications. In addition we extend the construction to address finite fields of order p^m. In this way we extend the algebraic generation from binary to M-ary sequences. In the first part of chapter 6, the applications chapter, we evaluate the performance of the generalized codes presented earlier. We then address the transversal problem of frame synchronization in the context of DS-CDMA systems using cyclic sequences and using signal processing methods in the frequency domain. We also briefly address how those sequences are employed in frequency domain channel estimators. We finish the chapter with the evaluation of a UWB (correlation and multi-rate) receiver proposing to use cyclic sequences for synchronization while also serving to spread and code data.

1.5. Thesis contributions

The core work of this thesis corresponds to the theoretical design of TCH-type codewords or sequences. We provide strong algebraic mechanisms to generate all codewords of a code from a single initial codeword. The construction works both for the time domain as well as for the frequency domain. Moreover we consider not only binary codewords but M-ary codewords and we do not restrict the length of codewords to be $p-1$ with p a Fermat prime. We consider p to belong in a specific class of primes opening a much greater range of codeword lengths as well as allowing us to explore a specific code structure. A summary of results was published in [**20**]. The work developed in chapter 4 is presented in [**21**] and that of chapter 5 in [**22**].

Another contribution uses an FFT signal processing block where the sequence length is partioned into a product, $N = L \cdot M$, with N, L, M positive integers. In the process of obtaining a N-point DFT we make use of M DFT's with L points each. This signal processing core process data in blocks of N samples rearranging them in such a way that an inherent matrix interleaver is involved. These two features are exploited in CDMA systems for synchronism acquisition (besides the usual data decoding), [**12**],[**23**],[**24**].

The application of TCH codewords in UWB systems was presented in [**9**].

Another contribution was made by modifying TCH codes in order to use them in frequency domain channel estimators. The spectrum of a basic TCH polynomial was changed by suppressing the two zero points at $k = 0$ and $k = N/2$. Although this has implications on the code amplitude (for odd samples) it does not alter significantly the code correlation properties.

A brief performance study comparing TCH sequences with the generalized ones, under a channel with Rayleigh fading, is presented in the applications chapter of the thesis.

CHAPTER 2

ALGEBRAIC PRELIMINARIES

2.1. Introduction

In general, a telecommunications engineer does not have a strong background in abstract algebra. It is the purpose of this chapter to introduce the reader to the core material necessary for a deeper understanding of the theory presented in chapters 4 and 5 of this thesis thus making it as much self contained as possible.

To find and study applications of abstract algebra a basic knowledge of set theory, equivalence relations, and matrices is a must. Therefore we have organized this chapter as follows. Sections 2.2 to 2.4 present introductory material including set theory, mappings and partitions. Then in section 2.5 we begin the study of algebraic structures by investigating sets associated with single operations that satisfy certain reasonable axioms. From here on we concentrate on group theory. The material presented here is mostly based on [**1**] but we provide other references for completion.

2.2. Preliminary definitions

In the study of abstract mathematics, we assume some rules about the structure of a collection of objects $\mathscr{S}$. These rules are called *axioms*. Our objective is to obtain other information about $\mathscr{S}$, using logical arguments and the axioms for $\mathscr{S}$.

DEFINITION 1. A *statement* in logic or mathematics is an assertion that is either true or false.

That statement is called a *proposition* if we can prove it to be true.

DEFINITION 2. A proposition of major importance is called a *theorem*.

Sometimes we break the proof a theorem into modules, i.e. into several supporting propositions, which are called *lemmas*, and refer back to these results to prove the main result. If we can prove a proposition or a theorem, it is generally possible to derive other related propositions called *corollaries*.

2.3. Sets and Equivalence Relations

2.3.1. Set Theory. A well-defined collection of objects is called a *set*. It's definition implies that we can determine for any given object x whether or not x belongs to the set. If x belong to a set then it is called an *element* or *member* of the set. We will denote by capital letters, such as A or X; if a is an element of the set A, we write $a \in A$.

Generally, there are two ways to specify a set: by listing all of its elements inside a pair of braces or by stating the property that determines whether or not an object x belongs to the set. Thus, we write

$$X = \{x_1, x_2, \ldots, x_n\}$$

for a set containing elements $x_1, x_2, \ldots, x_n$ or alternatively

$$X = \{x : x \text{ satisfies } \mathscr{P}\},$$

$$X = \{x \mid \text{given } \mathscr{P}\}$$

if each x in X satisfies a certain property $\mathscr{P}$.

Some of the more important sets that we will consider are the following:

$$\mathbb{N} = \{n : n \text{ is a natural number}\} = \{1, 2, 3, 4, \ldots\};$$

$$\mathbb{Z} = \{n : n \text{ is an integer}\} = \{\ldots, -2, -1, 0, 1, 2, 3, \ldots\};$$

$$\mathbb{Q} = \{r : r \text{ is a rational number}\} = \{p/q : p, q \in \mathbb{Z} \text{ where } q \neq 0\};$$

$$\mathbb{R} = \{x : x \text{ is a real number}\};$$

$$\mathbb{C} = \{z : z \text{ is a complex number}\}.$$

We find various relations between sets and can perform operations on sets. A set A is a *subset* of B, written $A \subset B$ or $B \supset A$, if every element of A is also an element of B. For example,

$$\{3, 6, 7\} \subset \{2, 3, 4, 5, 6, 7, 8, 9\}$$

and

$$\mathbb{N} \subset \mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R} \subset \mathbb{C}.$$

As expected, every set is a subset of itself. A set B is a *proper subset* of a set A if $B \subset A$ but $B \neq A$. If A is not a subset of B, we write $A \not\subset B$; for example, $\{2, 3, 7\} \not\subset \{2, 4, 5, 8, 9\}$. Two sets are *equal*, written $A = B$, if we can show that $A \subset B$ and $B \subset A$.

It is convenient to have a set with no elements in it. This set is called the *empty set* and is denoted by $\emptyset$. Note also that the empty set is a subset of every set.

To construct new sets out of old sets, we can perform certain operations: the *union* $A \cup B$ of two sets A and B is defined as

$$A \cup B = \{x : \ x \in A \text{or } x \in B\};$$

the *intersection* of A and B is defined by

$$A \cap B = \{x : \ x \in A \text{and } x \in B\}.$$

In the case of more than two sets, the union and the intersection, respectively, of the collection of sets $A_1, \ldots A_n$ are written

$$\bigcup_{i=1}^{n} A_i = A_1 \cup \ldots \cup A_n$$

and

$$\bigcap_{i=1}^{n} A_i = A_1 \cap \ldots \cap A_n$$

When two sets have no elements in common, they are said to be *disjoint*; for example, if E is the set of even integers and O is the set of odd integers, then E and O are disjoint. Two sets A and B are disjoint exactly when $A \cap B = \emptyset$.

Sometimes we will work within one fixed set U, called the *universal set*. For any set $A \subset U$, we define the *complement* of A, denoted by A', to be the set

$$A' = \{x : \ x \in U \text{and } x \notin A\}.$$

PROPOSITION 3. *Let A, B, and C be sets. Then*

(1) $A \cup A = A$, $A \cap A = A$, *and* $A \setminus A = \emptyset$;
(2) $A \cup \emptyset = A$ *and* $A \cap \emptyset = \emptyset$;
(3) $A \cup (B \cup C) = (A \cup B) \cup C$ *and* $A \cap (B \cap C) = (A \cap B) \cap C$;
(4) $A \cup B = B \cup A$ *and* $A \cap B = B \cap A$;
(5) $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$;
(6) $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

THEOREM 4. *(De Morgan's Laws) Let A and B be sets. Then*

(1) $(A \cup B)' = A' \cap B'$;
(2) $(A \cap B)' = A' \cup B'$.

2.3.2. Cartesian Products and Mappings.

DEFINITION 5. Given sets A and B, we can form a set of ordered pairs, i.e. the new set $A \times B$, called the *Cartesian product* of A and B. That is,

$$A \times B = \{(a,b) : \ a \in A \text{and } b \in B\}.$$

EXAMPLE. If $A = \{x,y\}$, $B = \{1,2,3\}$, and $C = \emptyset$, then $A \times B$ is the set

$$\{(x,1),(x,2),(x,3),(y,1),(y,2),(y,3)\}$$

and

$$A \times C = \emptyset.$$

We define the *Cartesian product of n sets* to be

$$A_1 \times \cdots \times A_n = \{(a_1,\ldots,a_n) : a_i \in A_i \text{ for } i = 1,\ldots,n\}.$$

If $A = A_1 = A_2 = \cdots = A_n$, we often write A^n for $A \times \cdots \times A$ (where A would be written n times). For example, the set $\mathbb{N}^3$ consists of all of 3-tuples of natural numbers. Subsets of $A \times B$ are called *relations*.

DEFINITION 6. A *mapping* or *function* $f \subset A \times B$ from a set A to a set B. This is a special type of relation (in which for each element $a \in A$ there is a unique element $b \in B$ such that $(a,b) \in f$).

Another way of saying this is that for every element in A, f assigns a unique element in B. We usually write $f : A \to B$ or $A \xrightarrow{f} B$. Instead of writing down ordered pairs $(a,b) \in A \times B$, we write $f(a) = b$ or $f : a \mapsto b$. The set A is the *domain* of f and

$$f(A) = \{f(a) : a \in A\} \subset B$$

is the *range* or *image* of f. We can think of the elements in the function's domain as input values and the elements in the function's range as output values.

EXAMPLE. Suppose $A = \{1,2,3\}$ and $B = \{a,b,c,d\}$. In Figure 2.1 we define relations f and g from A to B. The relation f is a mapping, but g is not because $1 \in A$ is not assigned to a unique element in B; that is, $g(1) = a$ and $g(1) = b$.

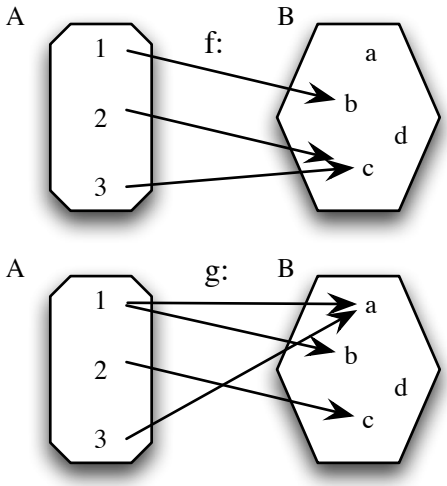


Figure 2.1. Mappings and relations

Given a function $f : A \to B$, it is often possible to write a list describing what the function does to each specific element in the domain. However, not all functions can be described in this manner. For example, the function $f : \mathbb{R} \to \mathbb{R}$ that sends each real number to its cube is a mapping that must be described by writing $f(x) = x^3$ or $f : x \mapsto x^3$. Consider the relation $f : \mathbb{Q} \to \mathbb{Z}$ given by $f(p/q) = p$. We know that $1/2 = 2/4$, but is $f(1/2) = 1$ or 2? This relation cannot be a mapping because it is not well-defined. A relation is *well-defined* if each element in the domain is assigned to a *unique* element in the range.

If $f : A \to B$ is a map and the image of f is B, i.e., $f(A) = B$, then f is said to be *onto* or *surjective*. A map is *one-to-one* or *injective* if $a_1 \neq a_2$ implies $f(a_1) \neq f(a_2)$. Equivalently, a function is one-to-one if $f(a_1) = f(a_2)$ implies $a_1 = a_2$. A *bijective* map is both one-to-one and onto.

Given two functions, we can construct a new function by using the range of the first function as the domain of the second function. Let $f : A \to B$ and $g : B \to C$ be mappings. Define a new map, composing f and g :

DEFINITION 7. The *composition* of f and g from A to C, is defined by $(g \circ f)(x) = g(f(x))$.

EXAMPLE. Consider the functions $f : A \to B$ and $g : B \to C$ that are defined in Figure 2.2(a). The composition of these functions, $g \circ f : A \to C$, is defined in Figure 2.2(b).

In general, order makes a difference; that is, in most cases $f \circ g \neq g \circ f$. One example where $f \circ g = g \circ f$ would be for $f(x) = x^3$ and $g(x) = \sqrt[3]{x}$.

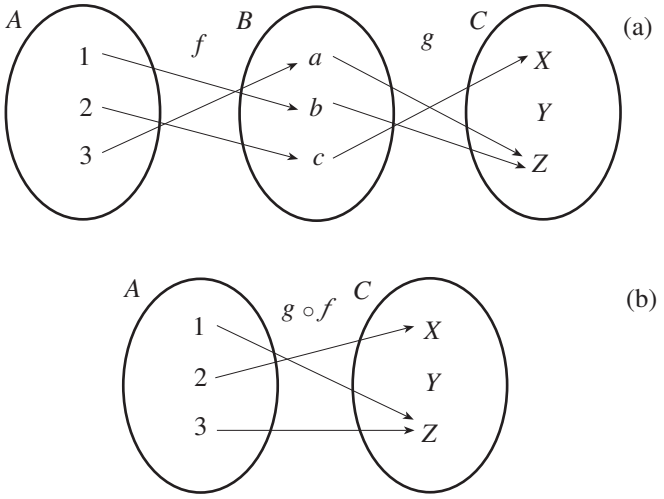


Figure 2.2. Composition of maps [**1**]

EXAMPLE. A matrix multiplication,

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} ax+by \\ cx+dy \end{pmatrix}$$

can be given a as map $T_A : \mathbb{R}^2 \to \mathbb{R}^2$ by $T_A(x,y) = (ax+by, cx+dy)$ for (x,y) in $\mathbb{R}^2$ and using a 2×2 matrix

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}.$$

DEFINITION 8. Maps from $\mathbb{R}^n$ to $\mathbb{R}^m$ given by matrices are called *linear maps* or *linear transformations*.

EXAMPLE. Suppose that $S = \{1,2,3\}$. Define a map $\pi : S \to S$ by

$$\begin{aligned} \pi(1) &= 2 \\ \pi(2) &= 1 \\ \pi(3) &= 3. \end{aligned}$$

This is a bijective map. An alternative way to write π is

$$\begin{pmatrix} 1 & 2 & 3 \\ \pi(1) & \pi(2) & \pi(3) \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix}.$$

DEFINITION 9. For any set S, a one-to-one and onto mapping $\pi : S \to S$ is called a *permutation* of S.

THEOREM 10. *Let $f : A \to B$, $g : B \to C$, and $h : C \to D$. Then*

(1) *The composition of mappings is associative; that is,* $(h \circ g) \circ f = h \circ (g \circ f)$*;*
(2) *If* f *and* g *are both one-to-one, then the composition* $g \circ f$ *(which is another mapping) is one-to-one;*
(3) *If* f *and* g *are both onto, then the composition* $g \circ f$ *is onto;*
(4) *If* f *and* g *are bijective, then so is the composition* $g \circ f$*.*

If S is any set, we will use *id* to denote the *identity mapping* from S to itself. Define this map by $id(s) = s$ for all $s \in S$.

A map $g : B \to A$ is an *inverse mapping* of $f : A \to B$ if $g \circ f = id_A$ and $f \circ g = id_B$; in other words, the inverse function of a function simply "undoes" the function. A map is said to be *invertible* if it has an inverse. We usually write f^{-1} for the inverse of f.

EXAMPLE. The function $f(x) = x^3$ has inverse $f^{-1}(x) = \sqrt[3]{x}$.

EXAMPLE. The natural logarithm and the exponential functions, $f(x) = \ln x$ and $f^{-1}(x) = e^x$, are inverses of each other provided that we are careful about choosing domains. Observe that

$$f(f^{-1}(x)) = f(e^x) = \ln e^x = x$$

and

$$f^{-1}(f(x)) = f^{-1}(\ln x) = e^{\ln x} = x$$

whenever composition makes sense.

EXAMPLE. Given the permutation

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

on $S = \{1,2,3\}$, it is easy to see that the permutation defined by

$$\pi^{-1} = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$$

is the inverse of π. In fact, any bijective mapping possesses an inverse, as we will see in the next theorem.

THEOREM 11. *A mapping is invertible if and only if it is both one-to-one and onto.*

2.3.3. Equivalence Relations and Partitions. To generalize equality we must introduce the concept of equivalence relations and equivalence classes. An *equivalence relation* (on a set X) is a relation $R \subset X \times X$ such that the following properties are obeyed,

- $(x,x) \in R$ for all $x \in X$ (*reflexive*);
- $(x,y) \in R$ implies $(y,x) \in R$ (*symmetric*);
- (x,y) and $(y,z) \in R$ imply $(x,z) \in R$ (*transitive*).

Given an equivalence relation R on a set X, we usually write $x \sim y$ instead of $(x,y) \in R$. If the equivalence relation already has an associated notation such as $=$, $\equiv$, or $\cong$, we will use instead that notation.

EXAMPLE. Let p, q, r, and s be integers, where q and s are nonzero. Define $p/q \sim r/s$ if $ps = qr$. Clearly $\sim$ is reflexive and symmetric. To show that it is also transitive, suppose that $p/q \sim r/s$ and $r/s \sim t/u$, with q, s, and u all nonzero. Then $ps = qr$ and $ru = st$. Therefore,

$$psu = qru = qst.$$

Since $s \neq 0$, $pu = qt$. Consequently, $p/q \sim t/u$.

DEFINITION 12. A *partition* $\mathscr{P}$ of a set X is a collection of nonempty sets $X_1, X_2, \ldots$ such that $X_i \cap X_j = \emptyset$ for $i \neq j$ and $\bigcup_k X_k = X$.

Let $x \in X$ and let $\sim$ be an equivalence relation on a set X. Then $[x] = \{y \in X : y \sim x\}$ is called the *equivalence class* of x. We will see that an equivalence relation gives rise to a partition via equivalence classes. Also, whenever a partition of a set exists, there is some natural underlying equivalence relation:

THEOREM 13. *Given an equivalence relation $\sim$ on a set X, a partition of X is formed by the equivalence classes of X. Conversely, if $\mathscr{P} = \{X_i\}$ is a partition of a set X, then there is an equivalence relation on X with equivalence classes X_i.*

COROLLARY 14. *Two equivalence classes of an equivalence relation are either equal or disjoint.*

EXAMPLE. Let r and s be two integers and suppose that $n \in \mathbb{N}$. We say that r is *congruent to s* modulo n, or r is *congruent to s* mod n, if $r - s$ is evenly divisible by n; that is, $r - s = nk$ for some $k \in \mathbb{Z}$. In this case we write $r \equiv s \pmod{n}$. For example, $41 \equiv 17 \pmod{8}$ since $41 - 17 = 24$ is divisible by 8. We claim that congruence modulo n forms an equivalence relation of $\mathbb{Z}$. Certainly any integer r is equivalent to itself since $r - r = 0$ is divisible by n.

We will now show that the relation is symmetric. If $r \equiv s \pmod{n}$, then $r-s=-(s-r)$ is divisible by n. So $s-r$ is divisible by n and $s \equiv r \pmod{n}$. Now suppose that $r \equiv s \pmod{n}$ and $s \equiv t \pmod{n}$. Then there exist integers k and l such that $r-s=kn$ and $s-t=ln$. To show transitivity, it is necessary to prove that $r-t$ is divisible by n. However,

$$r-t=r-s+s-t=kn+ln=(k+l)n,$$

and so $r-t$ is divisible by n. If we consider the equivalence relation established by the integers modulo 3, then

$$\begin{aligned}
[0] &= \{\ldots,-6,-3,0,3,6,9,\ldots\},\\
[1] &= \{\ldots,-5,-2,1,4,7,10,\ldots\},\\
[2] &= \{\ldots,-4,-1,2,5,8,\ldots\}.
\end{aligned}$$

Notice that $[0]\cup[1]\cup[2]=\mathbb{Z}$ and also that the sets are disjoint. We say that the sets $[0]$, $[1]$, and $[2]$ form a partition of the integers.

The integers modulo n are addressed in more detail in section 2.5 and will become quite useful in discussing various algebraic structures such as groups and rings. For further information about these topics the references [**25, 26, 27, 28, 29**] are appropriate.

2.4. Properties of integers

In this section we will state some fundamental properties of the integers, including mathematical induction and the Fundamental Theorem of Arithmetic.

2.4.1. Mathematical Induction. If we are attempting to verify a statement about some subset S of the positive integers $\mathbb{N}$ on a case-by-case basis, this is an impossible task if S is an infinite set. Instead, we give a specific proof for the smallest integer being considered, followed by a generic argument showing that if the statement holds for a given case, then it must also hold for the next case in the sequence.

As an example suppose we wish to show that

$$1+2+\cdots+n=\frac{n(n+1)}{2}$$

for any natural number n.

The formula is true for $n=1$ since

$$1=\frac{1(1+1)}{2},$$

but it is impossible to verify for all natural numbers on a case-by-case basis. To prove the formula true in general, a more generic method is required.

Suppose we have verified the equation for the first n cases, say $n = 1$, 2, 3, or 4. We can generate the formula for the $(n+1)$th case from this knowledge. If we have verified the first n cases, then

$$\begin{aligned} 1+2+\cdots+n+(n+1) &= \frac{n(n+1)}{2}+n+1 \\ &= \frac{n^2+3n+2}{2} \\ &= \frac{(n+1)[(n+1)+1]}{2}. \end{aligned}$$

This is exactly the formula for the $(n+1)$th case. This method of proof is known as *mathematical induction* and we summarize it in the following axiom.

2.4.2. First Principle of Mathematical Induction. Let $S(n)$ be a statement about integers for $n \in \mathbb{N}$ and suppose $S(n_0)$ is true for some integer n_0. If for all integers k with $k \geq n_0$ $S(k)$ implies that $S(k+1)$ is true, then $S(n)$ is true for all integers n greater than n_0.

We have an equivalent statement of the Principle of Mathematical Induction that is often very useful:

2.4.3. Second Principle of Mathematical Induction. Let $S(n)$ be a statement about integers for $n \in \mathbb{N}$ and suppose $S(n_0)$ is true for some integer n_0. If $S(n_0), S(n_0+1), \ldots, S(k)$ imply that $S(k+1)$ for $k \geq n_0$, then the statement $S(n)$ is true for all integers n greater than n_0.

A nonempty subset S of $\mathbb{Z}$ is *well-ordered* if S contains a least element. Notice that the set $\mathbb{Z}$ is not well-ordered since it does not contain a smallest element. However, the natural numbers are well-ordered.

2.4.4. Principle of Well-Ordering. Every nonempty subset of the natural numbers is well-ordered.

Note that the Principle of Well-Ordering given above is equivalent to the Principle of Mathematical Induction.

LEMMA 15. *The Principle of Mathematical Induction implies that* 1 *is the least positive natural number.*

PROOF. Let $S = \{n \in \mathbb{N} : n \geq 1\}$. Then $1 \in S$. Now assume that $n \in S$; that is, $n \geq 1$. Since $n+1 \geq 1$, $n+1 \in S$; hence, by induction, every natural number is greater than or equal to 1. □

THEOREM 16. *The Principle of Mathematical Induction implies that the natural numbers are well-ordered.*

Induction can also be very useful in formulating definitions. For instance, there are two ways to define $n!$, the factorial of a positive integer n.

- The *explicit* definition: $n! = 1 \cdot 2 \cdot 3 \cdots (n-1) \cdot n$.
- The *inductive* or *recursive* definition: $1! = 1$ and $n! = n(n-1)!$ for $n > 1$.

2.5. The Integers mod n and Symmetries

We want to define an operation on a set in a way that will generalize such familiar structures as the integers $\mathbb{Z}$ together with the single operation of addition, or invertible 2×2 matrices together with the single operation of matrix multiplication. The integers and the 2×2 matrices, together with their respective single operations, are examples of algebraic structures known as groups.

Modern group theory[1] arose from an attempt to find the roots of a polynomial (namely in terms of its coefficients). Groups nowadays play an important role in areas such as coding theory, statistics, and the study of symmetries (with examples in areas of biology, physics and chemistry).

We will address groups in section 2.6 but for now let us introduce some mathematical structures that can be viewed as sets with single operations.

2.5.1. The Integers mod n. In the theory and applications of algebra, the integers mod n have become indispensable. They are used in cryptography, coding theory, and the detection of errors in identification codes. We have already seen that two integers a and b are equivalent mod n if n divides $a-b$. The integers mod n also partition $\mathbb{Z}$ into n different equivalence classes; we will denote the set of these equivalence classes by $\mathbb{Z}_n$. Consider the integers modulo 12 and the corresponding partition of the integers:

$$\begin{aligned} [0] &= \{\ldots, -12, 0, 12, 24, \ldots\}, \\ [1] &= \{\ldots, -11, 1, 13, 25, \ldots\}, \\ &\vdots \\ [11] &= \{\ldots, -13, -1, 11, 23, 35, \ldots\}. \end{aligned}$$

When no confusion can arise, we will use $0, 1, \ldots, 11$ to indicate the equivalence classes $[0], [1], \ldots, [11]$ respectively. We can do arithmetic on $\mathbb{Z}_n$. For two integers a and b, define addition modulo n to

[1] William Burnside's [**30**], first published in 1897 one of the first modern treatments of group theory.

be $(a+b) \pmod{n}$; that is, the remainder when $a+b$ is divided by n. Similarly, multiplication modulo n is defined as $(ab) \pmod{n}$, the remainder when ab is divided by n.

EXAMPLE. The following examples illustrate integer arithmetic modulo n:

$$\begin{aligned} 7+4 &\equiv 1 \pmod 5 & \qquad 7\cdot 3 &\equiv 1 \pmod 5 \\ 3+5 &\equiv 0 \pmod 8 & \qquad 3\cdot 5 &\equiv 7 \pmod 8 \\ 3+4 &\equiv 7 \pmod{12} & \qquad 3\cdot 4 &\equiv 0 \pmod{12}. \end{aligned}$$

In particular, notice that it is possible that the product of two nonzero numbers modulo n can be equivalent to 0 modulo n.

Table 2.1. Multiplication table for $\mathbb{Z}_8$

$\cdot$	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	0	2	4	6
3	0	3	6	1	4	7	2	5
4	0	4	0	4	0	4	0	4
5	0	5	2	7	4	1	6	3
6	0	6	4	2	0	6	4	2
7	0	7	6	5	4	3	2	1

EXAMPLE. Most, but not all, of the usual laws of arithmetic hold for addition and multiplication in $\mathbb{Z}_n$. For instance, it is not necessarily true that there is a multiplicative inverse. Consider the multiplication table for $\mathbb{Z}_8$ in Table 2.1. Notice that 2, 4, and 6 do not have multiplicative inverses; that is, for $n = 2$, 4, or 6, there is no integer k such that $kn \equiv 1 \pmod 8$.

PROPOSITION 17. *Let $\mathbb{Z}_n$ be the set of equivalence classes of the integers mod n and $a, b, c \in \mathbb{Z}_n$.*

(1) *Addition and multiplication are commutative:*

$$\begin{aligned} a+b &\equiv b+a \pmod n \\ ab &\equiv ba \pmod n. \end{aligned}$$

(2) *Addition and multiplication are associative:*

$$\begin{aligned} (a+b)+c &\equiv a+(b+c) \pmod n \\ (ab)c &\equiv a(bc) \pmod n. \end{aligned}$$

(3) *There are both an additive and a multiplicative identity:*

$$\begin{aligned} a+0 &\equiv a \pmod{n} \\ a\cdot 1 &\equiv a \pmod{n}. \end{aligned}$$

(4) *Multiplication distributes over addition:*

$$a(b+c) \equiv ab+ac \pmod{n}.$$

(5) *For every integer a there is an additive inverse $-a$:*

$$a+(-a) \equiv 0 \pmod{n}.$$

(6) *Let a be a nonzero integer. Then* $\gcd(a,n) = 1$ *if and only if there exists a multiplicative inverse b for $a \pmod{n}$; that is, a nonzero integer b such that*

$$ab \equiv 1 \pmod{n}.$$

2.5.2. The Method of Repeated Squares. If we want to compute powers modulo n quickly and efficiently the first thing to notice is that any number a can be written as the sum of distinct powers of 2; that is, we can write

$$a = 2^{k_1} + 2^{k_2} + \cdots + 2^{k_n},$$

where $k_1 < k_2 < \cdots < k_n$. This is just the binary representation of a. For example, the binary representation of 57 is 111001, since we can write $57 = 2^0+2^3+2^4+2^5$. The laws of exponents still work in $\mathbb{Z}_n$; that is, if $b \equiv a^x \pmod{n}$ and $c \equiv a^y \pmod{n}$, then $bc \equiv a^{x+y} \pmod{n}$. We can compute $a^{2^k} \pmod{n}$ in k multiplications by computing

$$\begin{aligned} &a^{2^0} \pmod{n} \\ &a^{2^1} \pmod{n} \\ &\quad\vdots \\ &a^{2^k} \pmod{n}. \end{aligned}$$

Each step involves squaring the answer obtained in the previous step, dividing by n, and taking the remainder.

The method of repeated squares is a very useful tool for RSA cryptography [**31**]. To encode and decode messages in a reasonable manner under this scheme, it is necessary to be able to quickly compute large powers of integers mod n.

2.5.3. Symmetries. A *symmetry* of a geometric figure is a rearrangement of the figure preserving the arrangement of its sides and vertices as well as its distances and angles. A map from the plane to itself preserving the symmetry of an object is called a *rigid motion*. For example,

if we look at the rectangle in Figure 2.3, it is easy to see that a rotation of 180° or 360° returns a rectangle in the plane with the same orientation as the original rectangle and the same relationship among the vertices. A reflection of the rectangle across either the vertical axis or the horizontal axis can also be seen to be a symmetry. However, a 90° rotation in either direction cannot be a symmetry unless the rectangle is a square.

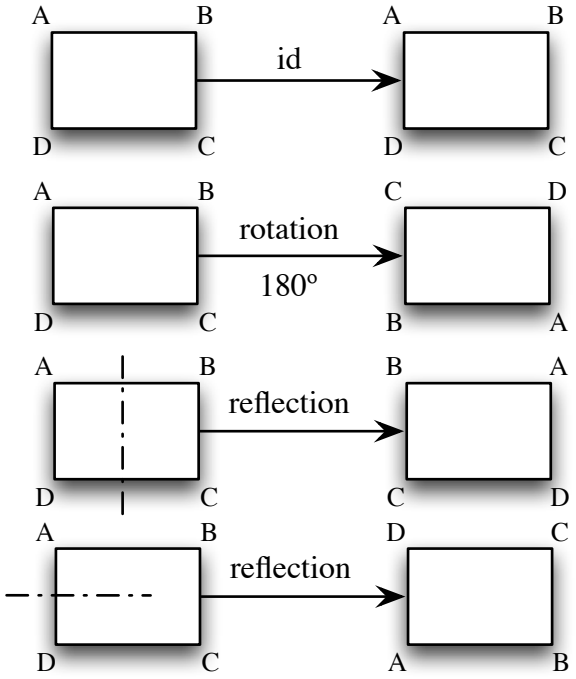
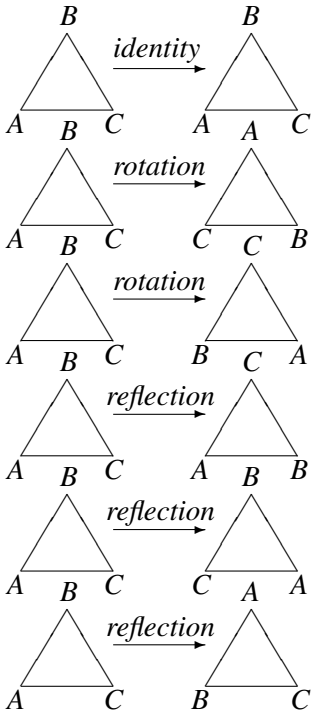


Figure 2.3. Rigid motions of a rectangle

Let us find the symmetries of the equilateral triangle $\triangle ABC$. To find a symmetry of $\triangle ABC$, we must first examine the permutations of the vertices A, B, and C and then ask if a permutation extends to a symmetry of the triangle. Recall that a *permutation* of a set S is a one-to-one and onto map i.e. $\pi : S \to S$. The three vertices have $3! = 6$ permutations, so the triangle has at most six symmetries. To see that there are six permutations, observe there are three different possibilities for the first vertex, and two for the second, and the remaining vertex is determined by the placement of the first two. So we have $3 \cdot 2 \cdot 1 = 3! = 6$ different arrangements. To denote the permutation of the vertices of an equilateral triangle that sends A to B, B to C, and C to A, we write the array

$$\begin{pmatrix} A & B & C \\ B & C & A \end{pmatrix}.$$

Notice that this particular permutation corresponds to the rigid motion of rotating the triangle by 120° in a clockwise direction. In fact, every permutation gives rise to a symmetry of the triangle. All of these symmetries are shown in Figure 2.4.



$$id = \begin{pmatrix} A & B & C \\ A & B & C \end{pmatrix}$$

$$\rho_1 = \begin{pmatrix} A & B & C \\ B & C & A \end{pmatrix}$$

$$\rho_2 = \begin{pmatrix} A & B & C \\ C & A & B \end{pmatrix}$$

$$\mu_1 = \begin{pmatrix} A & B & C \\ A & C & B \end{pmatrix}$$

$$\mu_2 = \begin{pmatrix} A & B & C \\ C & B & A \end{pmatrix}$$

$$\mu_3 = \begin{pmatrix} A & B & C \\ B & A & C \end{pmatrix}$$

Figure 2.4. Symmetries of a triangle

A natural question to ask is what happens if one motion of the triangle $\triangle ABC$ is followed by another. Which symmetry is $\mu_1\rho_1$; that is, what happens when we do the permutation ρ_1 and then the permutation μ_1?[2] We have

$$\begin{aligned} (\mu_1\rho_1)(A) = \mu_1(\rho_1(A)) &= \mu_1(B) = C \\ (\mu_1\rho_1)(B) = \mu_1(\rho_1(B)) &= \mu_1(C) = B \\ (\mu_1\rho_1)(C) = \mu_1(\rho_1(C)) &= \mu_1(A) = A. \end{aligned}$$

This is the same symmetry as μ_2. Suppose we do these motions in the opposite order, ρ_1 then μ_1. It is easy to determine that this is the same as the symmetry μ_3; hence, $\rho_1\mu_1 \neq \mu_1\rho_1$. A multiplication table for the symmetries of an equilateral triangle $\triangle ABC$ is given in Table 2.2. We will refer to this table in chapter 4.

Notice that in the multiplication table for the symmetries of an equilateral triangle, for every motion of the triangle α there is another motion α' such that $\alpha\alpha' = id$; that is, for every motion there is another motion that takes the triangle back to its original orientation.

[2]Remember that we are composing functions here. Although we usually multiply left to right, we compose functions right to left.

Table 2.2. Symmetries of an equilateral triangle

$\circ$	id	ρ_1	ρ_2	μ_1	μ_2	μ_3
id	id	ρ_1	ρ_2	μ_1	μ_2	μ_3
ρ_1	ρ_1	ρ_2	id	μ_3	μ_1	μ_2
ρ_2	ρ_2	id	ρ_1	μ_2	μ_3	μ_1
μ_1	μ_1	μ_2	μ_3	id	ρ_1	ρ_2
μ_2	μ_2	μ_3	μ_1	ρ_2	id	ρ_1
μ_3	μ_3	μ_1	μ_2	ρ_1	ρ_2	id

2.6. Groups

The integers mod n and the symmetries of a triangle or a rectangle are both examples of groups. A *binary operation* or *law of composition* on a set G is a function $G \times G \to G$ that assigns to each pair $(a,b) \in G$ a unique element $a \circ b$, or ab in G, called the composition of a and b. A *group* $(G, \circ)$ is a set G together with a law of composition $(a,b) \mapsto a \circ b$ that satisfies the following axioms.

- The law of composition is *associative*. That is,

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

  for $a, b, c \in G$.
- There exists an element $e \in G$, i.e. the *identity element*, such that for any element $a \in G$

  $$e \circ a = a \circ e = a.$$

- For each element $a \in G$, there exists an *inverse element* a^{-1} in G, such that

  $$a \circ a^{-1} = a^{-1} \circ a = e.$$

A group G with the property that $a \circ b = b \circ a$ for all $a, b \in G$ is called *abelian* or *commutative*. Groups not satisfying this property are said to be *nonabelian* or *noncommutative*.

EXAMPLE. The integers $\mathbb{Z} = \{\ldots, -1, 0, 1, 2, \ldots\}$ form a group under the operation of addition. The binary operation on two integers $m, n \in \mathbb{Z}$ is just their sum. Since the integers under addition already have a well-established notation, we will use the operator $+$ instead of $\circ$; that is, we shall write $m + n$ instead of $m \circ n$. The identity is 0, and the inverse of $n \in \mathbb{Z}$ is written as $-n$ instead of n^{-1}. Notice that the integers under addition have the additional property that $m + n = n + m$ and are therefore an abelian group.

Most of the time we will write ab instead of $a \circ b$; however, if the group already has a natural operation such as addition in the integers, we will use that operation. That is, if we are adding

two integers, we still write $m+n$, $-n$ for the inverse, and 0 for the identity as usual. We also write $m-n$ instead of $m+(-n)$.

Table 2.3. Cayley table for $(\mathbb{Z}_5, +)$

$+$	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

It is often convenient to describe a group in terms of an addition or multiplication table. Such a table is called a *Cayley table*.

EXAMPLE. The integers mod n form a group under addition modulo n. Consider $\mathbb{Z}_5$, consisting of the equivalence classes of the integers 0, 1, 2, 3, and 4. We define the group operation on $\mathbb{Z}_5$ by modular addition. We write the binary operation on the group additively; that is, we write $m+n$. The element 0 is the identity of the group and each element in $\mathbb{Z}_5$ has an inverse. For instance, $2+3=3+2=0$. Table 2.3 is a Cayley table for $\mathbb{Z}_5$. By Proposition 17, $\mathbb{Z}_n = \{0,1,\ldots,n-1\}$ is a group under the binary operation of addition mod n.

EXAMPLE. Not every set with a binary operation is a group. For example, if we let modular multiplication be the binary operation on $\mathbb{Z}_n$, then $\mathbb{Z}_n$ fails to be a group. The element 1 acts as a group identity since $1 \cdot k = k \cdot 1 = k$ for any $k \in \mathbb{Z}_n$; however, a multiplicative inverse for 0 does not exist since $0 \cdot k = k \cdot 0 = 0$ for every k in $\mathbb{Z}_n$. Even if we consider the set $\mathbb{Z}_n \setminus \{0\}$, we still may not have a group. For instance, let $2 \in \mathbb{Z}_6$. Then 2 has no multiplicative inverse since

$$\begin{aligned} 0\cdot 2 &= 0 & \qquad 1\cdot 2 &= 2 \\ 2\cdot 2 &= 4 & \qquad 3\cdot 2 &= 0 \\ 4\cdot 2 &= 2 & \qquad 5\cdot 2 &= 4. \end{aligned}$$

By Proposition 17, every nonzero k does have an inverse in $\mathbb{Z}_n$ if k is relatively prime to n. Denote the set of all such nonzero elements in $\mathbb{Z}_n$ by M_n(sometimes also written $U(n)$). Then M_n is a group called the *group of units* of $\mathbb{Z}_n$. Table 2.4 is a Cayley table for the group M_8. We will discuss this subject further in section 2.7.

Table 2.4. Multiplication table for M_8

$\cdot$	1	3	5	7
1	1	3	5	7
3	3	1	7	5
5	5	7	1	3
7	7	5	3	1

EXAMPLE. The symmetries of an equilateral triangle described in Section 2.5.3 form a nonabelian group. As we observed, it is not necessarily true that $\alpha\beta = \beta\alpha$ for two symmetries α and β. Using Table 2.2, which is a Cayley table for this group, we can easily check that the symmetries of an equilateral triangle are indeed a group. We will denote this group by either S_3 or D_3, for reasons that will be explained later (section A.2).

EXAMPLE. Let $\mathbb{C}^*$ be the set of nonzero complex numbers. This set $\mathbb{C}^*$, under the operation of multiplication, forms a group. The identity is 1. If $z = a + bi$ is a nonzero complex number, then

$$z^{-1} = \frac{a - bi}{a^2 + b^2}$$

is the inverse of z. It is easy to see that the remaining group axioms hold.

If a group contains a finite number of elements it is *finite*, or has *finite order*; otherwise, the group is said to be *infinite* or to have *infinite order*. The *order* of a finite group is the number of elements that it contains. If G is a group containing n elements, we write $|G| = n$. The group $\mathbb{Z}_5$ is a finite group of order 5; the integers $\mathbb{Z}$ form an infinite group under addition, and we sometimes write $|\mathbb{Z}| = \infty$.

2.6.1. Basic Properties of Groups.

PROPOSITION 18. *The identity element in a group G is unique; that is, there exists only one element $e \in G$ such that $eg = ge = g$ for all $g \in G$.*

Inverses in a group are also unique. If there were two different inverses g' and g'' of an element g in a group G, then $gg' = g'g = e$ and $gg'' = g''g = e$.

PROPOSITION 19. *If g is any element in a group G, then the inverse of g, g^{-1}, is unique.*

PROPOSITION 20. *Let G be a group. If $a, b \in G$, then $(ab)^{-1} = b^{-1}a^{-1}$.*

PROPOSITION 21. *Let G be a group. For any $a \in G$, $(a^{-1})^{-1} = a$.*

PROPOSITION 22. *If G is a group and $a, b, c \in G$, then $ba = ca$ implies $b = c$ and $ab = ac$ implies $b = c$.*

This proposition tells us that the *right and left cancellation laws* are true in groups. We can use exponential notation for groups just as we do in ordinary algebra. If G is a group and $g \in G$, then we define $g^0 = e$. For $n \in \mathbb{N}$, we define

$$g^n = \underbrace{g \cdot g \cdots g}_{n \text{ times}}$$

and

$$g^{-n} = \underbrace{g^{-1} \cdot g^{-1} \cdots g^{-1}}_{n \text{ times}}.$$

THEOREM 23. *In a group, the usual laws of exponents hold; that is, for all $g, h \in G$,*

(1) *$g^m g^n = g^{m+n}$ for all $m, n \in \mathbb{Z}$;*
(2) *$(g^m)^n = g^{mn}$ for all $m, n \in \mathbb{Z}$;*
(3) *$(gh)^n = (h^{-1}g^{-1})^{-n}$ for all $n \in \mathbb{Z}$. Furthermore, if G is abelian, then $(gh)^n = g^n h^n$.*

Notice that $(gh)^n \neq g^n h^n$ in general, since the group may not be abelian. If the group is $\mathbb{Z}$ or $\mathbb{Z}_n$, we write the group operation additively and the exponential operation multiplicatively; that is, we write ng instead of g^n. The laws of exponents now become

(1) $mg + ng = (m+n)g$ for all $m, n \in \mathbb{Z}$;
(2) $m(ng) = (mn)g$ for all $m, n \in \mathbb{Z}$;
(3) $m(g+h) = mg + mh$ for all $n \in \mathbb{Z}$.

It is important to realize that the last statement can be made only because $\mathbb{Z}$ and $\mathbb{Z}_n$ are commutative groups.

2.6.2. Subgroups. Sometimes we wish to investigate smaller groups sitting inside a larger group. The set of even integers $2\mathbb{Z} = \{\ldots, -2, 0, 2, 4, \ldots\}$ is a group under the operation of addition. This smaller group sits naturally inside of the group of integers under addition. We define a *subgroup* H of a group G to be a subset H of G such that when the group operation of G is restricted to H, H is a group in its own right. Observe that every group G with at least two elements will always have at least two subgroups, the subgroup consisting of the identity element alone and the entire group itself. The subgroup $H = \{e\}$ of a group G is called the *trivial subgroup*. A subgroup that is a proper subset of G is called a *proper subgroup*.

EXAMPLE. Consider the set of nonzero real numbers, $\mathbb{R}^*$, with the group operation of multiplication. The identity of this group is 1 and the inverse of any element $a \in \mathbb{R}^*$ is just $1/a$. Note that

$$\mathbb{Q}^* = \{p/q : p \text{ and } q \text{ are nonzero integers}\}$$

is a subgroup of $\mathbb{R}^*$. The identity of $\mathbb{R}^*$ is 1; however, $1 = 1/1$ is the quotient of two nonzero integers. Hence, the identity of $\mathbb{R}^*$ is in $\mathbb{Q}^*$. Given two elements in $\mathbb{Q}^*$, say p/q and r/s, their product pr/qs is also in $\mathbb{Q}^*$. The inverse of any element $p/q \in \mathbb{Q}^*$ is again in $\mathbb{Q}^*$ since $(p/q)^{-1} = q/p$. Since multiplication in $\mathbb{R}^*$ is associative, multiplication in $\mathbb{Q}^*$ is associative.

EXAMPLE. Recall that $\mathbb{C}^*$ is the multiplicative group of nonzero complex numbers. Let $H = \{1, -1, i, -i\}$. Then H is a subgroup of $\mathbb{C}^*$. It is quite easy to verify that H is a group under multiplication and that $H \subset \mathbb{C}^*$.

EXAMPLE. One way of telling whether or not two groups are the same is by examining their subgroups. Other than the trivial subgroup and the group itself, the group $\mathbb{Z}_4$ has a single subgroup consisting of the elements 0 and 2. From the group $\mathbb{Z}_2$, we can form another group of four elements as follows. As a set this group is $\mathbb{Z}_2 \times \mathbb{Z}_2$. We perform the group operation coordinate-wise; that is, $(a,b) + (c,d) = (a+c, b+d)$. Table 2.5 is an addition table for $\mathbb{Z}_2 \times \mathbb{Z}_2$. Since there are three nontrivial proper subgroups of $\mathbb{Z}_2 \times \mathbb{Z}_2$, $H_1 = \{(0,0),(0,1)\}$, $H_2 = \{(0,0),(1,0)\}$, and $H_3 = \{(0,0),(1,1)\}$, $\mathbb{Z}_4$ and $\mathbb{Z}_2 \times \mathbb{Z}_2$ must be different groups.

Table 2.5. Addition table for $\mathbb{Z}_2 \times \mathbb{Z}_2$

$+$	$(0,0)$	$(0,1)$	$(1,0)$	$(1,1)$
$(0,0)$	$(0,0)$	$(0,1)$	$(1,0)$	$(1,1)$
$(0,1)$	$(0,1)$	$(0,0)$	$(1,1)$	$(1,0)$
$(1,0)$	$(1,0)$	$(1,1)$	$(0,0)$	$(0,1)$
$(1,1)$	$(1,1)$	$(1,0)$	$(0,1)$	$(0,0)$

Let us examine some criteria for determining exactly when a subset of a group is a subgroup.

PROPOSITION 24. *A subset H of G is a subgroup if and only if it satisfies the following conditions.*

(1) *The identity e of G is in H.*
(2) *If $h_1, h_2 \in H$, then $h_1 h_2 \in H$.*
(3) *If $h \in H$, then $h^{-1} \in H$.*

PROPOSITION 25. *Let H be a subset of a group G. Then H is a subgroup of G if and only if $H \neq \emptyset$, and whenever $g, h \in H$ then gh^{-1} is in H.*

For more advanced topics in group theory see [**32, 33, 34, 35, 36**]. Reference [**27**], in particular, show how group theory can be used in error detection schemes.

2.7. Prime residue groups

This section presents some facts about the structure of the prime residue group denoted by M_m, that is, the multiplicative group of integers coprime to m.

2.7.1. Definitions.

(1) If $(a, m) = 1$ and a is of order e modulo m, the e residue classes a^1, a^2, $\cdots$, a^e, are called the *cycle* of a modulo m. The number of residue classes in M_n is given by the totient function $\phi(m)$ (see definition on page 51).

(2) If a set $\mathscr{S}$ of elements in a group $\mathscr{G}$ is closed under the group operation and contains the identity and the inverse of each of its elements, it is called a *subgroup* of $\mathscr{G}$.

2.7.2. Cycle graphs. It is clear that each cycle of M_m is a cyclic subgroup of M_m. A diagram of a group, which shows every cycle in the group, and the connectivity among these cycles, is called a *cycle graph* of the group. First consider the cycle graph of $M_{15} = \{1, 2, 4, 7, 8, 11, 13, 14\}$ illustrated in Figure 2.5. The powers of 2 (mod 15), namely 2, 4, 8, $16 \equiv 1$, etc., constitute the cycle of 2 modulo 15. This cyclic subgroup of M_{15} has order 4. The element $13 \equiv -2$ (mod 15) is not in this subgroup. Therefore the cycle of 13, which is also of order 4, is connected to the cycle of 2 only at their even powers, that is, it is connected at the quadratic residues. Two other cycles, those of 11 and 14, of order 2 complete the $\phi(15) = 8$ residue classes in M_{15}. No residue class is of order 8 (mod 15) and therefore M_{15} is not cyclic.

Here we are concerned only with the ordering in, and topology of, the cycles. Their actual size, shape and location does not matter. From the graph we can easily read off the powers, order and inverse of every residue class.

2.7.3. Collected knowledge. In the context of cyclic groups it is important to know for which values of m are the M_m cyclic.

THEOREM 26. *(Gauss) M_m is cyclic, that is, m has a primitive root, if and only if m is one of the following:*

$$m = 2, 4, , p^n, 2p^n$$

where p is an odd prime and $n \geq 1$.

For different m there are groups with the same order and therefore another important question is which M_m are isomorphic?

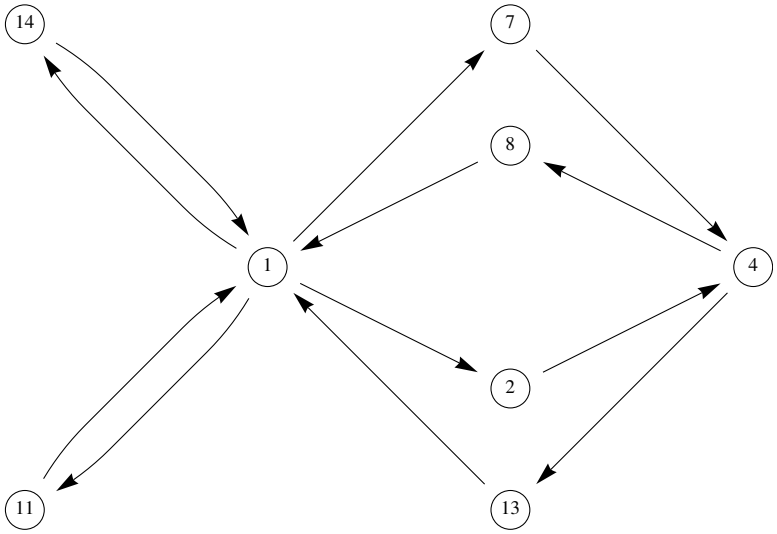


Figure 2.5. Cycle graph of M_{15}

THEOREM 27. *(Shanks factorization ϕ_m) $M_{m'} \simeq M_{m''}$ if and only if the factorizations $\phi_{m'}$ and $\phi_{m''}$ are identical.*

The factorization ϕ_m starts with the *standard factorization* $m = \prod p_i^{\alpha_i}$. The order of M_m is $\phi(m) = \prod p_i^{\alpha_i - 1}(p_i - 1)$.

Next, assume that $p_i - 1 = \prod_j q_{ij}^{\beta_{ij}}$ where q_{ij}, $j = 1, 2, \ldots$, are distinct primes. Then, $\phi(m)$ takes the form

$$\phi(m) = \prod_{i,j} p_i^{\alpha_i - 1} q_{ij}^{\beta_{ij}}.$$

ϕ_m takes the above representation with a minor modification (giving rise to the prime power orders of the abelian groups in the direct product decomposition of M_m). The modification is that if $2^\alpha \mid m$ for $\alpha \geq 3$, then a factor 2 has to be separated from $\phi(2^\alpha)$ and thus $\phi(2^\alpha) = 2^{\alpha-1}$ has to be written $2 \cdot 2^{\alpha-2}$ in such a case.

$m = 15 = 3 \cdot 5$, $\phi_{15} = 2 \cdot 4$
$m = 16 = 2^4$, $\phi_{16} = 2 \cdot 4$

Since ϕ_{15} and ϕ_{16} are identical, $M_{15} \simeq M_{16}$ as illustrated in Figure 2.6. We can also check that $M_{15} \simeq M_{16} \simeq M_{20} \simeq M_{30}$.

Note that for $m = 2^n$, $n > 3$, there are two classes of numbers $8k+1$ and $8k-1$ that play special roles in the structure of M_{2^n}. But the other two classes $8k+3$ and $8k-3$ play similar roles. For $n \geq 3$, r is a quadratic residue if and only if $r = 8k+1$.

It would be interesting if we could characterize the structure of M_m by a formula. That is the subject of the following theorem.

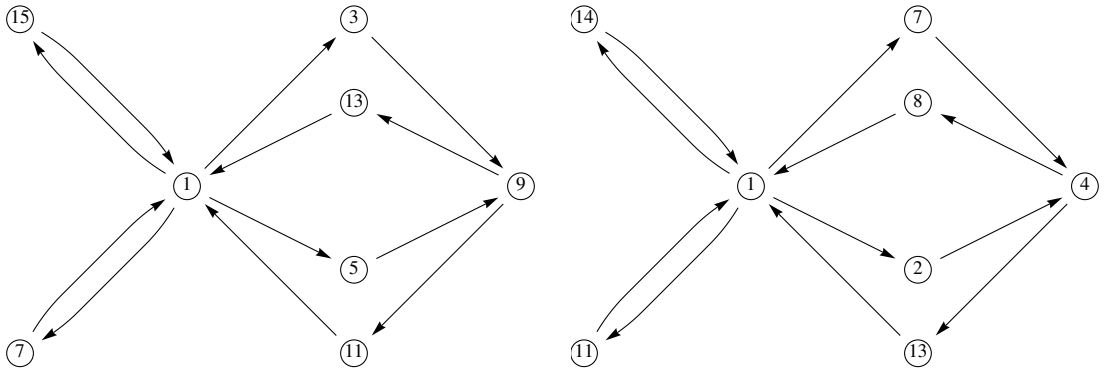


Figure 2.6. Isomorphic cycle graphs, $M_{15} \simeq M_{16}$

THEOREM 28. *(Shanks factorization Φ_m) If Φ_m is the product of s characteristic factors c_i, for each c_i there is a residue class g_i, of order c_i (mod m), such that every residue class a_j in M_m can be expressed as*

$$a_j = g_1^{u_{1,j}} g_2^{u_{2,j}} \cdots g_s^{u_{s,j}}$$

with $0 \leq u_{i,j} < c_i$ in one and only one way. We say that M_m is the direct product of the s cycles of the g_i.

It is well known that every finite abelian group can be written as a direct product of cyclic groups of prime power orders. Moreover, the direct product of two cyclic groups of orders m' and m'', if $(m', m'') = 1$, is itself a cyclic group. Therefore, in general, there is no need to reduce an abelian group into group factors of prime power orders in expressing that group as a product of cyclic groups. The decomposition can be effected with certain suitably chosen larger cyclic group factors. This is the idea behind the alternative factorization Φ_m.

As just explained we can multiply occurring powers of different primes, and thus (sometimes) find several different representations of M_m. Among all possibilities, Φ_m is the one that gives the least number of group factors. This is accomplished by first multiplying all the highest prime powers of each individual prime in the representation ϕ_m. The procedure is then repeated on the prime powers remaining after the first round, and so on.

$m = 65 = 5 \cdot 13$, $\phi_{65} = 4 \cdot 4 \cdot 3$, $\Phi_{65} = 12 \cdot 4$
$m = 104 = 2^3 \cdot 13$, $\phi_{104} = 2 \cdot 2 \cdot 4 \cdot 3$, $\Phi_{65} = 12 \cdot 2 \cdot 2$

The technique described leads to a result in which the first factor found is an integer multiple of all factors subsequently discovered. Those cyclic groups C_{c_i}, the direct product of which give

M_m, thus have orders c_i which all divide the largest of these orders, called Carmichael's function $\lambda(m)$.

If M_m is not cyclic, $a^{\frac{1}{2}\phi(m)} \equiv 1 \pmod{m}$ for every a prime to m. Further, the product R of all residue classes is given by $R = (g_1 g_2 \cdots g_s)^{\frac{1}{2}\phi(m)} \pmod{m}$ and thus $R \equiv -1$ or $R \equiv 1$ according as M_m is cyclic or not.

It is clear that if M_m is cyclic there is an element a of order $\phi(m) \pmod{m}$. On the other hand if M_m is not cyclic we can ask what is the largest order possible within the group? The answer results from a consequence of theorem 28.

THEOREM 29. *(Largest c_i) If c_i are characteristic factors of M_m, then $c_i \mid c_j$ if $i \leq j$. It follows that if c_s is the largest characteristic factor of M_m, $a^{c_s} \equiv 1$ (mod m) for every residue class a in M_m. The largest factor c_s corresponds to the Carmichael function $\lambda(m)$.*

If M_m is cyclic there are $\frac{1}{2}\phi(m)$ quadratic residues.

THEOREM 30. *(Number of quadratic residues) If $m > 2$, and M_m has s characteristic factors, m has $\phi(m) \cdot 2^{-s}$ quadratic residues, and each of these has 2^s square roots.*

Another useful theorem is this:

THEOREM 31. *In every finite Abelian group, if $x^2 = a$ possesses n solutions x, then every square, $y^2 = b$, possesses n solutions. In particular, in M_m, every quadratic residue has an equal number of square roots modulo m.*

Now, we'd like to ask, is every finite Abelian group isomorphic to a subgroup of an M_m?

THEOREM 32. *Every finite Abelian group is isomorphic to a subgroup of M_m for infinitely many different values of m.*

Assume, from group theory, that every finite abelian group $\mathscr{A}$ can be written as a direct product of cyclic subgroups, that is,

$$a_i = g_1^{\alpha_{1,i}} g_2^{\alpha_{2,i}} \cdots g_s^{\alpha_{s,i}}$$

for every a_i in $\mathscr{A}$. The generator g_j is of order m_j and $\mid \mathscr{A} \mid = m_1 m_2 \cdots m_s$. This implies that the cycles of any two generators g_j and g_k have no element in common except the identity $g_j^0 = g_k^0$.

The representation a_i above may be decomposed into cycles of prime power order. Assume this done, and that m_j is now equal to $p_j^{\alpha_j}$ for p_j prime and $\alpha_j \geq 1$.

Now let

$$N = q_1 q_2 \cdots q_s$$

where q_j is a prime of the form $kp_j^{\alpha_j}+1$. Then M_N will contain a cycle of order $q_j-1=kp_j^{\alpha_j}$ generated by a residue class r_j. Further $t_j \equiv r_j^k \pmod N$ has a cycle of order $p_j^{\alpha_j}$ and the subgroup of M_N generated by

$$t_1^{\alpha_{1,i}} t_2^{\alpha_{2,i}} \cdots t_s^{\alpha_{s,i}}$$

is isomorphic to $\mathscr{A}$.

EXAMPLE. Let $\mathscr{A}$ be an abelian group of order 9 represented by $a = x^\alpha y^\beta$ where x and y are elements of $\mathscr{A}$ both being of order 3. For the prime 3, the first primes of the form $k\cdot 3+1$ are 7 and 13 for $k=2,4$ respectively. Now $N = 91 = 7\cdot 13$, ($\phi_{91} = 2\cdot 3\cdot 4\cdot 3$, $\Phi_{91} = 6\cdot 12$). Therefore $\mathscr{A}$ is isomorphic to a subgroup of M_{91}. A representation of M_{91} is $g = 66^\alpha 15^\beta \pmod{91}$ with 66 of order 6, and 15 of order 12. Then $66^2 \equiv 79$, and $15^4 \equiv 29$, are both of order 3 and $\mathscr{A}$ is isomorphic to the subgroup of M_{91} given by $a = 79^\alpha 29^\beta \pmod{91}$.

2.7.4. Isomorphism with cyclic groups. The group M_n of residue classes (or congruence classes) prime to n under multiplication modulo n is the group of units (elements with a multiplicative inverse) of the ring of integers modulo n, i.e. $M_n \cong \mathbb{Z}_n^\times$. It is straightforward to show that this group is abelian. Modulo multiplication groups such as M_n are isomorphic to cyclic groups C_n. The structure of M_n depends on n. Assume first that n is a power of two. For $n=2$, there is only one relatively prime congruence class, 1, so $Z_2^\times \cong \{1\}$ is trivial. For $n=4$, there are two relatively prime congruence classes, 1 and 3, so $Z_4^\times \cong C_2$, the cyclic group with 2 elements. Modulo 8 there are four relatively prime congruence classes, 1, 3, 5 and 7. The square of each of these is 1 so $Z_8^\times \cong C_2 \times C_2$, also known as the Klein four-group. Modulo 16 there are eight relatively prime residue classes, 1, 3, 5, 7, 9, 11, 13, and 15. Note that $\{\pm 1, \pm 7\} \cong C_2 \times C_2$ is the 2-torsion subgroup (i.e. the square of each element is 1). The powers of 3, {1, 3, 9, 11} are a subgroup of order 4, as are the powers of 5, {1, 5, 9, 13}. Thus, $Z_{16}^\times \cong C_2 \times C_4$. The pattern occurring for $n=8$ and $n=16$ holds for higher powers 2^k, $k>2$: $\{\pm 1, 2^{k-1}\pm 1\} \cong C_2 \times C_2$ is the 2-torsion subgroup (so $Z_{2^k}^\times$ is not cyclic). Since the powers of 3 are a subgroup of order 2^{k-2}, it follows $Z_{2^k}^\times \cong C_2 \times C_{2^{k-2}}$. Next we consider n as a power of an odd prime. Then, $Z_{p^k}^\times \cong C_{\phi(p^k)}$. Lastly, for the composite case, $n = p_1^{k_1} p_2^{k_2} \cdots$, the group of units is the direct product of the groups corresponding to each of the prime power factors, i.e. $Z_n^\times \cong Z_{p_1^{k_1}}^\times \times Z_{p_2^{k_2}}^\times \times \cdots$. In general, the abstract group corresponding to a given M_n can be determined explicitly in terms of a group direct product of cyclic groups of the so called characteristic factors, whose product is denoted by $\Phi(n)$, [**37**]. Note that the group order $|\, M_n \,| = \phi(n)$ is the product of the orders of the cyclic groups in the direct product. Note also that the least common multiple of the orders of the cyclic groups is given by the Carmichael function $\lambda(n)$, meaning that if g and n are relatively prime, $g^{\lambda(n)} \equiv 1 \pmod n$.

2.8. Cyclic Groups

The groups $\mathbb{Z}$ and $\mathbb{Z}_n$, are both examples of what are called cyclic groups. In this section we will study the properties of cyclic groups and cyclic subgroups, which play a fundamental part in the classification of all abelian groups.

Often a subgroup will depend entirely on a single element of the group; that is, knowing that particular element will allow us to compute any other element in the subgroup.

EXAMPLE. Suppose that we consider $3 \in \mathbb{Z}$ and look at all multiples (both positive and negative) of 3. As a set, this is

$$3\mathbb{Z} = \{\ldots, -3, 0, 3, 6, \ldots\}.$$

It is easy to see that $3\mathbb{Z}$ is a subgroup of the integers. This subgroup is completely determined by the element 3 since we can obtain all of the other elements of the group by taking multiples of 3. Every element in the subgroup is "generated" by 3.

EXAMPLE. If $H = \{2^n : n \in \mathbb{Z}\}$, then H is a subgroup of the multiplicative group of nonzero rational numbers, $\mathbb{Q}^*$. If $a = 2^m$ and $b = 2^n$ are in H, then $ab^{-1} = 2^m 2^{-n} = 2^{m-n}$ is also in H. By proposition 25, H is a subgroup of $\mathbb{Q}^*$ determined by the element 2.

THEOREM 33. *Let G be a group and a be any element in G. Then the set*

$$\langle a \rangle = \{a^k : k \in \mathbb{Z}\}$$

is a subgroup of G. Furthermore, $\langle a \rangle$ is the smallest subgroup of G that contains a.

REMARK. If we are using the "+" notation, as in the case of the integers under addition, we write $\langle a \rangle = \{na : n \in \mathbb{Z}\}$.

For $a \in G$, we call $\langle a \rangle$ the *cyclic subgroup* generated by a. If G contains some element a such that $G = \langle a \rangle$, then G is a *cyclic group*. In this case a is a *generator* of G. If a is an element of a group G, we define the *order* of a to be the smallest positive integer n such that $a^n = e$, and we write $|a| = n$. If there is no such integer n, we say that the order of a is infinite and write $|a| = \infty$ to denote the order of a.

EXAMPLE. Notice that a cyclic group can have more than a single generator. Additively, both 1 and 5 generate $\mathbb{Z}_6$; hence, $\mathbb{Z}_6$ is a cyclic group. Not every element in a cyclic group is necessarily a generator of the group. The order of $2 \in \mathbb{Z}_6$ is 3. The cyclic subgroup generated by 2 is $\langle 2 \rangle = \{0, 2, 4\}$.

The groups $\mathbb{Z}$ and $\mathbb{Z}_n$ are cyclic groups. The elements 1 and -1 are generators for $\mathbb{Z}$. We can certainly generate $\mathbb{Z}_n$ with 1 although there may be other generators of $\mathbb{Z}_n$, as in the case of $\mathbb{Z}_6$.

EXAMPLE. The group of units, M_9, in $\mathbb{Z}_9$ is a cyclic group. As a set, M_9 is $\{1,2,4,5,7,8\}$. The element 2 is a generator for M_9 since

$$\begin{aligned} 2^1 &= 2 & 2^2 &= 4 \\ 2^3 &= 8 & 2^4 &= 7 \\ 2^5 &= 5 & 2^6 &= 1. \end{aligned}$$

Not every group is a cyclic group. Consider the symmetry group of an equilateral triangle S_3. The multiplication table for this group is Table 2.2. The subgroups of S_3 are shown in Figure 2.7. Note that every subgroup is cyclic but no single element generates the entire group.

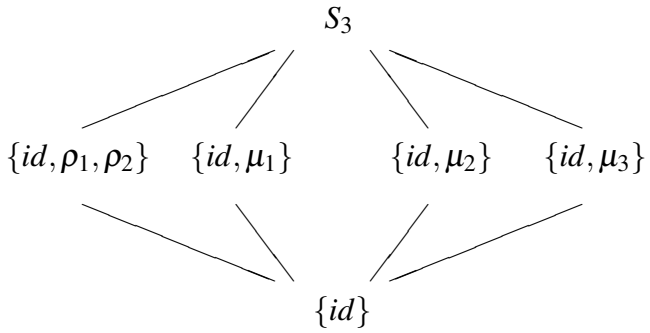


Figure 2.7. Subgroups of S_3

THEOREM 34. *Every cyclic group is abelian.*

We can ask some interesting questions about cyclic subgroups of a group and subgroups of a cyclic group. If G is a group, which subgroups of G are cyclic? If G is a cyclic group, what type of subgroups does G possess?

THEOREM 35. *Every subgroup of a cyclic group is cyclic.*

PROPOSITION 36. *Let G be a cyclic group of order n and suppose that a is a generator for G. Then $a^k = e$ if and only if n divides k.*

THEOREM 37. *Let G be a cyclic group of order n and suppose that $a \in G$ is a generator of the group. If $b = a^k$, then the order of b is n/d, where $d = \gcd(k,n)$.*

COROLLARY 38. *The generators of $\mathbb{Z}_n$ are the integers r such that $1 \leq r < n$ and* $\gcd(r,n) = 1$.

EXAMPLE. Let us examine the group $\mathbb{Z}_{16}$. The numbers 1, 3, 5, 7, 9, 11, 13, and 15 are the elements of $\mathbb{Z}_{16}$ that are relatively prime to 16. Each of these elements generates $\mathbb{Z}_{16}$. For example,

$$\begin{array}{lll} 1\cdot 9 = 9 & 2\cdot 9 = 2 & 3\cdot 9 = 11 \\ 4\cdot 9 = 4 & 5\cdot 9 = 13 & 6\cdot 9 = 6 \\ 7\cdot 9 = 15 & 8\cdot 9 = 8 & 9\cdot 9 = 1 \\ 10\cdot 9 = 10 & 11\cdot 9 = 3 & 12\cdot 9 = 12 \\ 13\cdot 9 = 5 & 14\cdot 9 = 14 & 15\cdot 9 = 7. \end{array}$$

2.9. The multiplicative group of complex numbers

The *complex numbers* are defined as $\mathbb{C} = \{a+bj : a,b \in \mathbb{R}\}$, where $j^2 = -1$. If $z = a+bj$, then a is the *real part* of z and b is the *imaginary part* of z. Every nonzero complex number $z = a+bj$ has a multiplicative inverse; that is, there exists a $z^{-1} \in \mathbb{C}^*$ such that $zz^{-1} = z^{-1}z = 1$. The *complex conjugate* of a complex number $z = a+bj$ is defined to be $z^* = a-bj$. The *absolute value* or *modulus* of $z = a+bj$ is $|z| = \sqrt{a^2+b^2}$.

We can represent a complex number $z = a+bj$ as an ordered pair on the xy plane where a is the x (or real) coordinate and b is the y (or imaginary) coordinate. This is called the *rectangular* or *Cartesian* representation. Nonzero complex numbers can also be represented using *polar coordinates*. To specify any nonzero point on the plane, it suffices to give an angle θ from the positive x axis in the counterclockwise direction and a distance r from the origin. We can see that

$$z = a+bj = r(\cos\theta + j\sin\theta).$$

Hence,

$$r = |z| = \sqrt{a^2+b^2}$$

and

$$\begin{aligned} a &= r\cos\theta \\ b &= r\sin\theta. \end{aligned}$$

We sometimes abbreviate $r(\cos\theta + j\sin\theta)$ as $r\operatorname{cis}\theta$. To assure that the representation of z is well-defined, we also require that $0^\circ \leq \theta < 360^\circ$. If the measurement is in radians, then $0 \leq \theta < 2\pi$.

The polar representation of a complex number makes it easy to find products and powers of complex numbers.

PROPOSITION 39. *Let $z = r\operatorname{cis}\theta$ and $w = s\operatorname{cis}\phi$ be two nonzero complex numbers. Then*

$$zw = rs\operatorname{cis}(\theta+\phi).$$

THEOREM 40. *(DeMoivre) Let $z = r\operatorname{cis}\theta$ be a nonzero complex number. Then*

$$[r\operatorname{cis}\theta]^n = r^n\operatorname{cis}(n\theta)$$

for $n = 1, 2, \ldots$.

The Circle Group and the Roots of Unity. The multiplicative group of the complex numbers, $\mathbb{C}^*$, possesses some interesting subgroups. Whereas $\mathbb{Q}^*$ and $\mathbb{R}^*$ have no interesting subgroups of finite order, $\mathbb{C}^*$ has many. We first consider the *circle group*,

$$\mathbb{T} = \{z \in \mathbb{C} : |z| = 1\}.$$

PROPOSITION 41. *The circle group is a subgroup of $\mathbb{C}^*$.*

Although the circle group has infinite order, it has many interesting finite subgroups. Suppose that $H = \{1, -1, i, -i\}$. Then H is a subgroup of the circle group. Also, 1, -1, i, and $-i$ are exactly those complex numbers that satisfy the equation $z^4 = 1$. The complex numbers satisfying the equation $z^n = 1$ are called the *nth roots of unity*.

THEOREM 42. *If $z^n = 1$, then the nth roots of unity are*

$$z = \operatorname{cis}\left(\frac{2k\pi}{n}\right),$$

where $k = 0, 1, \ldots, n-1$. The nth roots of unity form a cyclic subgroup of $\mathbb{T}$ and its order is n.

DEFINITION 43. A generator for the group of the nth roots of unity is called a *primitive nth root of unity*.

EXAMPLE. The 8th roots of unity can be represented as eight equally spaced points on the unit circle (Figure 2.8). The primitive 8th roots of unity are

$$\begin{aligned} \omega &= \frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2} j \\ \omega^3 &= -\frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2} j \\ \omega^5 &= -\frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2} j \\ \omega^7 &= \frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2} j. \end{aligned}$$

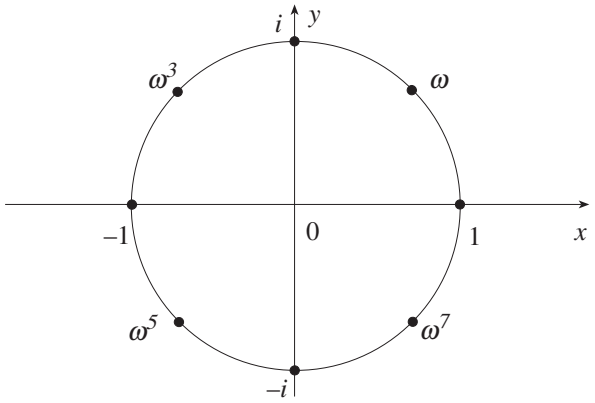


Figure 2.8. 8th roots of unity

2.10. Conclusion

In this chapter we provided important definitions and outlined, in theorem form, some results of algebraic structures, namely groups. We left the proofs outside the main text since these can be readily found in standard algebra books like those that were listed throughout. However, we used some examples to clarify the concepts.

After an introduction to basic material like set theory, mappings and partitions we concentrated on group theory. In chapter 4 we use group geometric properties and relate some operations on codes as equivalent to the symmetries of the dihedral group. In chapter 5, we focus mostly on group permutations as an efficient way to generate all codewords of a particular cyclic code. Both in chapter 4 and 5 we will provide links to the appropriate sections in this chapter to make the reading as straightforward as possible.

CHAPTER 3

FUNDAMENTALS OF CODING AND NUMBER THEORY

3.1. Introduction

Coding theory is an application of algebra that has become increasingly important over the last several decades. When we transmit data, we are concerned about sending a message over a channel that could be affected by "noise." We wish to be able to encode and decode the information in a manner that will allow the detection, and possibly the correction, of errors caused by noise. This situation arises in many areas of communications, including radio, telephone, television, computer communications, and even satellite technology. Probability, combinatorics, group theory, linear algebra, and polynomial rings over finite fields all play important roles in coding theory. However in this chapter we concentrate on the fundamentals, first giving an introduction to algebraic coding theory and then presenting some fundamental results of number theory. Both are important in providing the context for the presentation of TCH codes that were the starting point for the further developments presented in this thesis. Since TCH codewords were represented via a connection with primitive elements of a finite field we also make a bridge between the group theory presented in the previous chapter and the essentials of finite fields.

3.2. Error-Detecting and Correcting Codes

The material presented on the next two sections is mostly based on [**1**] but we provide other references for completion. Let us examine a simple model of a communications system for transmitting and receiving coded messages (Figure 3.1).

Uncoded messages may be composed of letters or characters, but typically they consist of binary m-tuples. These messages are encoded into codewords, consisting of binary n-tuples, by a device called an *encoder*. The message is transmitted and then decoded. We will consider the occurrence of errors during transmission. An *error* occurs if there is a change in one or

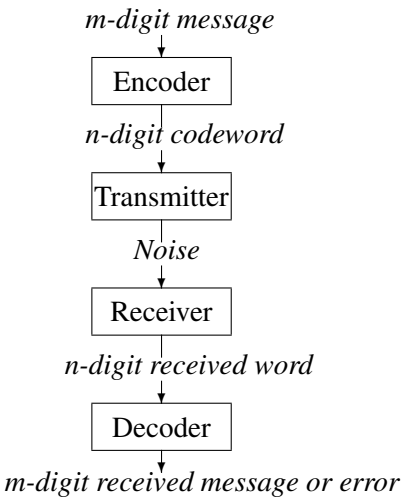


Figure 3.1. Encoding and decoding messages

more bits in the codeword. A *decoding scheme* is a method that either converts an arbitrarily received n-tuple into a meaningful decoded message or gives an error message for that n-tuple. If the received message is a codeword (one of the special n-tuples allowed to be transmitted), then the decoded message must be the unique message that was encoded into the codeword. For received noncodewords, the decoding scheme will give an error indication, or, if we are more clever, will actually try to correct the error and reconstruct the original message. Our goal is to transmit error-free messages as cheaply and quickly as possible.

EXAMPLE. One possible coding scheme would be to send a message several times and to compare the received copies with one another. Suppose that the message to be encoded is a binary n-tuple $(x_1, x_2, \ldots, x_n)$. The message is encoded into a binary $3n$-tuple by simply repeating the message three times:

$$(x_1, x_2, \ldots, x_n) \mapsto (x_1, x_2, \ldots, x_n, x_1, x_2, \ldots, x_n, x_1, x_2, \ldots, x_n).$$

To decode the message, we choose as the ith digit the one that appears in the ith place in at least two of the three transmissions. For example, if the original message is (0110), then the transmitted message will be (011001100110). If there is a transmission error in the fifth digit, then the received codeword will be (011011100110), which will be correctly decoded as (0110).[1] This triple-repetition method will automatically detect and correct all single errors, but it is slow and inefficient: to send a message consisting of n bits, $2n$ extra bits are required, and

[1] In this chapter we will adopt the convention that bits are numbered left to right in binary n-tuples, therefore bit 1 is the most significant bit and bit n the least significant one.

we can only detect and correct single errors. We will see that it is possible to find an encoding scheme that will encode a message of n bits into m bits with m much smaller than $3n$.

EXAMPLE. *Even parity*, a commonly used coding scheme, is much more efficient than the simple repetition scheme. The ASCII (American Standard Code for Information Interchange) coding system uses binary 8-tuples, yielding $2^8 = 256$ possible 8-tuples. However, only seven bits are needed since there are only $2^7 = 128$ ASCII characters. What can or should be done with the extra bit? Using the full eight bits, we can detect single transmission errors. For example, the ASCII codes for A, B, and C are

$$\begin{aligned} \text{A} &= 65_{10} = 01000001_2, \\ \text{B} &= 66_{10} = 01000010_2, \\ \text{C} &= 67_{10} = 01000011_2. \end{aligned}$$

Notice that the leftmost bit is always set to 0; that is, the 128 ASCII characters have codes

$$\begin{aligned} 00000000_2 &= 0_{10}, \\ &\vdots \\ 01111111_2 &= 127_{10}. \end{aligned}$$

The bit can be used for error checking on the other seven bits. It is set to either 0 or 1 so that the total number of 1 bits in the representation of a character is even. Using even parity, the codes for A, B, and C now become

$$\begin{aligned} \text{A} &= 01000001_2, \\ \text{B} &= 01000010_2, \\ \text{C} &= 11000011_2. \end{aligned}$$

Suppose an A is sent and a transmission error in the sixth bit is caused by noise over the communication channel so that (01000101) is received. We know an error has occurred since the received word has an odd number of 1's, and we can now request that the codeword be transmitted again. When used for error checking, the leftmost bit is called a *parity check bit*. By far the most common error-detecting codes used in computers are based on the addition of a parity bit. Typically, a computer stores information in m-tuples called *words*. Common word lengths are 8, 16, and 32 bits. One bit in the word is set aside as the parity check bit, and is not used to store information. This bit is set to either 0 or 1, depending on the number of 1's in the word. Adding a parity check bit allows the detection of all single errors because changing a single bit either increases or decreases the number of 1's by one, and in either case the parity has been changed from even to odd, so the new word is not a codeword. (We could also construct an

error detection scheme based on *odd parity*; that is, we could set the parity check bit so that a codeword always has an odd number of 1's.)

The even parity system is easy to implement, but has two drawbacks. First, multiple errors are not detectable. Suppose an A is sent and the first and seventh bits are changed from 0 to 1. The received word is a codeword, but will be decoded into a C instead of an A. Second, we do not have the ability to correct errors. If the 8-tuple (10011000) is received, we know that an error has occurred, but we have no idea which bit has been changed. We will now investigate a coding scheme that will not only allow us to detect transmission errors but will actually correct the errors.

Table 3.1. A repetition code

		Received Word							
		000	001	010	011	100	101	110	111
Transmitted	000	0	1	1	2	1	2	2	3
Codeword	111	3	2	2	1	2	1	1	0

EXAMPLE. Suppose that our original message is either a 0 or a 1, and that 0 encodes to (000) and 1 encodes to (111). If only a single error occurs during transmission, we can detect and correct the error. For example, if a 101 is received, then the second bit must have been changed from a 1 to a 0. The originally transmitted codeword must have been (111). This method will detect and correct all single errors. In Table 3.1, we present all possible words that might be received for the transmitted codewords (000) and (111). Table 3.1 also shows the number of bits by which each received 3-tuple differs from each original codeword.

Maximum-Likelihood Decoding[2]. The coding scheme presented in the last example is not a complete solution to the problem because it does not account for the possibility of multiple errors. For example, either a (000) or a (111) could be sent and a (001) received. We have no means of deciding from the received word whether there was a single error in the third bit or two errors, one in the first bit and one in the second. No matter what coding scheme is used, an incorrect message could be received: we could transmit a (000), have errors in all three bits, and receive the codeword (111). It is important to make explicit assumptions about the likelihood and distribution of transmission errors so that, in a particular application, it will be known whether a given error detection scheme is appropriate. We will assume that transmission errors are rare, and, that when they do occur, they occur independently in each bit; that is, if p is the probability of an error in one bit and q is the probability of an error in a different bit,

[2]This section requires a knowledge of probability, but can be skipped without loss of continuity.

then the probability of errors occurring in both of these bits at the same time is pq. We will also assume that a received n-tuple is decoded into a codeword that is closest to it; that is, we assume that the receiver uses *maximum-likelihood decoding*.

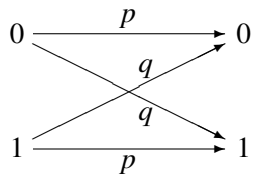


Figure 3.2. Binary symmetric channel

A *binary symmetric channel* is a model that consists of a transmitter capable of sending a binary signal, either a 0 or a 1, together with a receiver. Let p be the probability that the signal is correctly received. Then $q = 1 - p$ is the probability of an incorrect reception. If a 1 is sent, then the probability that a 1 is received is p and the probability that a 0 is received is q (Figure 3.2). The probability that no errors occur during the transmission of a binary codeword of length n is p^n. For example, if $p = 0.999$ and a message consisting of 10,000 bits is sent, then the probability of a perfect transmission is

$$(0.999)^{10,000} \approx 0.00005.$$

THEOREM 44. *If a binary n-tuple $(x_1, \ldots, x_n)$ is transmitted across a binary symmetric channel with probability p that no error will occur in each coordinate, then the probability that there are errors in exactly k coordinates is*

$$\binom{n}{k} q^k p^{n-k}.$$

PROOF. Fix k different coordinates. We first compute the probability that an error has occurred in this fixed set of coordinates. The probability of an error occurring in a particular one of these k coordinates is q; the probability that an error will not occur in any of the remaining $n-k$ coordinates is p. The probability of each of these n independent events is $q^k p^{n-k}$. The number of possible error patterns with exactly k errors occurring is equal to

$$\binom{n}{k} = \frac{n!}{k!(n-k)!},$$

the number of combinations of n things taken k at a time. Each of these error patterns has probability $q^k p^{n-k}$ of occurring; hence, the probability of all of these error patterns is

$$\binom{n}{k} q^k p^{n-k}.$$

□

EXAMPLE. Suppose that $p = 0.995$ and a 500-bit message is sent. The probability that the message was sent error-free is

$$p^n = (0.995)^{500} \approx 0.082.$$

The probability of exactly one error occurring is

$$\begin{pmatrix} n \\ 1 \end{pmatrix} qp^{n-1} = 500(0.005)(0.995)^{499} \approx 0.204.$$

The probability of exactly two errors is

$$\begin{pmatrix} n \\ 2 \end{pmatrix} q^2 p^{n-2} = \frac{500 \cdot 499}{2}(0.005)^2(0.995)^{498} \approx 0.257.$$

The probability of more than two errors is approximately

$$1 - 0.082 - 0.204 - 0.257 = 0.457.$$

Block Codes. If we are to develop efficient error-detecting and error-correcting codes, we will need more sophisticated mathematical tools. Group theory will allow faster methods of encoding and decoding messages. A code is an (n,m)-*block code* if the information that is to be coded can be divided into blocks of m binary digits, each of which can be encoded into n binary digits. More specifically,

DEFINITION 45. A (n,m)-block binary code consists of an *encoding function*

$$E : \mathbb{Z}_2^m \to \mathbb{Z}_2^n$$

and a *decoding function*

$$D : \mathbb{Z}_2^n \to \mathbb{Z}_2^m.$$

A *codeword* is any element in the image of E. We also require that E be one-to-one so that two information blocks will not be encoded into the same codeword. If our code is to be error-correcting, then D must be onto.

EXAMPLE. The even-parity coding system developed to detect single errors in ASCII characters is an $(8,7)$-block code. The encoding function is

$$E(x_7, x_6, \ldots, x_1) = (x_8, x_7, \ldots, x_1),$$

where $x_8 = x_7 + x_6 + \cdots + x_1$ with addition in $\mathbb{Z}_2$.

Let $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$ be binary n-tuples.

DEFINITION 46. The *Hamming distance* or *distance*, $d(\mathbf{x}, \mathbf{y})$, between $\mathbf{x}$ and $\mathbf{y}$ is the number of bits in which $\mathbf{x}$ and $\mathbf{y}$ differ.

The distance between two codewords is the minimum number of transmission errors required to change one codeword into the other.

DEFINITION 47. The *minimum distance* for a code, $d_{\min}$, is the minimum of all distances $d(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}$ and $\mathbf{y}$ are distinct codewords.

Another important concept is that of codeword weight:

DEFINITION 48. The *weight*, $w(\mathbf{x})$, of a binary codeword $\mathbf{x}$ is the number of 1's in $\mathbf{x}$. Clearly, $w(\mathbf{x}) = d(\mathbf{x}, \mathbf{0})$, where $\mathbf{0} = (00\cdots0)$.

The above definitions are made clear with the following example.

EXAMPLE. Let $\mathbf{x} = (10101)$, $\mathbf{y} = (11010)$, and $\mathbf{z} = (00011)$ be all of the codewords in some code C. Then we have the following Hamming distances:

$$\begin{aligned} d(\mathbf{x}, \mathbf{y}) &= 4, \\ d(\mathbf{x}, \mathbf{z}) &= 3, \\ d(\mathbf{y}, \mathbf{z}) &= 3. \end{aligned}$$

The minimum distance for this code is 3. We also have the following weights:

$$\begin{aligned} w(\mathbf{x}) &= 3, \\ w(\mathbf{y}) &= 3, \\ w(\mathbf{z}) &= 2. \end{aligned}$$

The following proposition lists some basic properties about the weight of a codeword and the distance between two codewords.

PROPOSITION 49. *Let* $\mathbf{x}$, $\mathbf{y}$, *and* $\mathbf{z}$ *be binary n-tuples. Then* $w(\mathbf{x}) = d(\mathbf{x}, \mathbf{0})$ *and*

- $d(\mathbf{x}, \mathbf{y}) \geq 0$*;*
- $d(\mathbf{x}, \mathbf{y}) = 0$ *exactly when* $\mathbf{x} = \mathbf{y}$*;*
- $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$*;*
- $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z}) + d(\mathbf{z}, \mathbf{y})$.

The weights in a particular code are usually much easier to compute than the Hamming distances between all codewords in the code. If a code is set up carefully, we can use this fact to our advantage. Suppose that $\mathbf{x} = (1101)$ and $\mathbf{y} = (1100)$ are codewords in some code. If we transmit (1101) and an error occurs in the rightmost bit, then (1100) will be received. Since (1100) is a codeword, the decoder will decode (1100) as the transmitted message. This code is clearly not very appropriate for error detection. The problem is that $d(\mathbf{x},\mathbf{y}) = 1$. If $\mathbf{x} = (1100)$ and $\mathbf{y} = (1010)$ are codewords, then $d(\mathbf{x},\mathbf{y}) = 2$. If $\mathbf{x}$ is transmitted and a single error occurs, then $\mathbf{y}$ can never be received. Table 3.2 gives the distances between all 4-bit codewords in which the first three bits carry information and the fourth is an even parity check bit. We can see that the minimum distance here is 2; hence, the code is suitable as a single error-correcting code.

Table 3.2. Distances between 4-bit codewords

	0000	0011	0101	0110	1001	1010	1100	1111
0000	0	2	2	2	2	2	2	4
0011	2	0	2	2	2	2	4	2
0101	2	2	0	2	2	4	2	2
0110	2	2	2	0	4	2	2	2
1001	2	2	2	4	0	2	2	2
1010	2	2	4	2	2	0	2	2
1100	2	4	2	2	2	2	0	2
1111	4	2	2	2	2	2	2	0

To determine exactly what the error-detecting and error-correcting capabilities for a code are, we need to analyze the minimum distance for the code. Let $\mathbf{x}$ and $\mathbf{y}$ be codewords. If $d(\mathbf{x},\mathbf{y}) = 1$ and an error occurs where $\mathbf{x}$ and $\mathbf{y}$ differ, then $\mathbf{x}$ is changed to $\mathbf{y}$. The received codeword is $\mathbf{y}$ and no error message is given. Now suppose $d(\mathbf{x},\mathbf{y}) = 2$. Then a single error cannot change $\mathbf{x}$ to $\mathbf{y}$. Therefore, if $d_{\min} = 2$, we have the ability to detect single errors. However, suppose that $d(\mathbf{x},\mathbf{y}) = 2$, $\mathbf{y}$ is sent, and a noncodeword $\mathbf{z}$ is received such that

$$d(\mathbf{x},\mathbf{z}) = d(\mathbf{y},\mathbf{z}) = 1.$$

Then the decoder cannot decide between $\mathbf{x}$ and $\mathbf{y}$. Even though we are aware that an error has occurred, we do not know what the error is. Suppose $d_{\min} \geq 3$. Then the maximum-likelihood decoding scheme corrects all single errors. Starting with a codeword $\mathbf{x}$, an error in the transmission of a single bit gives $\mathbf{y}$ with $d(\mathbf{x},\mathbf{y}) = 1$, but $d(\mathbf{z},\mathbf{y}) \geq 2$ for any other codeword $\mathbf{z} \neq \mathbf{x}$. If we do not require the correction of errors, then we can detect multiple errors when a code has a minimum distance that is greater than 3.

THEOREM 50. *Let C be a code with $d_{\min} = 2n+1$. Then C can correct any n or fewer errors. Furthermore, any $2n$ or fewer errors can be detected in C.*

EXAMPLE. In Table 3.3, the codewords $\mathbf{c}_1 = (00000)$, $\mathbf{c}_2 = (00111)$, $\mathbf{c}_3 = (11100)$, and $\mathbf{c}_4 = (11011)$ determine a single error-correcting code.

Table 3.3. Hamming distances for an error-correcting code

	00000	00111	11100	11011
00000	0	3	3	4
00111	3	0	4	3
11100	3	4	0	3
11011	4	3	3	0

Modern coding theory began in 1948 with C. Shannon's paper [**38**]. This paper offered an example of an algebraic code, and Shannon's Theorem proclaimed exactly how good codes could be expected to be. Coding theory has grown tremendously in the past several years. *The Theory of Error-Correcting Codes*, by MacWilliams and Sloane [**39**], already contained over 1500 references. Currently, very active research is being done with Goppa codes, which are heavily dependent on algebraic geometry.

For a more in depth treatment of the subject the references [**40, 41, 42, 43**] are appropriate.

3.3. Fundamentals of Number Theory

In this section we present some results from elementary Number Theory that will be useful from here on. For more in depth information on this subject the reader is referred to the references [**44, 45, 46**].

The Division Algorithm. An application of the Principle of Well-Ordering (discussed in subsection 2.4.4) that we will use often is the division algorithm.

THEOREM 51. *(Division Algorithm) Let a and b be integers, restricting $b > 0$. Then, there exist unique integers q and r such that*

$$a = bq + r$$

where $0 \leq r < b$.

Let a and b be integers. If $b = ak$ for some integer k, we write $a \mid b$(a divides b). An integer d is called a *common divisor* of a and b if $d \mid a$ and $d \mid b$. The *greatest common divisor* of integers a and b is a positive integer d such that d is a common divisor of a and b and if d' is any other common divisor of a and b, then $d' \mid d$. We write $d = \gcd(a,b)$; for example, $\gcd(24,36) = 12$ and $\gcd(120,102) = 6$. We say that two integers a and b are *relatively prime* if $\gcd(a,b) = 1$.

THEOREM 52. *Let a and b be nonzero integers. Then there exist integers r and s such that*

$$\gcd(a,b) = ar + bs.$$

Furthermore, the greatest common divisor of a and b is unique.

COROLLARY 53. *Let a and b be two integers that are relatively prime. Then there exist integers r and s such that $ar + bs = 1$.*

The Euclidean Algorithm. Among other things, Theorem 52 allows us to compute the greatest common divisor of two integers.

EXAMPLE. Let us compute the greatest common divisor of 945 and 2415. First observe that

$$\begin{aligned} 2415 &= 945 \cdot 2 + 525 \\ 945 &= 525 \cdot 1 + 420 \\ 525 &= 420 \cdot 1 + 105 \\ 420 &= 105 \cdot 4 + 0. \end{aligned}$$

Reversing our steps, 105 divides 420, 105 divides 525, 105 divides 945, and 105 divides 2415. Hence, 105 divides both 945 and 2415. If d were another common divisor of 945 and 2415, then d would also have to divide 105. Therefore, $\gcd(945, 2415) = 105$.

If we work backward through the above sequence of equations, we can also obtain numbers r and s such that $945r + 2415s = 105$. Observe that

$$\begin{aligned} 105 &= 525 + (-1) \cdot 420 \\ &= 525 + (-1) \cdot [945 + (-1) \cdot 525] \\ &= 2 \cdot 525 + (-1) \cdot 945 \\ &= 2 \cdot [2415 + (-2) \cdot 945] + (-1) \cdot 945 \\ &= 2 \cdot 2415 + (-5) \cdot 945. \end{aligned}$$

So $r = -5$ and $s = 2$. Notice that r and s are not unique, since $r = 41$ and $s = -16$ would also work.

To compute $\gcd(a,b) = d$, we are using repeated divisions to obtain a decreasing sequence of positive integers $r_1 > r_2 > \cdots > r_n = d$; that is,

$$\begin{aligned} b &= aq_1 + r_1 \\ a &= r_1 q_2 + r_2 \\ r_1 &= r_2 q_3 + r_3 \\ &\vdots \\ r_{n-2} &= r_{n-1} q_n + r_n \\ r_{n-1} &= r_n q_{n+1}. \end{aligned}$$

To find r and s such that $ar + bs = d$, we begin with this last equation and substitute results obtained from the previous equations:

$$\begin{aligned} d &= r_n \\ &= r_{n-2} - r_{n-1} q_n \\ &= r_{n-2} - q_n(r_{n-3} - q_{n-1} r_{n-2}) \\ &= -q_n r_{n-3} + (1 + q_n q_{n-1}) r_{n-2} \\ &\vdots \\ &= ra + sb. \end{aligned}$$

This algorithm that we used to find the greatest common divisor d of two integers a and b (also to write d as the linear combination of a and b) is known as the *Euclidean algorithm*.

Prime Numbers. Let p be an integer such that $p > 1$. We say that p is a *prime number*, or simply p is *prime*, if the only positive numbers that divide p are 1 and p itself. An integer $n > 1$ that is not prime is said to be *composite*.

LEMMA 54. *(Euclid) Let a and b be integers and p be a prime number. If $p \mid ab$, then either $p \mid a$ or $p \mid b$.*

THEOREM 55. *Let n be an integer such that $n > 1$. Then*

$$n = p_1 p_2 \cdots p_k,$$

where $p_1, \ldots, p_k$ are primes (not necessarily distinct). Furthermore, this factorization is unique; that is, if

$$n = q_1 q_2 \cdots q_l,$$

then $k = l$ and the q_i's are just the p_i's rearranged.

Definition of gcd and lcm. An important relation between two integers is their greatest common divisor and their least common multiple. We already mentioned that,

DEFINITION 56. the greatest common divisor of two integers m and n, not both equal to zero, denoted by gcd(m, n), is the largest positive integer that divides both of them.

As an example gcd(12, 15) = 3.

DEFINITION 57. The least common multiple of two nonzero integers m and n, denoted by lcm(m, n), is the smallest positive integer that is divisible by both of them.

As an example e.g. lcm(6, 4) = 12.

The symbol $m \mid n$ means that a natural number divides an integer with zero remainder, e.g. $3 \mid (-6)$ or $5 \mid 5$, otherwise we write $m \nmid n$. If $m \mid n$ and $m < n$, then m is called a proper divisor of n. If $1 < m < n$, then m is said to be a nontrivial divisor of n. We say that m^j exactly divides n, and write $m^j \parallel n$, if $m^j \mid n$, but $m^{j+1} \nmid n$.

Notion of congruence.

DEFINITION 58. Let a, b, m be given integers and $m \geq 1$. If $m \mid (a-b)$, i.e. $(a-b)$ is divisible by m, we write $a \equiv b \pmod{m}$ and say that a is congruent to b modulo m; b is called a residue of a modulo m.

It is clear that there are exactly m distinct in-congruent residues modulo m. The relation $\cdot \equiv \cdot \pmod{m}$ is transitive, reflexive and symmetric and therefore it is an equivalence relation over the set of integers.

Chinese Remainder Theorem.

THEOREM 59. *Let $m_1, m_2, \ldots, m_k$ be pairwise coprime natural numbers. Then for the system of simultaneous congruences $x \equiv r_i \pmod{m_i}$, with $i = 1, 2, \ldots k$, and integers r_i, there exists one and only one solution x modulo M, where $M = m_1 m_2 \cdots m_k$.*

Fermat's Little Theorem.

THEOREM 60. *Fermat's Little Theorem. If a is a natural number and p a prime number, then $p \mid a^p - a$.*

Another way of formulating this is by means of the congruence notation: If p is a prime and a is a natural number coprime to p, then $a^{p-1} \equiv 1 \pmod{p}$.

Fermat Primes. A natural number p is said to be a prime if $p > 1$ and p is divisible only be p and 1. The numbers $F_m = 2^{2^m} + 1$ for $m = 0, 1, \ldots$ are called Fermat numbers[3] and the first five members (F_0 to F_4) of the sequence are primes. Notice that each Fermat number F_m for $m > 0$ is of the form $\left(2^{2^{m-1}}\right)^2 + 1^2$.
Fermat numbers have some interesting properties. The formula $F_m = 6n - 1$ and the recurrence formulas $F_{m+1} = F_m + 2^{2^m} F_0 F_1 \cdots F_{m-1}$, $F_m = F_0 F_1 \cdots F_{m-1} + 2$, $F_m = (F_{m-1} - 1)^2 + 1$, hold for all $m \geq 1$. The formula $F_m = F_{m-1}^2 - 2(F_{m-2} - 1)^2$ holds for all $m \geq 2$. This is a special case of the following proposition. Each Fermat number F_m, where $m \geq 2$, has infinitely many representations of the form $x^2 - 2y^2$, with x and y positive integers. Every Fermat number F_m in the binary system has the form $100\ldots001$ with $2^m - 1$ zeros inside. For more information check [**47**].

THEOREM 61. *For $m \geq 1$ the Fermat number F_m is prime if and only if $3^{(F_m-1)/2} \equiv -1$ (mod F_m).*

Euler totient function.

DEFINITION 62. For every $n \in \mathbb{N}$ the value $\phi(n)$ is defined as the number of all natural numbers not greater than n that are coprime to n, i.e.,

$$\phi(n) = |\{m \in \mathbb{N} : 1 \leq m \leq n, \gcd(m, n) = 1\}|$$

where $|\cdot|$ denotes the number of elements. If p is prime then $\phi(p) = p - 1$ and therefore $\phi(F_m) = 2^{2^m}$.

Another interesting property of the Euler totient function $\phi(n)$ can be expressed as follows:

$$\gcd(m, n) = 1 \Rightarrow \phi(mn) = \phi(m)\phi(n)$$

We also recall Gauss's well-known formula

$$\sum_{d|N} \phi(d) = N$$

This next theorem by Euler is a direct generalization of Fermat's Little Theorem.

THEOREM 63. *Let a, $n \in \mathbb{N}$. Then $a^{\phi(n)} \equiv 1$ (mod n) if and only if gcd(a, n) = 1.*

Primitive roots.

[3]Pierre Fermat (1601?–1665) conjectured that $2^{2^n} + 1$ was prime for all n. However, it was shown by Leonhard Euler (1707–1783) that $2^{2^5} + 1 = 4{,}294{,}967{,}297$ is a composite number.

DEFINITION 64. Let p be a prime. By theorem 60, the maximum order modulo p of any integer a coprime to p is $p-1$. We call the integer $g \not\equiv 0 \pmod{p}$ a primitive root modulo p if $\mathrm{ord}_p g = p-1$. Alternatively, if $\gcd(a,n) = 1$ and a is of order $\phi(n)$ modulo n, then a is a primitive root of the integer n.

For example, 3 is a primitive root modulo 17, since $3^{16} \equiv 1 \pmod{17}$ and $3^k \not\equiv 1 \pmod{17}$ for all $k = 1, \ldots, 15$. In this case the multiplicative order of 3 modulo 17 is 16, i.e., $\mathrm{ord}_{17} 3 = 16$.

Multiplicative order.

DEFINITION 65. Let n and a be positive integers such that gcd(n, a) = 1. The smallest positive exponent e for which $n \mid a^e - 1$, i.e., $a^e \equiv 1 \pmod{n}$, is called the multiplicative order[4] of the number a modulo n, which we shall write as $e = \mathrm{ord}_n a$.

Note that $\mathrm{ord}_{F_m} 2 = 2^{m+1}$. A lemma of theorem 60 states that if $e = \mathrm{ord}_n a$, then $n \mid a^{k \cdot e} - 1$ for $k \in \{1, 2, \ldots\}$.

THEOREM 66. *If the integer a has order e modulo n and $k > 0$, then a^k has order $k/gcd(k,e)$ modulo n.*

As a corollary of the previous theorem we can write:

COROLLARY 67. *Let a have order e modulo n. Then a^k also has order e if and only if $gcd(k,e) = 1$.*

DEFINITION 68. The *modular (multiplicative) inverse* of an integer $b \pmod{n}$ is the integer b^{-1} such that $b \cdot b^{-1} \equiv 1 \pmod{n}$.

Every nonzero integer b has an inverse (modulo p) for p a prime and b not a multiple of p. If n is not a prime then a nonzero integer b has an inverse iff b and n are relatively prime, i.e. iff $\gcd(b,n) = 1$.

If b and n are relatively prime, then theorem 63 states that $b^{\phi(n)} \equiv 1 \pmod{n}$, where $\phi(n)$ is the totient function. Hence,

$$b^{-1} \equiv b^{\phi(n)-1} \pmod{n} \tag{3.1}$$

The next theorem determines all integers $n \geq 2$ that have primitive roots.

THEOREM 69. *Let $n \geq 2$. As referred in Theorem 26 there exists a primitive root modulo n if and only if $n \in \{2, 4, p^k, 2p^k\}$, where p is an odd prime and $k \geq 1$. Moreover, if n has a primitive root, then n has exactly $\phi(\phi(n))$ in-congruent primitive roots. Note that if p is prime , then p has exactly $\phi(p-1)$ in-congruent primitive roots.*

[4] in older terminology: the exponent to which a belongs modulo n.

Table 3.4. Residues modulo 17

$p = F_2 = 17$																	
a	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\text{ord}_p a = p-1$	✗	✗	✗	✓	✗	✓	✓	✓	✗	✗	✓	✓	✓	✗	✓	✗	✗
$\left(\frac{a}{p}\right)$	0	1	1	-1	1	-1	-1	-1	1	1	-1	-1	-1	1	-1	1	1

Legendre symbol. Let $n \geq 2$ and a be integers such that $\gcd(a, n) = 1$. If the quadratic congruence $x^2 \equiv a \pmod{n}$ has a solution x, then a is called a quadratic residue modulo n. Otherwise, a is called a non-quadratic residue modulo n.

Let p be an odd prime. Then the Legendre symbol $\left(\frac{a}{p}\right)$ is defined to be 1 if a is a quadratic residue modulo p, -1 if a is a non-quadratic residue modulo p, and 0 if $p \mid a$. If p is an odd prime, then the number of quadratic residues is equal to the number of non-quadratic residues, which is equal to $(p-1)/2$. In the case of Fermat primes the values a for which the Legendre symbol is $\left(\frac{a}{p}\right) = -1$ are exactly the primitive roots modulo p. This is illustrated Table 3.4 for $p = 17$.

We note that if g is a primitive root modulo p, where p is prime, then g is a generator of the set of all nonzero residues modulo p, that is, $\left\{g, g^2, g^3, \cdots, g^{p-1}\right\}$ consists of all the $p-1$ nonzero residues modulo p. Hence, for any integer $a \not\equiv 0 \pmod{p}$, there exists an exponent $n \in \{1, \ldots, p-1\}$ such that $g^n \equiv a \pmod{p}$.

Let the set QR be the set of quadratic residues. The set QR is a subgroup of $\mathbb{Z}_p^*$ of index 2. Let η be a non-quadratic residue. The coset $\text{NQR} = \eta\text{QR} = \{\eta j;\ j \in \text{QR}\}$ collects all the non-quadratic residues. It is not restrictive to assume that η is the smallest non-quadratic residue modulo p that generates the multiplicative group $\mathbb{Z}_p^*$.

The following criterion, due to Euler, gives a method for evaluating the Legendre symbol $\left(\frac{a}{p}\right)$.

THEOREM 70. *Let p be an odd prime. Then $a^{(p-1)/2} \equiv (\frac{a}{p})$ (mod p).*

One particular property of the Legendre symbol is $\left(\frac{2}{p}\right) = (-1)^{(p^2-1)/8}$. Restricting p to be a Fermat prime we can say that 2 is a primitive root only for $p = 3$ and $p = 5$. The Jacobi symbol $\left(\frac{a}{n}\right) = \prod_{i=1}^{r} \left(\frac{a}{p_i}\right)$ where $n = p_1 p_2 \cdots p_r$ (for odd primes p_i's not necessarily distinct) is a generalization of the Legendre symbol. Note that $\left(\frac{3}{F_m}\right) = -1$, therefore 3 is a primitive root for all known Fermat primes F_m, $m \geq 1$, i.e. for $p \in \{5, 17, 257, 65537\}$.

THEOREM 71. *The set of all non-quadratic residues of a Fermat prime is equal to the set of all its primitive roots.*

PROOF. Let n be a non-quadratic residue of the Fermat prime F_m. Then, by Euler's Criterion,

$$n^{(F_m-1)/2} = n^{2^{2^m-1}} \equiv -1 \pmod{F_m}$$

However, according to Fermat's Little Theorem,

$$\text{ord}_{F_m} n \mid F_m - 1 = 2^{2^m}$$

Suppose that

$$\text{ord}_{F_m} n = 2^k,\ 0 \leq k \leq 2^m$$

Then,

$$\text{ord}_{F_m} n \mid (F_m - 1)/2$$

Hence,

$$n^{(F_m-1)/2} \equiv 1 \pmod{F_m}$$

contradicting our initial congruence $n^{(F_m-1)/2} \equiv -1$. Therefore, $\text{ord}_{F_m} n = F_m - 1$, and n is a primitive root modulo F_m. Conversely, again by Euler's criterion, a quadratic residue r modulo a prime p cannot be a primitive root modulo p, since $r^{(p-1)/2} \equiv 1 \pmod{p}$. □

Regular polygons. Carl Friedrich Gauss (1777-1855) quite unexpectedly found through investigation of the roots of the equation $z^n = 1$ a theorem that expresses an interesting connection between the Euclidean construction of regular polygons and the Fermat primes. He showed that the regular polygon can be constructed by ruler (straightedge) and compass if the number of its sides is equal to 3, 4, 5, 6, 8, 10, 12, 15, 16, 17, etc. More precisely: There exists a construction of the regular polygon with n sides by ruler and compass if and only if $n = 2^i F_{m_1} F_{m_2} \cdots F_{m_j}$, where $n \geq 3$, $i \geq 0$, $j \geq 0$, and $F_{m_1}, F_{m_2}, \cdots F_{m_j}$ are distinct Fermat primes.

Fermat Number Transform. With F_m prime, let $\alpha \in \{2, 3, \ldots, F_m - 1\}$ be given and N be chosen such that $N \mid \text{ord}_{F_m} \alpha$. The number N is called a transformation length, and $\text{ord}_{F_m} \alpha$ the maximum transformation length. For instance, if $m \geq 1$ and $\alpha = 3$, then the maximum transformation length is 2^{2^m} due to Pepin's test (theorem 61). If $\alpha = 2$, then clearly $\text{ord}_{F_m} \alpha = 2^{m+1}$ for $m = 0, 1, \ldots$.

Given the vector $x = (x[0], x[1], \cdots, x[N-1])^{\mathrm{T}}$ of integers such that $x[k] \in \{0, 1, \ldots, F_m - 1\}$ for $k = 0, 1, \ldots, N-1$, the one-dimensional Fermat Number Transform (FNT) and its inverse are defined by

$$X[j] \equiv \sum_{k=0}^{N-1} x[k]\,\alpha^{jk} \,(\mathrm{mod}\, F_m),\ j = 0,1,\ldots,N-1$$

$$x[k] \equiv N^{-1} \sum_{j=0}^{N-1} X[j]\,\alpha^{-jk} \,(\mathrm{mod}\, F_m),\ k = 0,1,\ldots,N-1$$

respectively, where $X[j] \in \{0,1,\ldots,F_m-1\}$ for all $j \in \{0,1,\ldots,N-1\}$ and N^{-1} denotes the integer such that $NN^{-1} \equiv 1 \pmod{F_m}$.

Discrete convolution. Define the finite discrete convolution

$$y = h * x$$

of two vectors $h = (h[0], h[1], \ldots, h[N-1])^{\mathrm{T}}$ and $x = (x[0], x[1], \ldots, x[N-1])^{\mathrm{T}}$ by the relation

$$y[k] = \sum_{j=0}^{N-1} h[k-j]\,x[j],\ k = 0,1,\ldots,N-1$$

where $h[k-j] = h[k-j+N]$ if $k < j$ (i.e. the argument is evaluated modulo N). We could also assume that the sequences are periodically extended with the period N, and then we speak about cyclic convolutions.

In general we assume the complex valued functions $f[n]$ and $g[n]$ to be defined for $n = 0,1,\ldots,N-1$, i.e. the argument n is computed $(\mathrm{mod}\, N)$. Then, the convolution, correlation and the Discrete Fourier Transform (DFT) are defined as:

$$\text{Convolution:} \quad (f * g)[n] \doteq \sum_{m=0}^{N-1} f[m]\,g[n-m] \tag{3.2}$$

$$\text{Correlation:} \quad (f \star g)[n] \doteq \sum_{m=0}^{N-1} \overline{f[m]}\,g[m+n] \tag{3.3}$$

$$\text{DFT:} \quad F[k] \doteq \frac{1}{N} \sum_{n=0}^{N-1} f[n]\,\mathrm{e}^{-2\pi ikn/N} \tag{3.4}$$

with $i = \sqrt{-1}$, and $k = 0,1,\ldots,N-1$. Table 3.5 lists some common transform rules.

The FNT shares many properties with the DFT, in particular, the following convolution identity

$$y = h * x = \mathrm{FNT}^{-1}\{\mathrm{FNT}\{h\} \cdot \mathrm{FNT}\{x\}\}$$

where $(\cdot)$ denotes the pointwise product

Table 3.5. DFT transform rules

Reflection	$f[-n] \xrightarrow{\mathscr{F}} F[-k]$
Conjugation	$\overline{f[n]} \xrightarrow{\mathscr{F}} \overline{F[-k]}$
Translation	$f[n-n_0] \xrightarrow{\mathscr{F}} \mathrm{e}^{-2\pi i k n_0/N} F[k]$
Modulation	$\mathrm{e}^{2\pi i k_0 n/N} f[n] \xrightarrow{\mathscr{F}} F[k-k_0]$
Convolution	$(f_1 * f_2)[n] \xrightarrow{\mathscr{F}} N F_1[k] F_2[k]$
Correlation	$(f_1 \star f_2)[n] \xrightarrow{\mathscr{F}} N \overline{F_1[k]} F_2[k]$
Inversion	$F[n] \xrightarrow{\mathscr{F}} \frac{1}{N} f[-k]$
Dilation	$f[mn] \xrightarrow{\mathscr{F}} \begin{cases} F[m'k],\ mm' \equiv 1 \pmod N & \text{if } (m,N)=1 \\ \begin{cases} \sum_{\ell=0}^{m-1} F[k/m - \ell N/m] & \text{if } m \mid k \\ 0 & \text{otherwise} \end{cases} & \text{if } m \mid N \end{cases}$

$$H \cdot X = (H[0]X[0], H[1]X[1], \ldots, H[N-1]X[N-1])^{\mathrm{T}}$$

and where capital letters denote the transformed sequences. Therefore, the convolution $*$ of two periodic signals of length N can be performed by taking the inverse transform of the pointwise product of the transformed signals.

Sometimes, it is advantageous to adopt the following balanced representation of the signal x,

$$x[k] \in \left\{ -\frac{F_m - 1}{2}, \ldots, -2, -1, 0, 1, 2, \ldots, \frac{F_m - 1}{2} \right\}$$

With F_m prime, let $\alpha \in \{2, 3, \ldots, F_m - 1\}$ be given and N be chosen such that $N \mid \mathrm{ord}_{F_m} \alpha$. If $T \equiv \left(\alpha^{jk}\right)_{j,k=0}^{N-1} \pmod{F_m}$ and $T^{-1} \equiv N^{-1} \left(\alpha^{-jk}\right)_{j,k=0}^{N-1} \pmod{F_m}$ are the matrices corresponding to the transformations FNT and FNT^{-1}, then $TT^{-1} \equiv T^{-1}T \equiv I \pmod{F_m}$, where I is the identity matrix.

3.4. Some common sequences

3.4.1. Preliminaries. Let $\mathbb{Z}_M$ be the ring of integers modulo M and $\omega = e^{j2\pi/M}$ where $j = \sqrt{-1}$. A sequence $\{s(t)\}$ is called a M-ary sequence if $s(t) \in \mathbb{Z}_M \, \forall t$, and the sequence $\left\{ \omega^{s(t)} \right\}$ is referred to as a polyphase sequence corresponding to $\{s(t)\}$. Let $\mathscr{F} = \{\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_{L-1}\}$ be a set of L cyclically distinct M-ary sequences with period N, where $\mathbf{s}_l = \{s_l(t)\}$, $0 \le l < L$.

Multiplicative characters. For a primitive element α of $\mathbb{F}_q$ with $q = p^m$, let $\psi(x)$ be a multiplicative character of $\mathbb{F}_q$ such that

$$\psi\left(\alpha^t\right) = \exp\left(j\frac{2\pi l}{q-1}t\right),\ 0 \le t \le q-2 \tag{3.5}$$

for an integer l and $\psi(0) = 0$. When $l = 0$, the multiplicative character $\psi(x)$ is called trivial since $\psi(x) = 1$ for any $x \in \mathbb{F}_q^*$. For a divisor M of $q-1$ and any integer l with $\gcd(l, q-1) = (q-1)/M$, $\psi(x)$ is called a multiplicative character of $\mathbb{F}_q$ of order M.

Cyclotomic numbers. Let $q = Mf + 1$. For a primitive element α of $\mathbb{F}_q$, the set of nonzero elements in $\mathbb{F}_q$ is $\mathbb{F}_q^*$ which can be decomposed into M disjoint subsets

$$D_k = \left\{\alpha^{Ml+k} \mid 0 \le l \le f-1\right\},\ k = 0, 1, \ldots, M-1 \tag{3.6}$$

which are called the cyclotomic classes of $\mathbb{F}_q$ of order M. For $0 \le i, j \le M-1$, the number defined by

$$(i, j)_M \doteq \mid (D_i + 1) \cap D_j \mid \tag{3.7}$$

is called the cyclotomic number of $\mathbb{F}_q$ of order M.

An integer R is said to belong to the cyclotomic class (or index class) D_k with respect to α if there exists an integer l such that $R \equiv \alpha^{Ml+k} \pmod q$. Thus the index class k consists of f distinct numbers α^k, α^{M+k}, ... , $\alpha^{M(f-1)+k}$ modulo q. The cyclotomic number $(i, j)_M$ counts the number of times $R+1$ belongs to the cyclotomic class j when R belongs to the cyclotomic class i. That is, $(i, j)_M$ is the number of solutions x, y of the congruence $\alpha^{Mx+i} + 1 \equiv \alpha^{My+j} \pmod q$, where the integers x, y are chosen from 0, 1, $\cdots$, $f-1$. This congruence shows that there are at most M^2 distinct cyclotomic numbers of order M and that these numbers depend not only on q, M, i, j but also on which of the $\phi(q-1)$ primitive roots α of q is chosen.

Cyclotomic numbers have some properties:

$$(i, j)_M = \left(i', j'\right)_M,\ i \equiv i',\ j \equiv j' \pmod M \tag{3.8}$$

$$(i, j)_M = (M-i, j-i)_M = \begin{cases} (i, j)_M & f \text{ even} \\ \left(i + \frac{M}{2}, j + \frac{M}{2}\right) & f \text{ odd} \end{cases} \tag{3.9}$$

$$\sum_{j=0}^{M-1} (i, j)_M = f - \begin{cases} 1 & i \equiv 0 \pmod M,\ f \text{ even} \\ 1 & i \equiv \frac{M}{2} \pmod M,\ f \text{ odd} \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

$$(i, j)_{M'} = (si, sj)_M, \quad \alpha' = \alpha^s \tag{3.11}$$

For $(i,j)_{M'}$ based on the primitive root $\alpha' = \alpha^s \pmod q$ we have $(i,j)_{M'} = (si, sj)_M$ necessarily with s prime to $q-1$.

The sum (3.10) is simply the number of successors of members of cyclotomic class D_k which belong to any cyclotomic class at all.

3.4.2. Sidel'nikov sequences. Let p be a prime, n a positive integer and α a primitive element in the finite field $\mathbb{F}_{p^n}$ with p^n elements. Let $M \mid p^n - 1$. Let S_k, k=0,1,..., $M-1$, be the disjoint subsets of $\mathbb{F}_{p^n}$ defined as

$$S_k = \left\{ \alpha^{Mi+k} - 1 \mid 0 \le i < \frac{p^n - 1}{M} \right\} \tag{3.12}$$

The M-ary Sidel'nikov sequence $s(t)$ of period $p^n - 1$ is defined as

$$s(t) = \begin{cases} k, & \text{if } \alpha^t \in S_k,\ 0 \le k \le M-1 \\ k_0, & \text{if } t = \frac{p^n-1}{2} \end{cases} \tag{3.13}$$

where k_0 is some integer modulo M. Note that $\alpha^{\frac{p^n-1}{2}} = -1$, $\cup_{k=0}^{M-1} S_k = \mathbb{F}_{p^n} \setminus \{-1\}$, and $0 \in S_0$.

3.4.3. M-ary Power Residue sequences. Let p be an odd prime and M a divisor of $p-1$, i.e. $M \mid p-1$. Let μ be a primitive root modulo p. The M-ary power residue sequence of period p is defined as

$$r(t) = \begin{cases} 0, & \text{if } t \equiv 0 \pmod p \\ k, & \text{if } t \in D_k \end{cases} \tag{3.14}$$

where D_k, $0 \le k \le M-1$, is the cyclotomic class of $\mathbb{F}_p$ of order M defined as

$$D_k = \left\{ \mu^{Ml+k} \mid 0 \le l \le \frac{p-1}{M} - 1 \right\} \tag{3.15}$$

Using a multiplicative character ψ of of $\mathbb{F}_p$ of order M, the polyphase sequence corresponding to $\mathbf{r}$ can be expressed as

$$\omega^{r(t)} = \psi(t) + I(t) \tag{3.16}$$

where I is the indicator function such that $I(0) = 1$ and $I(t) = 0$, for $t \neq 0$.

3.4.4. Lempel/Cohn/Eastman sequences. Let G denote the multiplicative group of $\mathbb{F}_{p^n}$. Since α is a primitive element of $\mathbb{F}_{p^n}$ it is also a generator of the cyclic group G. Consider the subset S of G defined by

$$S = \left\{ \alpha^{2i+1} - 1 \mid 0 \le i \le K-1 \right\} \tag{3.17}$$

where

$$K = \frac{1}{2}(p^n - 1) \tag{3.18}$$

Note that S contains exactly one half of the elements of G and that every element of S is equal to some power of α. Let f denote the function from G onto $\{1,-1\}$ defined by

$$f\left(\alpha^t\right) = \begin{cases} 1, & \text{if } \alpha^t \in S \\ -1, & \text{if } \alpha^t \notin S \end{cases} \quad 0 \le t < 2K \tag{3.19}$$

3.4.5. TCH sequences. In section 3.6 we will present the original generation method for obtaining TCH sequences. However to complete the current section we give the formulation in the same context as the other sequences.

Let p be restricted to a Fermat prime. Consider the subset S of G defined by

$$S = \left\{ 1 + \alpha^{2i+1} \mid 0 \le i < \frac{p-1}{2} \right\} \tag{3.20}$$

The binary TCH sequence $f(t)$ of period $p-1$ is defined as

$$f(t) = \begin{cases} 1, & \text{if } \alpha^t \in S \\ 0, & \text{if } \alpha^t \notin S \end{cases} \quad 0 \le t < p-1 \tag{3.21}$$

3.5. Rings and Fields

In the previous chapter we discussed sets with a single binary operation satisfying certain axioms, but often we are more interested in working with sets that have two binary operations. For example, one of the most natural algebraic structures to study is the integers with the operations of addition and multiplication. These operations are related to one another by the distributive property. If we consider a set with two such related binary operations satisfying certain axioms, we have an algebraic structure called a ring. In a ring we add and multiply such elements as real numbers, complex numbers, matrices, and functions.

DEFINITION 72. A nonempty set R is a *ring* if it has two closed binary operations, addition and multiplication, satisfying the following conditions.

(1) $a+b=b+a$ for $a,b \in R$.
(2) $(a+b)+c=a+(b+c)$ for $a,b,c \in R$.
(3) There is an element 0 in R such that $a+0=a$ for all $a \in R$.
(4) For every element $a \in R$, there exists an element $-a$ in R such that $a+(-a)=0$.
(5) $(ab)c=a(bc)$ for $a,b,c \in R$.
(6) For $a,b,c \in R$,

$$\begin{aligned} a(b+c) &= ab+ac \\ (a+b)c &= ac+bc. \end{aligned}$$

This last condition, the distributive axiom, relates the binary operations of addition and multiplication. Notice that the first four axioms simply require that a ring be an abelian group under addition, so we could also have defined a ring to be an abelian group $(R,+)$ together with a second binary operation satisfying the fifth and sixth conditions given above.

If there is an element $1 \in R$ such that $1 \neq 0$ and $1a = a1 = a$ for each element $a \in R$, we say that R is a ring with *unity* or *identity*. A ring R for which $ab = ba$ for all a, b in R is called a *commutative ring*. A commutative ring R with identity is called an *integral domain* if, for every $a, b \in R$ such that $ab = 0$, either $a = 0$ or $b = 0$, i.e. if it has no zero divisors, a commutative ring with identity is an *integral domain*. A *division ring* is a ring R, with an identity, in which every nonzero element in R is a *unit*; that is, for each $a \in R$ with $a \neq 0$, there exists a unique element a^{-1} such that $a^{-1}a = aa^{-1} = 1$.

DEFINITION 73. A commutative division ring is called a *field*.

The relationship among rings, integral domains, division rings, and fields is shown in Figure 3.3.

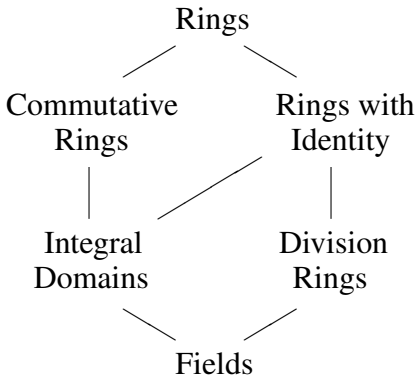


Figure 3.3. Types of rings

If R is a ring and r is a nonzero element in R, then r is said to be a *zero divisor* if there is some nonzero element $s \in R$ such that $rs = 0$. If an element a in a ring R with identity has a multiplicative inverse, we say that a is a *unit*. If every nonzero element in a ring R is a unit, then R is called a *division ring*. Again, a commutative division ring is called a *field*.

PROPOSITION 74. *Let R be a ring and S a subset of R. Then S is a subring of R if and only if the following conditions are satisfied.*

(1) $S \neq \emptyset$.
(2) $rs \in S$ *for all* $r, s \in S$.
(3) $r - s \in S$ *for all* $r, s \in S$.

In the study of groups, a homomorphism is a map that preserves the operation of the group. Similarly, a homomorphism between rings preserves the operations of addition and multiplication in the ring. More specifically, if R and S are rings, then a *ring homomorphism* is a map $\varphi : R \to S$ satisfying

$$\begin{aligned} \varphi(a+b) &= \varphi(a)+\varphi(b) \\ \varphi(ab) &= \varphi(a)\varphi(b) \end{aligned}$$

for all $a, b \in R$. If $\varphi : R \to S$ is a one-to-one and onto homomorphism, then φ is called an *isomorphism* of rings.

The set of elements that a ring homomorphism maps to 0 plays a fundamental role in the theory of rings. For any ring homomorphism $\varphi : R \to S$, we define the *kernel* of a ring homomorphism to be the set

$$\ker \varphi = \{r \in R : \varphi(r) = 0\}.$$

In ring theory there are a special class of subrings called ideals.

DEFINITION 75. An *ideal* in a ring R is a subring I of R such that if a is in I and r is in R, then both ar and ra are in I; that is, $rI \subset I$ and $Ir \subset I$ for all $r \in R$.

Let $[a]$ (alternatively $a+(n)$) denote the coset or residue class of the integer a modulo the positive integer n, where (n) denotes the principal ideal generated by n. The elements of the residue class ring $\mathbb{Z}/(n)$ are

$$[0] = 0+(n),\ [1] = 1+(n), \ldots, [n-1] = n-1+(n)$$

Let $\mathbb{Z}/(p)$ be the ring of residue classes of the integers modulo the principal ideal generated by a prime p. Let $\mathbb{F}_p$ be the set $\{0, 1, \ldots, p-1\}$ of integers and let $\varphi : \mathbb{Z}/(p) \to \mathbb{F}_p$ be the mapping defined by $\varphi([a]) = a$ for $a \in \mathbb{F}_p$. Then, $\mathbb{F}_p$ endowed with the field structure induced by φ, is a finite field, called the *Galois field* of order p and represented by $\mathrm{GF}(p)$.

3.6. Introduction to TCH sequences

With a few corrections this subsection summary closely follows the work presented in [**6**]. The motivation behind the development of TCH sequences was to obtain a sort of pseudo-noise sequences with good correlation properties and length equal to a power of two, i.e., 2^m.

Codewords can be represented by elements of a finite (Galois) field. One way of representing elements of $\mathrm{GF}(q^n)$ is by using a prime polynomial of degree n over $\mathrm{GF}(q)$. The elements of this field can be given by integers, by n-tuples, by polynomials of degree $(n-1)$ or by an exponential representation.

For TCH cyclic codes we want to exclude the 0 element represented by a n-tuple with all zeros since any cyclic shift yields the same codeword. Since the all-zero codeword is excluded TCH are non-linear codes.

With $q = p^m$ there exists a field $\mathrm{GF}(p)$ with p elements. These elements are represented with polynomials of degree $N-1$ with $N = 2^m$ being the codeword length. If, on the other hand, we also exclude the 0 element, the number of elements in $\mathrm{GF}(p)$ is $N = p-1$ and these two conditions imply,

$$p = N+1 = 2^m + 1 \tag{3.22}$$

resulting in the candidate primes p being Fermat primes, i.e. $F_i = 2^{2^i} + 1$ for which there are 5 known thus far, $F_0 = 3$, $F_1 = 5, F_2 = 17, F_3 = 257$, and $F_4 = 65537$.

For binary codewords we have to consider $\mathrm{GF}(2^m)$ with characteristic 2. A TCH sequence or codeword can thus be described by

$$p(x) = \sum_{i=0}^{N-1} a_i \cdot x^{K_i} \tag{3.23}$$

where a_i is either 0 or 1 and the placement of those zeros and ones in the codeword are given by the exponents K_i. Since we have N exponents we can associate them with the N elements of $\mathrm{GF}(p) \setminus \{0\}$.

Now we impose two restrictions. The first regards the balance of the codeword. A codeword with the same number of zeros and ones has better spectra from a transmission point of view. Moreover, if for any time shift the weight is $\frac{N}{2}$ the correlation between two of those balanced codewords is 0. With this restriction in place we can write,

$$p(x) = \sum_{i=0}^{\frac{p-1}{2}-1} x^{K_i} \tag{3.24}$$

Now we must select $\frac{N}{2}$ elements from the N elements of $\mathrm{GF}(p) \setminus \{0\}$. If we start with a primitive element α then all the elements can be described as α^i with $i = 0, 1, \ldots, p-2$. Splitting this list in half by considering even or odd powers of α we obtain two lists: α^{2i} and α^{2i+1} with $i = 0, 1, \ldots, \frac{p-3}{2}$. We will return later to the question as to which of these two lists is more appropriate.

The second restriction is a property similar to the shift-and-add properties of maximum length sequences (m-sequences), stating that the modulo-2 sum of a m-sequence and any phase shift of the same sequence is another phase of the same sequence. We use instead a different version of this property stating that the modulo-2 sum of a m-sequence and any phase shift of the same sequence is another sequence which can exhibit a vary low cross-correlation with the original one.

Table 3.6. Verifying $\alpha = 3$ as a primitive element of GF(17).

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
3^i	1	3	9	10	13	5	15	11	16	14	8	7	4	12	2	6

The sequence resulting from the shift-and-add process must have weight close or equal to $\frac{N}{2}$. Therefore we consider a cyclic shift of k positions of $p(x)$ as

$$p'(x) = x^k \cdot p(x) \tag{3.25}$$

and adding this shifted version with itself results in

$$p''(x) = p(x) + p'(x) = p(x)\left(1 + x^k\right) \tag{3.26}$$

Now, $p''(x)$ cannot be a shifted version of $p(x)$ so we must write

$$p''(x) = x^K \cdot p(x) \tag{3.27}$$

where capital K is a variable and x^K represents a vector with $\frac{N}{2}$ different elements previously designated as K_i. Thus,

$$\alpha^{K_i} = 1 + \alpha^j \tag{3.28}$$

and now we return to the question of which list α^{2i} or α^{2i+1} is more useful in representing the $\frac{N}{2}$ elements α^j.

Since $\alpha^j \in [1, 2, \cdots, p-1]$, adding 1 results in $1 + \alpha^j \in [2, 3, \cdots, p-1] \cup \{0\}$. We observe that the addition of the element 1 caused the cancellation of that element and the appearance of the element 0 which is undesirable. Computing modulo p, the element 0 is obtained by $1 + \alpha^j = 1 + (p-1)$ and since p is odd α^j must come from the list of even powers, i.e., α^{2i}. In consequence, we must choose the other list, i.e. that of odd powers, resulting in

$$\alpha^{K_i} = 1 + \alpha^{2i+1},\ i = 0, 1, \ldots, \frac{p-1}{2} - 1 \tag{3.29}$$

This equation is responsible for the generation of the so called basic TCH polynomial or codeword. Following the workflow proposed in **[6]** we present an example of the generation of a basic polynomial.

We will consider the case $p = 17$, for generating a basic polynomial of length 16. The first thing to do is to find among the values of the set $[2, \ldots, 16]$ a primitive element α of GF(17). Starting with $\alpha = 2$ and computing its powers α^i we find that $2^8 = 256 \equiv 1 \pmod{17}$ and therefore 2 is not a primitive element. Then, we select $\alpha = 3$ and table 3.6 presents its powers verifying that it generates all 16 elements without repetition.

The next step is using (3.29) to determine the values of K_i. This is presented in table 3.7.

Table 3.7. Determination of K_i for $(p = 17, \alpha = 3)$.

i	$2i+1$	$1+3^{2i+1}$	K_i
0	1	4	12
1	3	11	7
2	5	6	15
3	7	12	13
4	9	15	6
5	11	8	10
6	13	13	4
7	15	7	11

Using (3.24) the polynomial representation of the basic codeword associated with the primitive element $\alpha = 3$ is,

$$p_3(x) = x^4 + x^6 + x^7 + x^{10} + x^{11} + x^{12} + x^{13} + x^{15} \tag{3.30}$$

that can also be written as a binary word 1011 1100 1101 0000 or alternatively as BCD0 in hexadecimal format. To determine the other basic polynomials of length 16 we would repeat this process by first finding the primitive element and then solving the congruence (3.29).

Having presented the generation of a basic polynomial we now address the search for other polynomials either with equal length or with length different from the restricted set given by $p-1$ where p is a Fermat prime. To that end there are a number of ad-hoc methods proposed in [**6**]. The first and most straightforward one is the method called *Shift-and-Add*. As the name implies this method consists of circular shifting the codeword (i.e. rotating it) and then adding it with the original one. Assume that we rotate $p_3(x)$ one bit to the left. Then we obtain,

$$p_3^{'}(x) = 1 + x^5 + x^7 + x^8 + x^{11} + x^{12} + x^{13} + x^{14} \tag{3.31}$$

and adding $p_3(x)$ with $p_3^{'}(x)$ results in,

$$p_3^{''}(x) = 1 + x^4 + x^5 + x^6 + x^8 + x^{10} + x^{14} + x^{15} \tag{3.32}$$

or C571 in hexadecimal format. Note that this new codeword has the same weight of the original one. This procedure can be repeated for all the other bit shifts and the resulting codewords are then evaluated in terms of the their auto-correlation and their cross-correlation with different codewords. By specifying a target in terms of minimum distance it is possible to determine a code with a larger number of codewords. Instead of using just the original codeword BCD0 we can determine, from the process above, 3 other codewords (4F92, 26C7, D48E) with weight equal to 8 and minimum distance equal to 4 forming a code (16,7).

Several other methods have been proposed to obtain codewords with lengths equal to a power of two but not in the set $[16, 256, 65536]$. The methods are named random search, FFT,

Square-Wave, Decimation, etc. In general, the methods take one basic polynomial or a set of polynomials and algorithmically manipulate some properties to obtain new codewords using constraints like the codeword weight, cross-correlation values, or minimum distance. In any case, these methods follow a search and test procedure that is somewhat *ad-hoc* and, as also happens with the generating equation 3.29, only address binary codewords. For the interested reader, section 4.2.9 of [**6**] presents a comparative evaluation of these methods.

3.7. Conclusion

In this chapter we first provided the context and introduced some definitions of coding theory. Then we addressed some core results from number theory since they are essential for understanding the theoretical basis of certain codes. We also discussed codes as sequences since sometimes we are not interested in the error-correcting capability of the codewords but are more focused on their random-like nature when they are used in spreading applications. We presented some time-domain signal-processing operations on sequences and their relation with the frequency-domain. Finally we introduced a class of cyclic codes, named TCH, that provided the motivation for the work presented in this thesis.

CHAPTER 4

GENERALIZED TCH VIA ZECH LOGS

4.1. Introduction

In the previous chapter we saw that the generation of TCH basic codewords is associated to the primitive elements (see section 3.3) of the Galois field GF(p), p being a Fermat prime. This implies that the length N (or period) of these codewords is limited to the set $[4, 16, 256, 65536]$. In this chapter, based on the properties of Zech logarithms we develop an alternative generation procedure that is valid for all values of $N = 2^k$, k being a positive integer. We call generalized TCH to the generated codewords. This chapter begins with an introduction to discrete logarithms and their application to finite fields in a form known as Jacobi or Zech logarithm. We then present several common properties and for a certain class of primes we derive an alternative method of computing the logarithm that explores the algebraic properties of the dihedral group D_3. We then show how the application of the Zech logarithm to a list of odd integers generates not only the basic TCH codewords but also all the others. Examples of computation are provided and a comparison between the best known TCH codes and the generalized ones obtained here is discussed in terms of correlation signatures and minimum distance.

4.2. Discrete logarithms

In abstract algebra and its applications, discrete logarithms are group-theoretic analogues of ordinary logarithms. In particular, an ordinary logarithm $\log_a(b)$ is a solution of the equation $a^x = b$ over the real or complex numbers. Similarly, if g and h are elements of a finite cyclic group G then a solution x of the equation $g^x = h$ is called a discrete logarithm (to the base g) of h in the group G.

In general, let G be a finite cyclic group with n elements. We assume that the group is written multiplicatively. Let b be a generator of G; then every element g of G can be written in

the form $g = b^k$ for some integer k. We know that any two such integers k_1 and k_2 representing g will be congruent modulo n. Thus we define a function $\log_b : G \to \mathbb{Z}_n$ (where $\mathbb{Z}_n$ denotes the ring of integers modulo n) by assigning to each g the *congruence class* of k modulo n. This function is a group isomorphism, called the discrete logarithm to base b. The familiar base change formula for ordinary logarithms remains valid: If c is another generator of G, then we have $\log_c(g) = \log_c(b) \cdot \log_b(g)$.

DEFINITION 76. Let g be an arbitrary integer relatively prime to n. There exists among the numbers $0, 1, 2, \cdots, \phi(n) - 1$ exactly one number k such that $g \equiv b^k \pmod{n}$. The number k is called the *generalized multiplicative order* (or discrete logarithm) of g with respect to base b modulo n.

4.3. Zech logarithm

Another form of logarithm, called Zech logarithm, have advantages when computing in a finite field $GF(q)$, of order $q = p^m$, with p a prime number and m a positive integer.

DEFINITION 77. Let α be a primitive element of a finite field $GF(q)$. Then, $Z(x)$, the *Zech logarithm* of an integer x is defined such that

$$\alpha^{Z(x)} \equiv 1 + \alpha^x \pmod{q} \tag{4.1}$$

Note that if α^x is the minus one element of the field, then $Z(x)$ is undefined. It is therefore useful to define the set of possible exponents $N_q = \{0, 1, \cdots, q-2\} \cup \{-\infty\}$ so that $Z(x)$ is a mapping $Z : N_q \to N_q$. Any field element β is assumed to be given in its polar representation i.e. as a power of the primitive element α : $\beta = \alpha^k$ and we define $\alpha^{-\infty} = 0$ using the formal symbol $-\infty$ to represent the element 0.

When finite field elements are in polar representation, the multiplication operation is straightforward, i.e., involves adding the exponents modulo $q-1$. The addition operation is performed by,

$$\alpha^n + \alpha^m = \alpha^n \cdot \left(1 + \alpha^{m-n}\right) = \alpha^{n+Z(m-n)} \tag{4.2}$$

Zech logarithms are also called Jacobi logarithms after Jacobi who used them for number theoretic investigations [**48**]. For the case of $GF(p^m)$ with field characteristic p and degree $m \geq 2$ the computation of Zech's logarithms was addressed in [**49**] and [**50**]. A logic architecture (considering the restriction $p = 2$) was presented in [**51**]. The case for $p > 2$ is briefly discussed in [**52**].

4.3.1. Properties. This subsection uses results from the theory of congruences as described in Chapter 4 of [**53**]. From [**49**] it is easy to write the following properties of Zech's logarithm.

$$Z(q-1-x) = Z(x) - x \pmod{q-1}, x \neq -\infty \tag{4.3}$$

$$Z(0) = -\infty,\ p = 2 \tag{4.4}$$

$$Z\left(\frac{q-1}{2}\right) = -\infty,\ p \neq 2 \tag{4.5}$$

$$Z(-\infty) = 0 \tag{4.6}$$

$$Z^{-1}(x) = \begin{cases} Z(x), & p = 2 \\ r + Z(Z(x) - r), & p \neq 2, \alpha^r = p - 2 \end{cases} \tag{4.7}$$

We now focus our attention towards computing Zech's logarithms with a restricted set of elements that we call key elements. We therefore define the mapping $I : N_q \to N_q$ with,

$$I(x) = q - 1 - x \tag{4.8}$$

Note that $I(x)$ corresponds to the additive inverse modulo $q-1$. Hence,

$$Z(I(x)) = Z(x) - x \pmod{q-1} \tag{4.9}$$

and

$$Z^{-1}(Z(x) - x) = I(x). \tag{4.10}$$

We notes from a previous chapter (see theorem 70) that $-1 \equiv \alpha^{(q-1)/2}$. Therefore $-\alpha^{-x+Z^{-1}(x)}$ $= -\alpha^{-x}(-1+\alpha^x) = \alpha^{Z^{-1}(-x)}$. Then,

$$Z^{-1}(I(x)) = \frac{q-1}{2} + Z^{-1}(x) - x, \pmod{q-1} \tag{4.11}$$

We also define

$$\alpha^{Z^{\{i\}}(x)} = i + \alpha^x \tag{4.12}$$

Then in a similar manner we get

$$Z(\xi_i - x) = Z^{\{i\}}(x) - x,\ \alpha^{\xi_i} = i \tag{4.13}$$

Using (4.9) and (4.11) if we alternate between Z and Z^{-1} we get a cycle $x_0 \overset{Z}{\to} x_1 \overset{I}{\leftrightarrow} x_2 \overset{Z}{\leftarrow}$ $x_3 \overset{I}{\leftrightarrow} x_4 \overset{Z}{\to} x_5 \overset{I}{\leftrightarrow} x_0$. Since this involves 6 elements the maximal length of this coset is twelve. Therefore, if we know a $(x, Z(x))$ pair it is easy to determine the corresponding coset.

EXAMPLE. Let $Z(a) = b$ and c (for center) is $c = \frac{q-1}{2}$, i.e. $c = -c \pmod{q-1}$. Then,

$$\begin{aligned}
x_0 &= a,\\
x_1 &= Z(a) = b,\\
x_2 &= -x_1 = -b,\\
x_3 &= Z^{-1}(x_2) = c + a - b,\\
x_4 &= -x_3 = c + b - a,\\
x_5 &= Z(x_4) = c - a,\\
x_6 &= -x_5 = c + a,\\
&\vdots \quad\quad \vdots\\
x_{11} &= Z^{-1}(x_{10}) = -a,\\
x_{12} &= -x_{11} = a.
\end{aligned}$$

4.3.2. Alternative computation of Zech logarithms. The direct form of computing Zech logarithms involves a discrete exponentiation, an addition modulo q and a discrete logarithm. In Mathematica® language, since there is no internal function, we have to create one resulting in

```
Zech[n_,a_,q_]:=MultiplicativeOrder[a,q,Mod[1+Power[a,n],q]]
```

To derive an alternative and less complex way of computing Zech's logarithm we present the following two theorems.

THEOREM 78. *For a non-zero field element in polar representation (exponent x) the following congruences (mod $q-1$) hold,*

$$Z(-x) \equiv Z(x) - x \tag{4.14}$$

$$Z\left(Z(x) + \frac{q-1}{2}\right) \equiv x + \frac{q-1}{2} \tag{4.15}$$

$$Z\left(-Z(x) + \frac{q-1}{2}\right) \equiv x - Z(x) \tag{4.16}$$

$$Z\left(-Z(x) + \frac{q-1}{2} + x\right) \equiv -Z(x) \tag{4.17}$$

$$Z\left(Z(x) + \frac{q-1}{2} - x\right) \equiv \frac{q-1}{2} - x \tag{4.18}$$

PROOF. With respect to (4.14) taking the discrete logarithm of the congruence $\alpha^{Z(-x)} \equiv 1 + \alpha^{-x} \equiv (1+\alpha^x)\,\alpha^{-x} \equiv \alpha^{Z(x)-x}$ yields $Z(-x) \equiv Z(x) - x$.

With respect to (4.15) we have $\alpha^{Z\left(Z(x)+\frac{q-1}{2}\right)} \equiv 1 - \alpha^{Z(x)} \equiv 1 - (1+\alpha^x) \equiv \alpha^{x+\frac{q-1}{2}}$ and therefore $Z\left(Z(x)+\frac{q-1}{2}\right) \equiv x + \frac{q-1}{2}$.

With respect to (4.16) we use (4.14) to write $Z\left(-Z(x)+\frac{q-1}{2}\right) \equiv Z\left(-\left(Z(x)+\frac{q-1}{2}\right)\right) \equiv Z\left(Z(x)+\frac{q-1}{2}\right)-Z(x)+\frac{q-1}{2}$. Using (4.15) we get $Z\left(-Z(x)+\frac{q-1}{2}\right) \equiv x+\frac{q-1}{2}-Z(x)+\frac{q-1}{2} \equiv x-Z(x)$.

The other congruences follow similarly[1]. □

Exploring these symmetries allows us to say that the set of all polar integers x can be partioned into subsets such that the Zech logarithm of all integers in each subset can be computed using the knowledge of only one logarithm in the subset. In a formal way we use the next theorem.

THEOREM 79. *Let the exponent $x \in \mathbb{Z}_{q-1} \setminus \left\{\frac{q-1}{2}\right\}$ and let the key elements $f_i(x)$ be mappings $f_i : \mathbb{Z}_{q-1} \to \mathbb{Z}_{q-1}$, given by*

$$\begin{array}{cc} i & f_i(x) \\ 1 & -x \\ 2 & Z(x)+\frac{q-1}{2} \\ 3 & -Z(x)+\frac{q-1}{2} \\ 4 & -Z(x)+\frac{q-1}{2}+x \\ 5 & Z(x)+\frac{q-1}{2}-x \end{array} \tag{4.19}$$

Let also $F(x)$ be a set such that $F(x) \doteq \{x, f_1(x), f_2(x), f_3(x), f_4(x), f_5(x)\}$. Then, we have $F(x) = F(f_1(x)) = F(f_2(x)) = \cdots = F(f_5(x))$.

PROOF. Let $i \in \{1,2,3,4,5\}$. We consider $f_i(f_1)$ as follows,

$$f_1(f_1(x)) \equiv -f_1(x) \equiv x$$

$$f_2(f_1(x)) \equiv Z(f_1(x)) + \frac{q-1}{2} \equiv Z(x) - x + \frac{q-1}{2} \equiv f_5(x)$$

$$f_3(f_1(x)) \equiv \frac{q-1}{2} - Z(f_1(x)) \equiv \frac{q-1}{2} - (Z(x)-x) \equiv f_4(x)$$

$$f_4(f_1(x)) \equiv \frac{q-1}{2} + f_1(x) - Z(f_1(x)) \equiv \frac{q-1}{2} - x - (Z(x-x)) \equiv f_3(x)$$

$$f_5(f_1(x)) \equiv \frac{q-1}{2} - f_1(x) + Z(f_1(x)) \equiv \frac{q-1}{2} + x + Z(x) - x \equiv f_2(x)$$

and thus $F(f_1(x)) = \{f_1(x), x, f_5(x), f_4(x), f_3(x), f_2(x)\} = F(x)$. All elements $f_i\left(f_j(x)\right)$, for $j = 2,3,4,5$ can be obtained in a similar way. □

All elements $f_i\left(f_j(x)\right)$ are shown in Table 4.1. By row re-ordering it is clear that this table is the same as the Cayley table for the dihedral group D_3, the group of symmetries of an

[1]Note that computations involving field elements are modulo q (e.g. $\alpha^{\frac{q-1}{2}} \equiv -1$) while those involving Zech values are modulo $q-1$.

Table 4.1. Composition of maps corresponding to $f_i(f_j(x))$

$\circ$	j	1	2	3	4	5
i	x	f_1	f_2	f_3	f_4	f_5
1	f_1	x	f_3	f_2	f_5	f_4
2	f_2	f_5	x	f_4	f_3	f_1
3	f_3	f_4	f_1	f_5	f_2	x
4	f_4	f_3	f_5	f_1	x	f_2
5	f_5	f_2	f_4	x	f_1	f_3

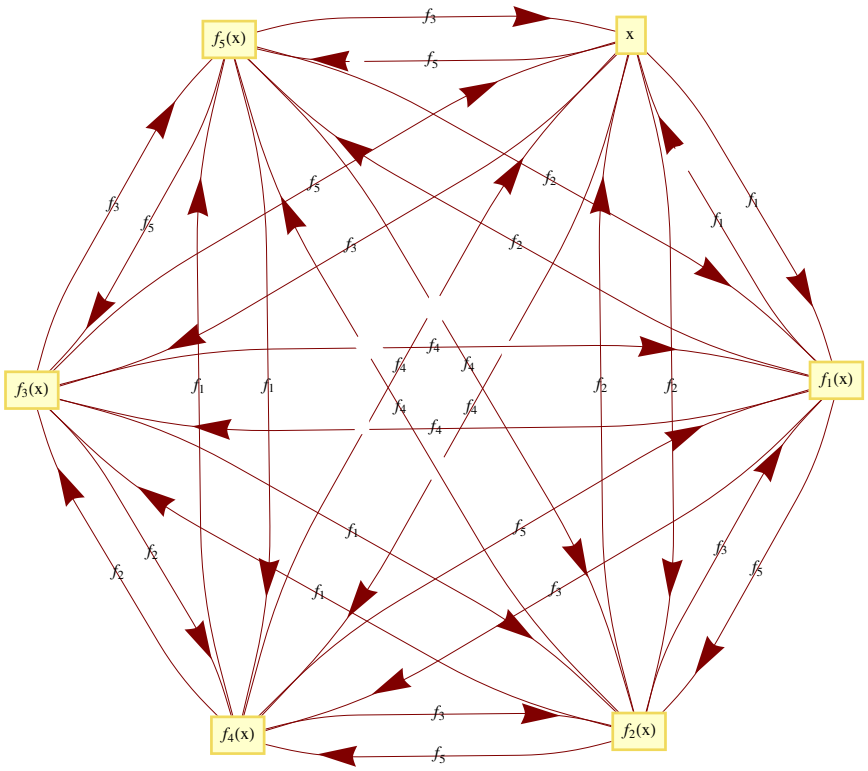


Figure 4.1. Relations between polar integers, i.e. x and the mappings $f_i(x)$

equilateral triangle (see section 2.8). We can consider D_3 as the semi-direct product of cyclic groups $\mathbb{Z}_3$ and $\mathbb{Z}_2$, with $\mathbb{Z}_2$ acting on $\mathbb{Z}_3$ by inversion. Therefore we have 2 pairs of triplets as illustrated in Figure 4.1. Note that the 3 top left elements form the triplet $\{x, f_3(x), f_5(x)\}$ and then by inversion we have $\{-x, -f_3(x), -f_5(x)\} = \{f_1(x), f_2(x), f_4(x)\}$.

Note also that $x \xrightarrow{f_3} f_3(x) \xrightarrow{f_3} f_5(x) \xrightarrow{f_3} x$ so $f_3(x)$ acts like a 120º rotation. For reference, Table 4.2 condenses the information of this subsection.

Table 4.2. Relationship with the dihedral group D_3 of order 6

i	$f_i(x)$	$Z(f_i(x))$	Action
1	$-x$	$Z(x)-x$	Reflection
2	$Z(x)+\frac{q-1}{2}$	$x+\frac{q-1}{2}$	–
3	$-Z(x)+\frac{q-1}{2}$	$-Z(x)+x$	120° Rotation
4	$-Z(x)+\frac{q-1}{2}+x$	$-Z(x)$	–
5	$Z(x)+\frac{q-1}{2}-x$	$\frac{q-1}{2}-x$	–

Table 4.3. All the values of $f_i(x)$ and $Z(f_i(x))$ for $p=17$, $x=0,1,\cdots,p-2$.

x	0	1	2	3	4	5	6	7	9	10	11	12	13	14	15
$f_1(x)$	0	15	14	13	12	11	10	9	7	6	5	4	3	2	1
$f_2(x)$	6	4	11	15	1	7	0	5	14	10	2	13	12	9	3
$f_3(x)$	10	12	5	1	15	9	0	11	2	6	14	3	4	7	13
$f_4(x)$	10	13	7	4	3	14	6	2	11	0	9	15	1	5	12
$f_5(x)$	6	3	9	12	13	2	10	14	5	0	7	1	15	11	4
$Z(x)$	14	12	3	7	9	15	8	13	6	2	10	5	4	1	11
$Z(f_1(x))$	14	11	1	4	5	10	2	6	13	8	15	9	7	3	12
$Z(f_2(x))$	8	9	10	11	12	13	14	15	1	2	3	4	5	6	7
$Z(f_3(x))$	2	5	15	12	11	6	14	10	3	8	1	7	9	13	4
$Z(f_4(x))$	2	4	13	9	7	1	8	3	10	14	6	11	12	15	5
$Z(f_5(x))$	8	7	6	5	4	3	2	1	15	14	13	12	11	10	9

The set of all integers $x \in \mathbb{Z}_{q-1} \setminus \left\{\frac{q-1}{2}\right\}$ can be partioned into disjoint subsets of size six. For $q=p=17$, the complete subset list is presented in Table 4.3. However, one of the subsets only comprises three integers since $f_1(0)=0, f_3(0)=f_4(0), f_2(0)=f_5(0)$ and therefore the set $F(0)=\{0,f_3(0),f_5(0)\}=\left\{0,\frac{q-1}{2}-Z(0),\frac{q-1}{2}+Z(0)\right\}$ which has size three.

Suppose the Zech logarithm of one integer, say y, from each of the above subsets of size six is stored in a table. Then the Zech logarithm $Z(x)$ of an arbitrary integer x can be computed as follows,

(1) Find the unique integer y contained in $F(x)$.
(2) Read $Z(y)$ from the table.
(3) Use the congruences of the above two theorems to compute $Z(x)$ from x, y and $Z(y)$.

4.3.3. Subsets of size six. In the partition of $\mathbb{Z}_{q-1} \setminus \left\{\frac{q-1}{2}\right\}$ there are $\frac{q-5}{6}$ sets of size six. The algorithm to obtain S with all subsets of size six plus the size three subset is as follows[2]:

Require: $q \equiv 5 \pmod{12}$
$S \leftarrow \{\}$

[2]MemberQ[list, form] returns True if an element of list matches form, and False otherwise.
Flatten[list] flattens out nested lists.

$\alpha = \text{PrimitiveRoot}[q]$
$N_S = 1 + (q-5)/6$
$i = -1;\ k = N_S$
while $k > 0$ **do**
  while MemberQ$[\text{Flatten}[S], ++i]$ **do**
    {Empty while}
  end while
  $S \leftarrow \text{Append}\left[S, f_j(i)\right]\ for\ j = 0,1,\ldots,5$ {Note: $f_0 = i$}
  $k \leftarrow k-1$
end while
return S

EXAMPLE. The application of this algorithm for $q = 17$ results in 2 subsets of six elements and 1 subset of three elements $(2 \cdot 6 + 1 \cdot 3 = 15)$,
$S = \{\{0,0,6,10,10,6\}, \{1,15,4,12,13,3\}, \{2,14,11,5,7,9\}\}$.

4.4. Equivalent form of standard TCH

From the introduction to TCH codes we know that each codeword is associated with a primitive element α of GF(p), p being a Fermat prime. Another way of looking at the standard TCH generating equation $\alpha^{K_i} \equiv 1 + \alpha^{2i+1} \pmod p$ is to say that the $\frac{p-1}{2}$ exponents K_i are the Zech logarithms, taken with respect to base α, of all the odd integers up to $(p-2)$, i.e. the integers $1,3,5,\cdots,p-2$. Instead of computing all the Zech logarithms of this set, we can use the information from the previous section to construct a look-up table with just 3 pairs $(x_i, Z(x_i))$,$i = 1,2,3$, from which we compute all remaining values. We give an example.

EXAMPLE. Assume for $p = 17$ that we already have the set S (resulting from Theorem 79) and for the second subset we used the pair $(x_2, Z(x_2)) = (12, Z(12))$. Suppose that we want to compute $Z(3)$. Then we must find the subset where 3 belongs. This is the second subset and from our table we read $Z(12) = 5$. Now, using (4.16) we can compute $Z(8-5) = 12-5 \equiv 7 \pmod{16}$.

Imagine that instead of $x_2 = 12$ we had selected $x_2 = 15$. Now we would read from our table $Z(15) = 11$. Using (4.15) we would compute $Z(11+8) = 15+8 \equiv 7 \pmod{16}$ arriving at the same value. This is the implication of using arbitrarily one of the six key elements in each subset. Moreover if we are only interested in the Zech logarithm of odd integers then only 4 values of those 6 are relevant.

Table 4.4. Zech values, $Z(x)$ for odd x up to $p-2$ using the smallest correspondent primitive element α

$(p,\alpha)\setminus x$	1	3	5	7	9	11	13	15	17	19	21	23	25	27
(17,3)	12	7	15	13	6	10	4	11						
(19,2)	13	8	7	11	$-\infty$	4	2	5	12					
(23,5)	18	9	13	12	20	$-\infty$	11	5	8	6	17			
(29,2)	5	10	2	18	24	9	27	14	26	15	11	25	7	4

4.5. Generalized TCH codewords

We now lift the restriction of p being a Fermat prime and proceed by searching an appropriate finite field for which $p-1$ is closer to the desired value of N. Then to generate a single codeword we use (arbitrarily) the smaller of $GF(p)$ primitive elements.

4.5.1. Single codeword. In order to generalize the generation of TCH codewords we compute the Zech logarithms for all the odd integers up to $p-2$ for other values of p, i.e. for primes not restricted to Fermat primes. In Table 4.4 we present such values for the first three primes after $p=17$. Since we want balanced codewords, i.e. binary codewords with weight equal to 0, we must disregard the cases for $p=19$ and $p=23$ and consider as good cases those of $p=17$ and $p=29$. Note also that the bad cases, where $Z(x)=-\infty$, correspond to odd values $x=(p-1)/2$.

We are working odd primes p. Modulo an odd prime p there are $(p+1)/2$ quadratic residues (including 0) and $(p-1)/2$ nonresidues. As we know an integer g is a quadratic residue modulo p if it is congruent to a perfect square modulo p and is a quadratic nonresidue modulo p otherwise. We introduced the Legendre symbol earlier as a function of g and p and defined as follows

$$\left(\frac{g}{p}\right) \equiv g^{\frac{p-1}{2}} \pmod{p} \tag{4.20}$$

giving the value 1 if g is a quadratic residue and -1 if g is a quadratic nonresidue.

The first supplement to the law of quadratic reciprocity (see theorem 70) tell us that

$$\left(\frac{-1}{p}\right) = (-1)^{\frac{p-1}{2}} = \begin{cases} 1 & \text{if } p \equiv 1 \pmod{4} \\ -1 & \text{if } p \equiv 3 \pmod{4} \end{cases} \tag{4.21}$$

Now $-1 \pmod{p}$ i.e. $p-1 \pmod{p}$ cannot be a primitive root because $(p-1)^2 = p^2 - 2p+1 \equiv 1 \pmod{p}$, i.e. the multiplicative order of $p-1$ is 2 while that of a generator is $p-1$. Therefore we conclude that the class of good primes is the class corresponding to Pythagorean primes, i.e. primes $p \equiv 1 \pmod{4}$. For this class, p is of the form $4k+1$ and thus $(p-1)/2 = 4k/2$ is always even.

We are interested in obtaining codewords with length N equal to a power of two. We also know that this can only be obtained exactly for Fermat primes so for the other cases, i.e. $N \in \{32, 64, 128, 512, 1024\}$ we need to select the closest possible adequate prime.

DEFINITION 80. The discrete autocorrelation R_{xx} at lag j for a discrete periodic sequence $\{x_n\}_{n=0}^{N-1}$ is

$$R_{xx}(j) = \sum_{n=0}^{N-1} x_n \bar{x}_{n+j} \tag{4.22}$$

where $\bar{x}$ denotes the complex conjugate of x.

EXAMPLE. Let $N = 32$. The closest prime congruent to 1 modulo 4 is is $p = 29$ which will generate codewords of length $p-1 = 28$. The smallest primitive root modulo 29 is 2. Applying the Zech logarithm to the list of odd integers gives us the codeword F04CEB4_{16} in hexadecimal format. We can check the 'ON' bits by looking at the position given by the Zech values on the last line of Table 4.4. For instance, the bits at positions 24 to 27 are all set translating into the most significant F_{16} in the codeword.

However the length (or period) of this codeword is 28. In order to get a codeword with $N = 32$ we will prepend a 4 bit code from the Fermat case $p = 5$, i.e. either C_{16} or 6_{16}. The best alternative is the one that maximizes the minimum distance of the codeword which for this example corresponds to the prefix 6_{16} giving the generalized 32-bit TCH codeword 6F04CEB4_{16}. This codeword has a weight of 16 and a minimum distance of 12. Its autocorrelation signature is depicted in Figure 4.2. The peak at lag 0 corresponds to the length of the codeword, i.e. 32, and the other correlation values fall between $[-8, 8]$ instead of $[-4, 0]$ as was the case for the standard TCH codewords. Note, however that the correlation value c_i for odd lag i is always 0 and $\sum_{i=0}^{31} c_i = 0$.

In Figure 4.3 we present the autocorrelation signature of a previously obtained 32-bit codeword, from TCH(32,7,5). In the notation TCH(n, k, t), t is the error correcting capability such that $d_{min} \geq 2t+1$. Even though the values for lag $j \neq 0$ are between -8 and 8 the odd lag values are not 0 as in our generalized example.

4.5.2. The set of generalized codewords. Each codeword is associated with a primitive element of $GF(p)$. We can calculate all the primitive elements using the Legendre symbol to obtain all the quadratic nonresidues (Table 4.5) which for $p = 29$ results in, 2, then $3, 8, 10, 11, \cdots, 21, 26$ and 27. Then we check the multiplicative order of each nonresidue. Those who have order $p-1$ are primitive elements. In this case we have to remove the elements 12 and 17, giving a total of 12 primitive elements.

Applying the same procedure described for the single codeword we now generate all 12 codewords for $p = 29$ and determine the best prefix for each one by evaluating the minimum

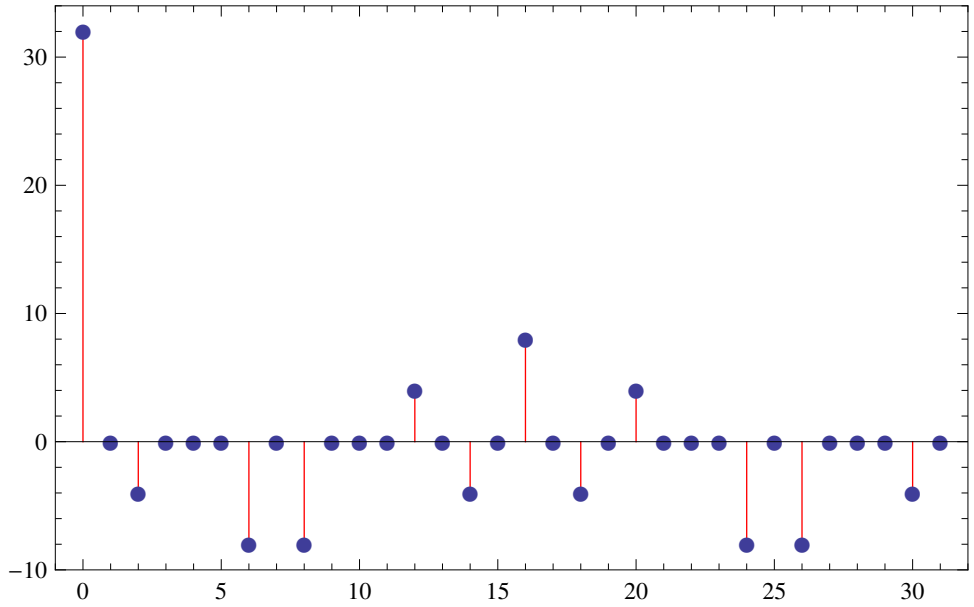


Figure 4.2. The autocorrelation of generalized codeword $6F04CEB4_{16}$ resulting from $(p = 29, \alpha = 2)$.

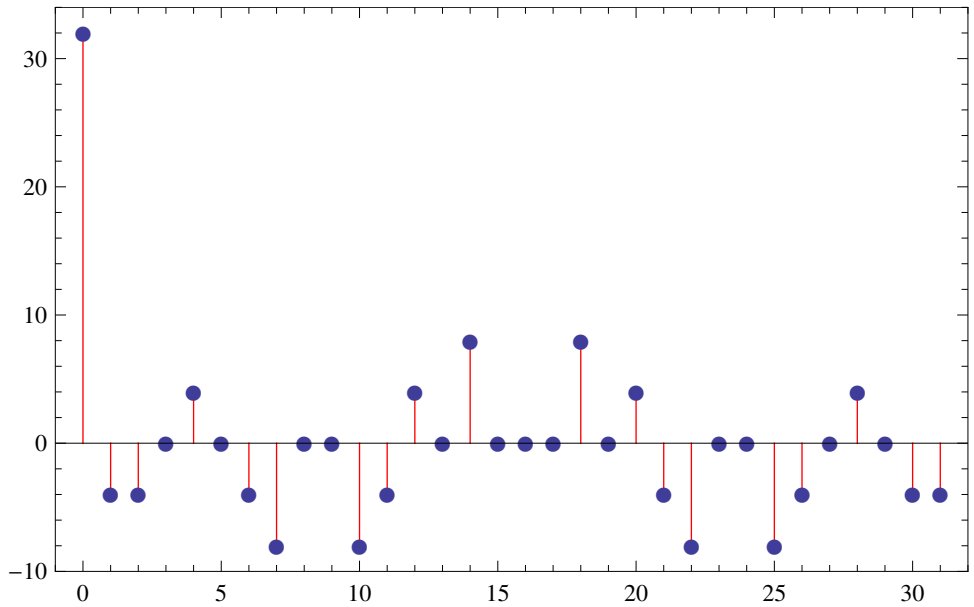


Figure 4.3. The autocorrelation of a codeword from TCH(32,7,5)

Table 4.5. All primitive elements of $GF(29)$ are a subset of the nonresidues given by the Legendre symbol.

i	1	2	3	4	5	6	7	8	9	10	11	$\cdots$	24	25	26	27	28
$\left(\frac{i}{29}\right)$	1	−1	−1	1	1	1	1	−1	1	−1	−1	$\cdots$	1	1	−1	−1	1

Table 4.6. The best generalized 32-bit TCH codewords via Zech logarithms in $GF(29)$.

α	Codeword	0	C	6
2	F04CEB4	12	8	12
8	8746F68	14	14	10
3	44978EE	14	14	14
19	4C17ACE	12	14	14
18	74CF6A	14	14	12
14	D06C6BC	12	14	10
27	7AC6C16	12	10	14
21	ADE65C0	14	14	14
26	E6BD064	12	14	14
10	EE3D244	14	14	14
11	2DEC5C2	14	10	14
15	5AE641E	12	12	8

distance. The list is presented in Table 4.6. The first column has the primitive element used. The second column contains the 28-bit codeword in hexadecimal format. The values in the last three columns correspond to the minimum distance of the prefixed code (4+28=32 bit) and the 4-bit prefix is the hexadecimal label of each column. Take the second line as an example. The value 14 under the heading "C" means that the codeword 8746F68 when prefixed with C, i.e. the codeword C8746F68, has a minimum distance of 14.

DEFINITION 81. The discrete cross-correlation R_{xy} at lag j between the discrete signals x_n and y_n is

$$R_{xy}(j) = \sum_n x_n \overline{y}_{n+j} \tag{4.23}$$

where $\overline{x}$ denotes the complex conjugate of x.

For comparison we present in Figure 4.4 the cross-correlation between the prefixed codewords $C8746F68_{16}$ and $C44978EE_{16}$. Note that all cross-correlation values are contained in the interval $[-8,12]$ staying well below the peak value (in this case below half the peak), i.e. $|c_i| < \frac{32}{2} = 16$.

For comparison we also present in Figure 4.5the cross-correlation between codewords from a previously obtained TCH(32,7,5).

From the list of Table 4.6 it is possible to select 8 codewords with minimum distance of $d_{min} = 14$ to define a code TCH(32,9).

The best 32-bit codewords presented in [**6**] correspond to a code TCH$(32,6)$ with $d_{min} = 14$. We have therefore gained 3 information bits by our approach. To explore the relationship between rate and distance, a set of lower and upper bounds of block codes are known. In Figure 4.6 we used as upper bound the maximum of Plotkin and Hamming and as a lower bound the

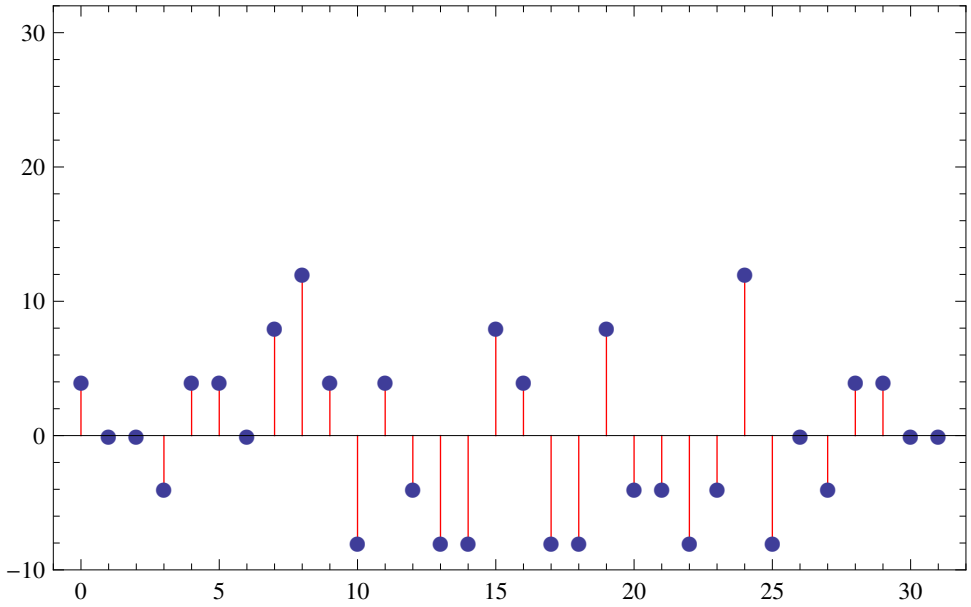


Figure 4.4. The cross-correlation between the (prefixed by C_{16}) generalized codewords $C8746F68_{16}$ from $(p = 29, \alpha = 8)$ and $C44978EE_{16}$ from $(p = 29, \alpha = 3)$.

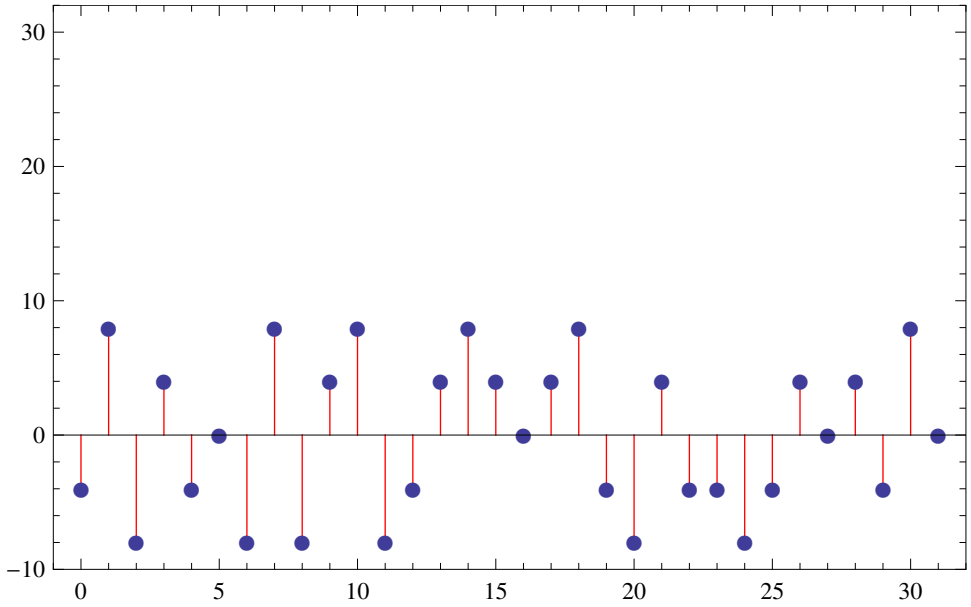


Figure 4.5. Cross-correlation between codewords from TCH(32,7,5)

minimum of Gilbert and TCH (the latter being $k = 1 + \log_2 N$). We also plotted the best standard TCH and the best generalized TCH developed in this chapter.

Before we present the results for codewords of length $64, 128$ we end this section with an outlier set of codewords. By outlier we mean the following. If we disregard the restriction $p \equiv 1 \pmod 4$ we can use $p = 31$ which will produce 30-bit codewords listed in Table 4.7. The

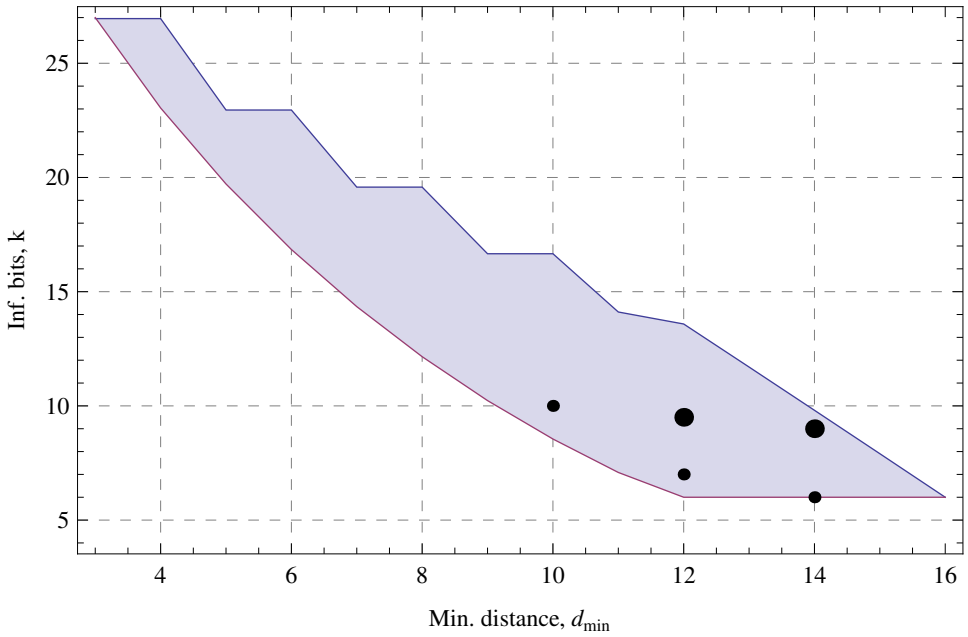


Figure 4.6. Comparison between the best 32-bit codes. The smaller points are for the standard TCH and larger points are for the generalized TCH via Zech logarithms.

Table 4.7. The best generalized 32-bit TCH codewords via Zech logarithms in $GF(31)$.

α	Codeword	0	C	6
3	1C4FAC48	12	12	12
17	11AD8F90	14	10	12
13	2C9581CE	14	14	14
24	2063B656	12	14	12
22	3536E302	12	12	12
12	39C0D49A	14	14	14
11	4F8DAC4	14	10	12
21	91AF91C	12	8	8

first column has the primitive element used. The second column contains the 30-bit codeword in hexadecimal format. The values in the last three columns correspond to the minimum distance of the prefixed code. The most significant 4 bits of the original codeword are bit OR-ed with the 4 bits of the prefix (the hexadecimal label of each column).

In order to get 32 bits we need an additional prefix of two bits but the prefixes we have been using have 4 bits. Thus in this case the prefix operation involves the OR operation between the 4 bits of the prefix and the most significant bits of the original codeword. As an example the first codeword $1C4FAC48_{16}$ when prefixed by 6_{16} will turn into $7C4FAC48_{16}$. On the other hand, the last codeword $91AF91C_{16}$ has only 7 hexadecimal characters so it will turn into the usual $691AF91C_{16}$ when prefixed by 6_{16}.

Table 4.8. The best generalized 64-bit TCH codewords via Zech logarithms in $GF(53)$ with codeword prefixes from $GF(13)$.

α	Codeword	000	978	1DA	B70	3D2
2	E01B79D971AA4	26	24	24	24	24
8	9F12417AE3B58	26	28	26	28	26
32	3BED29D83C512	26	20	20	24	24
22	3872BDDD03692	26	26	22	26	26
35	5DA9306B381DE	24	18	26	18	26
34	B9536C73F90A	26	24	26	24	28
14	A315BC6D1FB20	26	26	22	26	26
3	570B986990B7E	26	26	26	26	26
12	9AF21F75A3418	26	24	26	24	24
48	9B4FAB7014F18	26	28	24	24	26
33	153069D86B3FA	26	28	28	28	28
26	CA3953D3F920C	26	28	24	28	20
51	6093F979538A6	26	20	28	24	28
45	BF9AC372C1950	26	28	28	28	28
21	31E501DABE5B2	26	26	24	24	28
31	3058B5DF09EB2	26	24	24	26	24
18	FDA132C33A1D4	26	26	26	26	26
19	9BF16C7B518A	26	26	24	26	26
39	A13F9C6D953A0	26	24	22	28	26
50	F7039AC192B74	24	26	18	26	18
41	92D81777A9C38	26	26	26	22	26
5	9147837296FB8	26	24	24	20	20
20	35B8EBD0491F2	26	26	28	26	28
27	4AB1D373DB00E	26	24	24	24	24

4.6. Codewords for other lengths

For the case of 64-bit codewords we use the prime $p = 53$ giving codewords of period 52. In order to obtain the extra 12 bits we prepend codewords from using $p = 13$ as given in table 4.8. For each prefix used we list the minimum distance obtained. Thus, by selectively gathering codewords we can construct a code GTCH(64,10) with $d_{min} = 28$ which is much better than the same size code TCH(64,10) of minimum distance 24.

Alternatively we can use the prime $p = 61$ to obtain 60-bit codewords which are then prefixed with 4-bit codewords. Figure 4.7 illustrates the autocorrelation of one example, given by the codeword from $(p = 61, \alpha = 10)$ prefixed with C_{16}.

For the case of 128-bit codewords we use $p = 113$ generating 48 codewords of 112 bits each. To obtain the full length codewords we prepend with codewords from the case $p =$

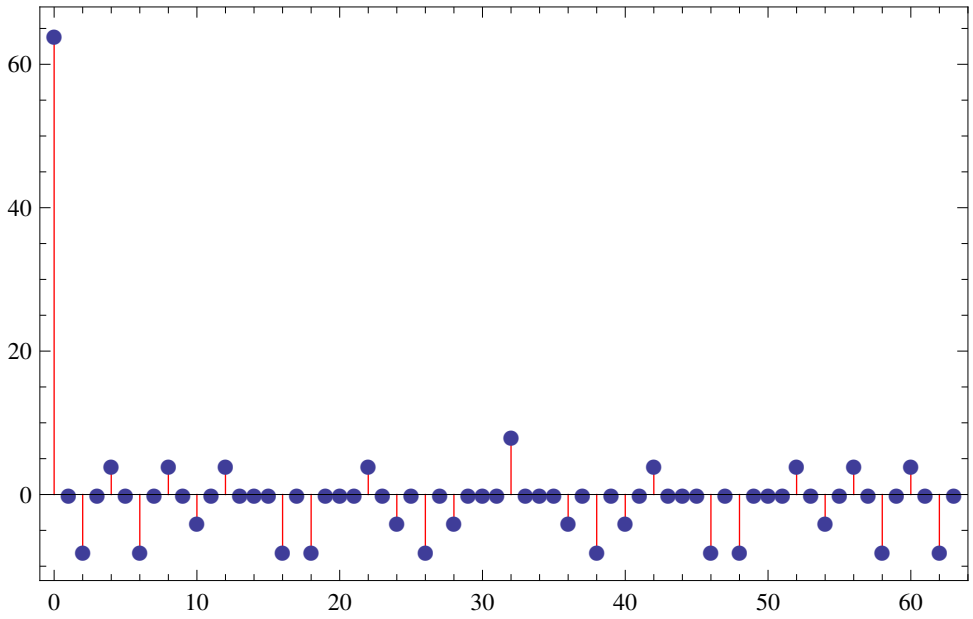


Figure 4.7. The autocorrelation of the 64-bit generalized codeword resulting from $(p = 61, \alpha = 10)$ using a C prefix.

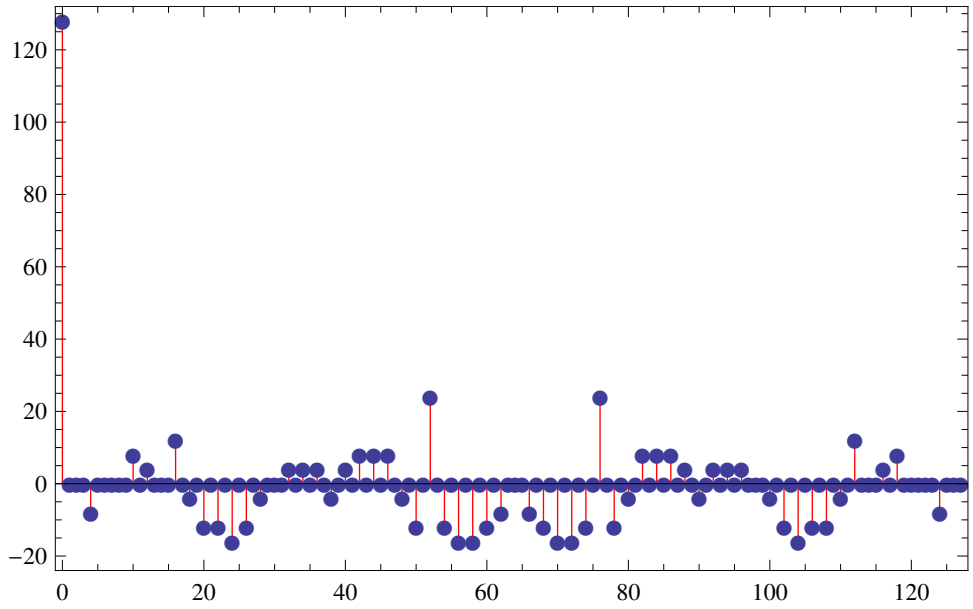


Figure 4.8. The autocorrelation of a 128-bit generalized codeword. The values for a lag $\neq 0$ are within $[-16, 24]$.

17, since 112+16=128. In figures 4.8 and 4.9 we plot samples of the autocorrelation and the crosscorrelation.

4.6.1. Zech relation between codewords. Since a primitive element, say β, can be computed as a power of another primitive element, say α, we can relate two different codewords

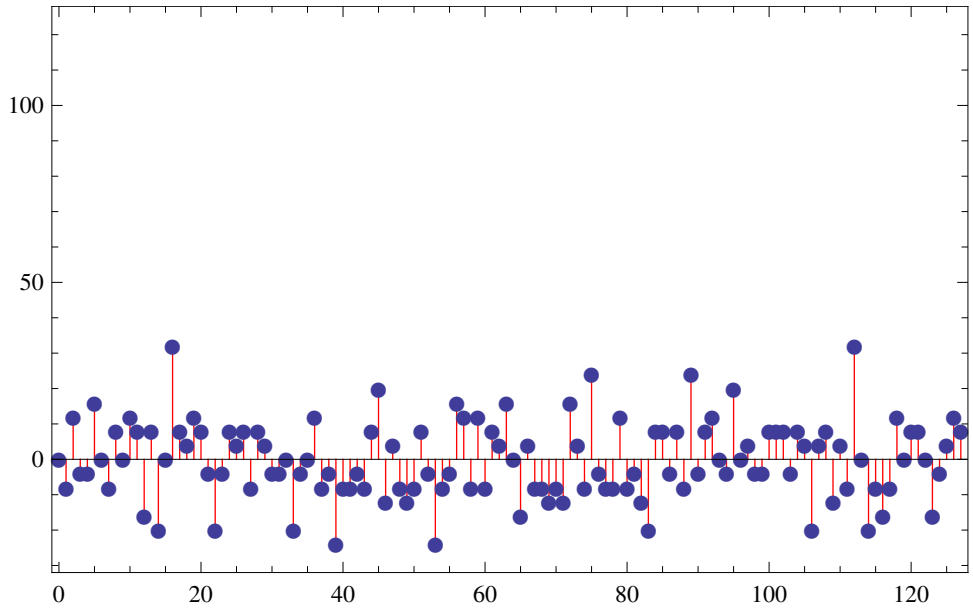


Figure 4.9. The crosscorrelation of two 128-bit generalized codewords. All values are within $[-24, 32]$.

using Zech's log properties. Using the definition (Eq. 4.1) of Zech's logarithm and assuming $\beta = \alpha^u$, we have $1 + \alpha^{ux} \equiv \alpha^{uZ_\beta(x)}$ but we also have $1 + \alpha^{ux} \equiv \alpha^{Z_\alpha(ux)}$ where the subscript of Z indicates the primitive element used. Then the following relations are easily derived:

$$Z_\alpha(ux) \equiv uZ_\beta(x) \mod (p-1)$$

$$Z_\beta(x) \equiv u^{\phi(p-1)-1} Z_\alpha(ux) \mod (p-1)$$

with $u^{\phi(p-1)-1}$ being the multiplicative inverse of u and $\phi(\cdots)$ the Euler phi-function.

EXAMPLE. Let us consider the codeword $\mathrm{BCD0}_{16}$ from $(p = 17, \alpha = 3)$ given by the list of Zech values $\{12, 7, 15, 13, 6, 10, 4, 11\}$. Now we want to determine the codeword from $(p = 17, \beta = 10)$. Since $10 \equiv \alpha^3$ we first find the multiplicative inverse of 3 which is 11 (since $3 \cdot 11 = 33 = 1$ modulo 16). Then we multiply the list of Zech values by 11 giving $\{4, 13, 5, 15, 2, 14, 12, 9\}$ resulting in the codeword $\mathrm{F234}_{16}$. Note that we have substituted a complete computation of Zech values by a much simpler multiplication modulo $p - 1$.

4.6.2. Iterative construction of the permutations matrix. We are working with codewords generated by $1 + \alpha^{2j+1}$. As we saw in the previous example any primitive element β can be expressed as an odd power of α, i.e. $\beta = \alpha^{2k+1}$. Therefore the odd powers of β are $\beta^{2j+1} \equiv$

$\alpha^{(2k+1)(2j+1)}$. We then have, $(2k+1)(2j+1) = 4kj+2k+2j+1 = 2[2kj+(k+j)]+1$ resulting in,

$$j \equiv 2kj + j + k \quad \left(\mathrm{mod}\ \frac{N}{2}\right) \tag{4.24}$$

that we tabulate (for the case $p = 17$) in the following permutation matrix,

$$X = \left[x_{kj}\right] = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 4 & 7 & 2 & 5 & 0 & 3 & 6 \\ 2 & 7 & 4 & 1 & 6 & 3 & 0 & 5 \\ 3 & 2 & 1 & 0 & 7 & 6 & 5 & 4 \\ 4 & 5 & 6 & 7 & 0 & 1 & 2 & 3 \\ 5 & 0 & 3 & 6 & 1 & 4 & 7 & 2 \\ 6 & 3 & 0 & 5 & 2 & 7 & 4 & 1 \\ 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \end{bmatrix} \tag{4.25}$$

where $x_{kj} = 2kj + j + k$ modulo $\frac{N}{2}$. The application of the Zech logarithm to each row k results in the codeword associated with the primitive element α^{2k+1} (we assume a reference primitive element, α, which most often is the smallest of all the primitive elements). For each codeword, half of the bits of the codeword, i.e. those at position $Z_\beta\left(x_{kj}\right)$ have the value 1 while the remaining half have the value 0. Note also that (in this case) since $N = p-1 = 16$, the congruence of Eq. (4.24) is taken modulo 8.

To generalize this formulation we obtain i from the equation $N = 4 \cdot 2^i$ where N is the desired codeword length. So for $N = 32$, i would be $i = 3$. Then we construct the matrix

$$x_{kj} \quad = \quad 2kj + j + k \quad \left(\mathrm{mod}\ 2^{i+1}\right), \tag{4.26}$$

for row $k = 0, 1, \ldots, 2^{i+1}-1$ and column $j = 0, 1, \ldots, 2^{i+1}-1$. Then for each row we apply the Zech log taken to base α^{2k+1} with an arbitrarily chosen primitive element α.

The matrix (4.25) is of the form $\begin{bmatrix} A & A+4 \\ A+4 & A \end{bmatrix} (\mathrm{mod}\ 8)$ and we also note that A (mod 4) $= \begin{bmatrix} B & B+2 \\ B+2 & B \end{bmatrix}$ with $B = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. In addition we have $A = \begin{bmatrix} B & B+2 \\ B+2 & B \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$. This gave us the motivation to pursue an iterated construction of the matrix of permutations. Before we present the resulting algorithm we need to define the Kronecker product.

DEFINITION 82. If A is an m-by-n matrix and B is a s-by-t matrix, then the Kronecker product $A \otimes B$ is the ms-by-nt block matrix,

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix} \tag{4.27}$$

Let us work out an example for the iterated construction of the matrix of permutations X. We start with $X_0 = [0]$, the scalar value 0.

At step $i = 0$ we construct

$$X_1 = \begin{bmatrix} X_0 & X_0+1 \\ X_0+1 & X_0 \end{bmatrix} \left(\mathrm{mod}\ 2^{i+1}\right) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} (\mathrm{mod}\ 2)$$

We then add a ring matrix R_0 which in this step is a 2×2 matrix with zeros. So, $X_1 = X_1 + R_0$ remains unaltered.

At step $i = 1$, we repeat

$$X_2 = \begin{bmatrix} X_1 & X_1+2 \\ X_1+2 & X_1 \end{bmatrix} \left(\mathrm{mod}\ 2^{i+1}\right) = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 0 & 3 & 2 \\ 2 & 3 & 0 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix} (\mathrm{mod}\ 4)$$

We then add a ring matrix R_1 which in this step is

$$R_{i=1} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2^{i+1} & 2^{i+1} & 0 \\ 0 & 2^{i+1} & 2^{i+1} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 4 & 4 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

generating, $X_2 = X_2 + R_1 = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 1 & 4 & 7 & 2 \\ 2 & 7 & 4 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix}$.

At step $i = 2$, we have

$$X_3 = \begin{bmatrix} X_2 & X_2+4 \\ X_2+4 & X_2 \end{bmatrix} \pmod{2^{i+1}} = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 4 & 7 & 2 & 5 & 0 & 3 & 6 \\ 2 & 7 & 4 & 1 & 6 & 3 & 0 & 5 \\ 3 & 2 & 1 & 0 & 7 & 6 & 5 & 4 \\ 4 & 5 & 6 & 7 & 0 & 1 & 2 & 3 \\ 5 & 0 & 3 & 6 & 1 & 4 & 7 & 2 \\ 6 & 3 & 0 & 5 & 2 & 7 & 4 & 1 \\ 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0 \end{bmatrix} \pmod 8$$

which is the matrix (4.25) obtained above. The number of steps depends on the codeword length N.

The complete generalization is given by the following algorithm.

Let I_k be the $k \times k$ identity matrix, so that $I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$. Let J_k be the $k \times k$ exchange matrix (with ones on the counter diagonal), so that $J_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. Let also B_k be the $k \times k$ box matrix (filled with 1's), so that $B_2 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$.

Require: $N = 2^v, v \geq 2$

$i = 0;\ X = [0];\ R = \left[r_{kj}\right]$

repeat

$X \leftarrow B_2 \otimes X + J_2 \otimes 2^i B_{2^i}$

$$r_{kj} = \begin{cases} 2^{i+1} & \text{if Mod}\left[2kj + j + k,\, 2^{i+1}\right] > \left(2^{i+1} - 1\right) \\ 0 & \text{Otherwise} \end{cases} \quad \{k, j = 0, 1, \ldots, 2^{i+1} - 1\}$$

$X \leftarrow X + R$

$i \leftarrow i + 1$

until $4 \cdot 2^i > N$

return $X - R$

4.7. Conclusion

In this chapter we have presented an algebraic method to obtain binary generalized TCH codewords of length $N = 2^k, k = 1, 2, \ldots, 16$. By exploring Zech logarithm's properties as well as a group theoretic isomorphism this method is both faster and less complex than what was proposed before. In addition, it is valid for all relevant cases relating the codeword length N and not only those resulting from $N = p_i - 1$ for Fermat primes p_i. The method also derives the

maximum set of all the codewords of a certain code bringing clear advantages in terms of code size and minimum distance. This work was accepted for publication in [**21**].

CHAPTER 5

PERMUTATIONS OF SIDEL'NIKOV SEQUENCES

5.1. Introduction

In high-speed data communications, M-ary modulation schemes are commonly used. Several classes of M-ary codes with good error-correction capabilities as well as M-ary sequences with good correlation properties have been employed. Auto-correlation properties are important for synchronization purposes and cross-correlation properties are useful for multi-access techniques. In this scenario, a large number of distinct sequences is necessary to support as many distinct users or channels as possible. From the implementation point of view it is important to generate such large sequence sets as efficiently as possible, i.e. reducing the computational resources (i.e., time and memory) employed in sequence set generation and its subsequent use.

This chapter is organized as follows. Section 5.2 identifies permutations between TCH codewords. Using the notion of quadratic residues we partition the codeword in half and use appropriate permutations of the set $\mathbb{Z}_{p-1}$ in section 5.3. Section 5.4 discusses some relevant facts from group theory in order to extend the results to the generalized TCH codewords. Restricting the primes to the class of Pythagorean primes reveals a very interesting isomorphism that is presented in section 5.5. Armed with the previous results we propose, in Section 5.6, an efficient method to generate, in time and frequency domains, certain sequences associated to the primitive elements of the Galois field $GF(p)$. In Section 5.7 we recall the definition of Sidel'nikov sequences and we apply the results of Section 5.6 to generate all Sidel'nikov sequences of period $p-1$. We then link TCH codewords with Sidel'nikov sequences and prove, in section 5.8, that TCH codewords are a subset of those sequences. Some examples are included in Section 5.9. Finally, in section 5.10, the permutation-based generation procedure is extended to Sidel'nikov sequences of period p^n-1 allowing the construction of M-ary sequences.

Table 5.1. Two solutions of equation (5.1) for $\alpha = 3$ and $\beta = 5$.

i	$\alpha = 3$		$\beta = 5$	
	$1+\alpha^{2i+1}$	K_i	$1+\beta^{2i+1}$	K_i
0	4	12	6	3
1	11	7	7	15
2	6	15	15	14
3	12	13	11	11
4	15	6	13	4
5	8	10	12	9
6	13	4	4	12
7	7	11	8	2

5.2. Permutations among TCH codewords

In this chapter we use standard results and nomenclature from finite field theory [**54**]. However the essential theory is presented in Chapters 2 and 3. Let α be a primitive element of GF(p). Then α generates the group $<\alpha>=\{\alpha^0, \alpha^1, \alpha^2, \cdots, \alpha^{p-2}\}$. We recall the generating equation for TCH basic polynomials:

$$\alpha^{K_i} \equiv 1+\alpha^{2i+1},\ i = 0, 1, \cdots, \frac{p-1}{2} - 1 \tag{5.1}$$

where p is an odd prime of the form $p = F_n = 2^{2^n}+1$, i.e. p is a Fermat prime.

The exponents $K_i \in \{1, 2, \cdots, 2^n - 1\}$ specify the powers of x in the polynomial $h_\alpha(x)$,

$$h_\alpha(x) = \sum_{i=0}^{\frac{p-1}{2}-1} x^{K_i} \tag{5.2}$$

In table 5.1 we give two solutions of (5.1) for two different primitive elements $\alpha = 3$ and $\beta = 5$.

It is clear that the set $1+\beta^{2i+1}$ is a permutation $1+\alpha^{2i+1}$ which provided the motivation for studying permutations among TCH codewords.

Consider $\mathbb{F}_p^* = \text{GF}(p) \setminus \{0\}$ the multiplicative group of units (invertible elements) of GF(p), a cyclic group of order $p-1$. Let Q_R be the set of quadratic residues, i.e. $Q_R = \{\alpha^k \mid \gcd(k, p-1) \neq 1\} \pmod p$ This set is a multiplicative subgroup of $\mathbb{F}_p^*$ of index 2. Let η be a non-quadratic residue. The coset $\overline{Q_R} = \eta Q_R = \{\eta j;\ j \in Q_R\}$ consists of the non-quadratic residues. We may assume that η is the smallest non-quadratic residue modulo p that generates the multiplicative group, i.e., $\mathbb{F}_p^* = Q_R \cup \overline{Q_R}$.

The set $\{\alpha^{K_i},\ i = 0, 1, \cdots, \frac{p-1}{2} - 1\}$ in (5.1) is $1+\overline{Q_R}$ that we partition into,

$$1+\overline{Q_R} = \Sigma \cup \overline{\Sigma} \tag{5.3}$$

with

$$\Sigma = (1 + \overline{Q_R}) \cap \overline{Q_R} \tag{5.4}$$

and

$$\overline{\Sigma} = (1 + \overline{Q_R}) \cap Q_R \tag{5.5}$$

Denoting,

$$\Theta(\alpha) = \left\{ j \in \mathbb{Z}_{\frac{p-1}{2}} : \alpha^{2j+1} \in \Sigma \right\} \tag{5.6}$$

$$\overline{\Theta}(\alpha) = \left\{ j \in \mathbb{Z}_{\frac{p-1}{2}} : \alpha^{2j} \in \overline{\Sigma} \right\} \tag{5.7}$$

the polynomial (5.2) can be alternatively defined as

$$h_\alpha(x) = \sum_{j \in \Theta(\alpha)} x^{2j+1} + \sum_{j \in \overline{\Theta}(\alpha)} x^{2j}. \tag{5.8}$$

We denote by $H_\alpha = (a_{\frac{p-1}{2}-1}, \cdots, a_1, a_{\bar{1}}, a_0, a_{\bar{0}})$, $a_i \in \mathbb{Z}_2$, the binary representation (codeword) of $h_\alpha(x) = \sum_{k=0}^{p-2} b_k x^k$ where $b_k = 1$ if the monomial x^k exists in (5.8) and $b_k = 0$ otherwise. To complete the definition,

$$\begin{cases} a_{\bar{k}} = b_{2k} \\ a_k = b_{2k+1} \quad k = 0, 1, \ldots, \frac{p-1}{2} - 1 \end{cases} \tag{5.9}$$

5.3. Permutations of $\mathbb{Z}_{p-1}$

Permutations are discussed in section A.1. Given primitive elements α and β, there is a permutation $\pi_{\alpha \to \beta}$ of the set $\{0, 1, 2, \cdots, p-2\}$ such that

$$\pi(\theta(\alpha)) = \theta(\beta) \qquad \pi\left(\bar{\theta}(\alpha)\right) = \bar{\theta}(\beta). \tag{5.10}$$

Since p is a Fermat prime we have,

$$\overline{Q_R} = \left\{ \mathrm{p}_\alpha^{\mathrm{odd}}[k] : k = 0, 1, 2, \ldots, \frac{p-1}{2} - 1 \right\} \tag{5.11}$$

$$Q_R = \left\{ \mathrm{p}_\alpha^{\mathrm{ev}}[k] : k = 0, 1, 2, \ldots, \frac{p-1}{2} - 1 \right\} \tag{5.12}$$

where,

$$\mathrm{p}_\alpha^{\mathrm{odd}}[k] \doteq \alpha^{2k+1}, k \in \mathbb{Z}_{\frac{p-1}{2}} \tag{5.13}$$

and

$$\mathrm{p}_\alpha^{\mathrm{ev}}[k] \doteq \alpha^{2k}, k \in \mathbb{Z}_{\frac{p-1}{2}}. \tag{5.14}$$

LEMMA 83. *If* $\beta = p_\alpha^{odd}[k] = \alpha^{2k+1}$ *then*

a) $\mathrm{p}_{\beta}^{\mathrm{odd}}[j] = \mathrm{p}_{\alpha}^{\mathrm{odd}}[k+(2k+1)\,j]$
b) $\mathrm{p}_{\beta}^{\mathrm{ev}}[j] = \mathrm{p}_{\alpha}^{\mathrm{ev}}[(2k+1)\,j]$

PROOF. By variable substitution and working out the product the proof follows trivially from the definition (5.13), (5.14) and we omit it. □

The map $j \mapsto k+(2k+1)j$, $j \in \mathbb{Z}_{\frac{p-1}{2}}$, defines a permutation,

$$\sigma_k = \begin{pmatrix} 0 & 1 & \cdots & j & \cdots & \frac{p-1}{2}-1 \\ k+(2k+1)\,0 & k+(2k+1)\,1 & \cdots & k+(2k+1)\,j & \cdots & k+(2k+1)\,(\frac{p-1}{2}-1) \end{pmatrix} \tag{5.15}$$

Analogously the map $\bar{j} \mapsto \overline{(2k+1)\,j}$, $\bar{j} \in \mathbb{Z}_{\frac{p-1}{2}}$, defines a permutation,

$$\bar{\sigma}_k = \begin{pmatrix} \overline{0} & \overline{1} & \cdots & \overline{j} & \cdots & \overline{\frac{p-1}{2}-1} \\ \overline{(2k+1)\,0} & \overline{(2k+1)\,1} & \cdots & \overline{(2k+1)\,j} & \cdots & \overline{(2k+1)\,(\frac{p-1}{2}-1)} \end{pmatrix}. \tag{5.16}$$

The permutation relating the binary representations H_α and H_β, of the polynomials $h_\alpha(x)$ and $h_\beta(x)$, associated with α and $\beta = \alpha^{2k+1}$ respectively, is the "intertwined" application of σ_k and $\bar{\sigma}_k$, i.e. $H_\alpha \overset{\pi_k}{\to} H_\beta$:

$$\pi_k = \begin{pmatrix} \frac{p-1}{2}-1 & \cdots & \overline{1} & 0 & \overline{0} \\ k+(2k+1)\,(\frac{p-1}{2}-1) & \cdots & \overline{(2k+1)\,1} & k+(2k+1)\,0 & \overline{(2k+1)\,0} \end{pmatrix} \tag{5.17}$$

LEMMA 84. *The permutation π_k has k-dependent order $L = ord_{p-1}(1+2k)$.*

PROOF. After L iterations of the map $j \mapsto k+(2k+1)j$, we get

$$j \mapsto \sum_{i=0}^{L-1} k\,(2k+1)^i + j \prod_{i=0}^{L-1} (2k+1) \tag{5.18}$$

and since $\sum_{i=0}^{L-1} A^i = \frac{1-A^L}{1-A}$, $A \neq 1$,

$$j \mapsto \frac{1}{2}\left[-1+(2k+1)^L\right] + j\,(2k+1)^L \tag{5.19}$$

Likewise, after L iterations of the map $\bar{j} \mapsto \overline{(2k+1)\,j}$, we get

$$\bar{j} \mapsto \overline{j \prod_{i=0}^{L-1} (2k+1)} \tag{5.20}$$

$$\bar{j} \mapsto \overline{j\,(1+2k)^L} \tag{5.21}$$

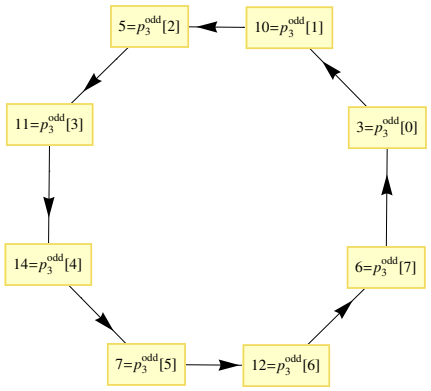


Figure 5.1. The circular structure relating all the odd powers of a primitive element α. In this case $p = 17$ and $\alpha = 3$.

To complete a cycle we want the coefficient of j to be congruent to 1 and the other term in (5.18) to be congruent to 0. The satisfaction condition is therefore $(1+2k)^L \equiv 1$ giving the order $L \mid \phi(p-1)$. □

By the previous lemma and since $\phi(p-1)$ is even, there are permutations ρ and ρ' with orders $\frac{p-1}{4}$ and 2, respectively. Therefore the successive application of the permutation ρ to $h_\alpha(x)$ and $\rho'(h_\alpha(x))$ generates the set of all basic TCH codewords.

Example: Consider the case for $p = 17$. There are $\phi(\phi(p))$ primitive elements of $\mathbb{F}_p^*$, where $\phi(\cdot)$ is the Euler-Phi or totient function. Take $\alpha = 3$ as a primitive element. The set of non quadratic residues (in this case also the set of primitive elements since p is a Fermat prime) is, $\overline{Q_R} = \{3^1, 3^3, 3^5, 3^7, 3^9, 3^{11}, 3^{13}, 3^{15}\} = \{3,10,5,11,14,7,12,6\}$ as illustrated in Figure 5.1. The set of quadratic residues is $Q_R = \{3^0, 3^2, 3^4, 3^6, 3^8, 3^{10}, 3^{12}, 3^{14}\} = \{1,9,13,15,16,8,4,2\}$.

The remaining sets are $\Sigma = \{11,6,12,7\}$, $\overline{\Sigma} = \{4,15,8,13\}$, $\Theta(3) = \{3,5,6,7\}$ and $\overline{\Theta}(3) = \{2,3,5,6\}$. Therefore, $h_3(x) = x^7 + x^{11} + x^{13} + x^{15} + x^4 + x^6 + x^{10} + x^{12}$ represented in binary by the codeword $H_3 = 1011\,1100\,1101\,0000$ (where the least significant bit is at the rightmost position) or in hexadecimal format by BCD0. The divisors of $\phi(17-1) = 8$ are 1, 2 and 4. These are the possible orders for the permutations π. We select $k = 1$ since $\text{ord}_{16}(1+2\cdot 1) = 4$. The new primitive element is $\beta = \alpha^3 = 10$. With $k = 1$ the permutations $\overline{\sigma}$ and σ are:

$$\overline{\sigma} = \begin{pmatrix} \overline{0} & \overline{1} & \overline{2} & \overline{3} & \overline{4} & \overline{5} & \overline{6} & \overline{7} \\ \overline{0} & \overline{3} & \overline{6} & \overline{1} & \overline{4} & \overline{7} & \overline{2} & \overline{5} \end{pmatrix}$$

$$\sigma = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 4 & 7 & 2 & 5 & 0 & 3 & 6 \end{pmatrix}$$

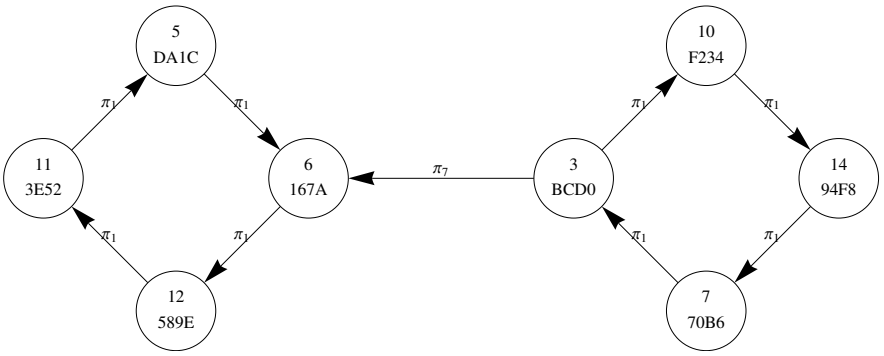


Figure 5.2. Digraph of all TCH basic codewords for $p = 17$. Each vertex contains the primitive element α_i and the hexadecimal representation of its associated 16 bit codeword $H_{\alpha_i}(x)$. Using just two permutations, π_1 with order 4, π_7 with order 2, all $\phi(17-1) = 8$ codewords are generated from a single one.

Thus the intertwined permutation π_1 is

$$\begin{pmatrix} 7 & \overline{7} & 6 & \overline{6} & 5 & \overline{5} & 4 & \overline{4} & 3 & \overline{3} & 2 & \overline{2} & 1 & \overline{1} & 0 & \overline{0} \\ 6 & \overline{5} & 3 & \overline{2} & 0 & \overline{7} & 5 & \overline{4} & 2 & \overline{1} & 7 & \overline{6} & 4 & \overline{3} & 1 & \overline{0} \end{pmatrix}$$

which applied to H_3 gives the new codeword $H_{10} \Leftrightarrow$ F234

	7	$\overline{7}$	6	$\overline{6}$	5	$\overline{5}$	4	$\overline{4}$	3	$\overline{3}$	2	$\overline{2}$	1	$\overline{1}$	0	$\overline{0}$
BCD0	1	0	1	1	1	1	0	0	1	1	0	1	0	0	0	0
F234	1	1	1	1	0	0	1	0	0	0	1	1	0	1	0	0
	6	$\overline{5}$	3	$\overline{2}$	0	$\overline{7}$	5	$\overline{4}$	2	$\overline{1}$	7	$\overline{6}$	4	$\overline{3}$	1	$\overline{0}$

Now consider the power recursion sequence, 3^3, $\left(3^3\right)^3, \cdots$, giving $10, 14, 7, 3$. Note that at each step k is fixed at $k = K = 1$ and therefore the successive application of the above permutation generate half of all codewords as depicted in the rightmost circle of figure 5.2. To get the other half we need to consider another primitive element for the recursion sequence, e.g. 6^3, $\left(6^3\right)^3, \cdots$, giving $12, 11, 5, 6$. Since $6 = 3^{15}$ we have $k = 7$ and the permutation π_7 to use to get H_6 from H_3 is

$$\begin{pmatrix} 7 & \overline{7} & 6 & \overline{6} & 5 & \overline{5} & 4 & \overline{4} & 3 & \overline{3} & 2 & \overline{2} & 1 & \overline{1} & 0 & \overline{0} \\ 0 & \overline{1} & 1 & \overline{2} & 2 & \overline{3} & 3 & \overline{4} & 4 & \overline{5} & 5 & \overline{6} & 6 & \overline{7} & 7 & \overline{0} \end{pmatrix}$$

Note that this permutation corresponds to a composition of two bit-wise operations, a reflection around the middle axis and then a single bit rotation to the left. The complete solution, with permutations π_1 and π_7 is depicted in Figure 5.2.

5.4. Extension to generalized TCH codes

The original generating equation (5.1) allows the determination of TCH codewords of length 16, 256 and 65536. For most applications codewords of intermediate length, i.e. 32, 64, 128, 512, 1024 etc., are also necessary and useful. In this section we lift the restriction on p being a Fermat prime.

Instead of considering primitive elements we only look at the exponents g of a certain primitive element power α^g. Let G=$\{g : \gcd(g, p-1) = 1\}$ be a multiplicative subgroup of $\mathbb{F}_p^*$. We write G as the disjoint union of H and gH,

$$G = H \dot{\cup} gH \tag{5.22}$$

where $H = \left\{g \in G : g^2 \equiv 1 \pmod{p-1}\right\}$ is a subgroup of G with a set of elements of order 2. Note that the order of g is the order $|<g>|$ of this cyclic subgroup. It follows that gH is a left coset of H for $g \in$ G/H collecting elements of higher order. The number of cosets is $[G : H]$ and when $G = H$ the group is simple with all elements having order 2. The structure of G is therefore isomorphic to $C_2 \times C_{\frac{|G|}{2}}$. Therefore to select the best permutation we choose a primitive element with the highest order. These insights from group theory are exploited in the next section.

Another way to look at G is to consider a subset of all odd integers as follows. The number of all odd integers between 1 and $p-1$ is equal to $N_{odd} = \frac{p-1}{2}$. For primes of the form $p = 4k+1$ and excluding the Fermat primes ($p = 2^{2^m}+1$) the set G can be partioned into a set with 4 blocks of elements, each block with b elements, where $b = \frac{N_{odd}-2}{4} = \frac{p-5}{8}$. Starting from a full set of odd integers, constructing G is as easy as taking the first b odd integers, skipping one integer, taking the next $2b$ odd integers, skipping one integer and taking the remaining b odd integers. As an example, for $p = 13$, the starting full set of odd integers is $\{1,3,5,7,9,11\}$. Now $b = (13-5)/8 = 1$ and the set G is obtained by taking one integer, skipping one, taking the next two, skipping one and taking the last one, resulting in $\{1, \not{3}, 5, 7, \not{9}, 11\}$. For p a Fermat prime, $b = \frac{p-1}{8}$ and there is no skipping, i.e. G is the full set of odd integers.

5.5. Useful isomorphisms

The set of primitive elements of $\mathbb{F}_p$ is constituted by all the elements of the form α^k with α a fixed primitive element and $k \in M_{p-1}$, where M_{p-1} denotes the set of positive integers inferior to $p-1$ that are co-prime to $p-1$. The set M_{p-1} with multiplication modulo $p-1$ is a multiplicative group called the *prime residue group* associated to $p-1$. The order of this group is given by the *Euler function* $\phi(p-1)$ and its exponent, i.e., the smallest positive integer ℓ such that $k^\ell \equiv 1 \pmod{(p-1)}$ for every $k \in M_{p-1}$, is given by the *Carmichael function* $\lambda(p-1)$.

Table 5.2. Isomorphisms between prime residue groups and the direct product of cyclic subgroups, for the first few primes.

p	$p \bmod 4$	$p-1$	$\simeq M_{p-1}$	$\phi(p-1)$	$\lambda(p-1)$
5	1	2^2	C_2	2	2
7	3	$2^1 \cdot 3^1$	C_2	2	2
11	3	$2^1 \cdot 5^1$	C_4	4	4
13	1	$2^2 \cdot 3^1$	$C_2 \times C_2$	4	2
17	1	2^4	$C_2 \times C_4$	8	4
19	3	$2^1 \cdot 3^2$	C_6	6	6
23	3	$2^1 \cdot 11^1$	C_{10}	10	10
29	1	$2^2 \cdot 7^1$	$C_2 \times C_6$	12	6
31	3	$2^1 \cdot 3^1 \cdot 5^1$	$C_2 \times C_4$	8	4
37	1	$2^2 \cdot 3^2$	$C_2 \times C_6$	12	6
41	1	$2^3 \cdot 5^1$	$C_2 \times C_2 \times C_4$	16	4

Since M_{p-1} is a finite abelian group, M_{p-1} is isomorphic to a direct product of s cyclic subgroups, C_{m_1}, C_{m_2}, ... C_{m_s} of orders $m_1, m_2, \ldots, m_s$, respectively. We may assume $m_1 \leq m_2 \leq \cdots \leq m_s = \lambda(p-1)$. These isomorphisms and the values of the Carmichael function are gathered in Table 5.2 for the first few primes. In this table the class residues of $p \pmod 4$ are also indicated. When $p \equiv 1 \pmod 4$, p is called a *Pythagorean prime*. If in addition, $p = 4s^t + 1$ for some positive integer t and prime number s, p is called a *sub-Pythagorean prime*. In this later case, the above isomorphism reduces to $M_{p-1} \simeq C_2 \times C_{\lambda(p-1)}$.

We refer to [**54, 55**] for further details.

5.6. Sequences associated to partitions of $\mathbb{F}_p^*$

In this section we introduce a class of sequences associated to balanced partitions of $\mathbb{F}_p^*$, and we derive a simple way to obtain these sequences, in time and frequency domains.

An ordered set partition $\mathscr{P} = (P_0, \ldots, P_{M-1})$ of $\mathbb{F}_p^*$ into M sets with the same cardinality is called a *balanced partition* of $\mathbb{F}_p^*$. Let α be a fixed primitive element of $\mathbb{F}_p$. For each k co-prime to $p-1$ we define an M-ary sequence of length $p-1$, f_{α^k}, which we call a $(\mathscr{P}, \alpha)$-*sequence* by

$$f_{\alpha^k}(t) = \ell \quad \text{if} \quad \alpha^{kt} \in P_{k\ell}, \tag{5.23}$$

where the exponent product is computed (mod $(p-1)$) and the subscript product is computed (mod M). The sequence f_{α^k} is well defined since M divides $p-1$ and k is co-prime to $p-1$. We denote by $S_{\mathscr{P},\alpha}$ the set $\{f_{\alpha^k} : k \in M_{p-1}\}$ and we call it a *complete set of* $(\mathscr{P}, \alpha)$-*sequences*.

For each k co-prime to $p-1$, let π_k (respectively σ_k) be the permutation of $\{0,\ldots,p-2\}$ (respectively $\{0,\ldots,M-1\}$) obtained by multiplication by k (mod $(p-1)$) (respectively (mod M).

PROPOSITION 85. *For M-ary $(\mathscr{P},\alpha)$-sequences of period $p-1$ and $k \in M_{p-1}$, we have*

$$f_{\alpha^k} = \sigma_{k^{-1}} \circ f_\alpha \circ \pi_k.$$

In particular, $S_{\mathscr{P},\alpha} = \{\sigma_{k^{-1}} \circ f_\alpha \circ \pi_k : k \in M_{p-1}\}$.

PROOF. For every $k \in M_{p-1}$ and $\ell \in \{0,\ldots M-1\}$, we have

$$\begin{aligned} f_{\alpha^k}(t) = \ell &\Leftrightarrow \alpha^{kt} \in P_{k\ell} \\ &\Leftrightarrow f_\alpha(kt) = k\ell \pmod{M} \\ &\Leftrightarrow k^{-1} f_\alpha(kt) = \ell \pmod{M} \end{aligned}$$

and thus $f_{\alpha^k} = \sigma_{k^{-1}} \circ f_\alpha \circ \pi_k$. □

For each $k \in M_{p-1}$ we shall denote by $T_k : \mathscr{S}_{\mathscr{P},\alpha} \to \mathscr{S}_{\mathscr{P},\alpha}$ the map defined by $T_k(f) = \sigma_{k^{-1}} \circ f \circ \pi_k$.

Note that $T_k \circ T_j = T_j \circ T_k$ for every $j,k \in M_{p-1}$.

By the isomorphism $M_{p-1} \simeq C_{m_1} \times C_{m_2} \times \cdots \times C_{m_s}$ (referred in Section 5.5), M_{p-1} is generated by elements $a_1,\ldots,a_s \in M_{p-1}$, of orders $\mathrm{ord}_{p-1}(a_i) = m_i$, $i = 1,\ldots,s$. The generators $a_1,\ldots,a_s$ can be obtained using one of various standard computer algebra systems, e.g. GAP [**56**]. This isomorphism along with Proposition 85 yields the following result.

COROLLARY 86. *Every sequence in $S_{\mathscr{P},\alpha}$ can be obtained from the "initial sequence" f_α by applying the maps T_{a_i}, $i = 1,\ldots,s$, a suitable number of times.*

In communication applications, correlating sequences of length above a certain threshold (typically 100) takes less time if the operation is performed in the frequency domain rather than in the time domain. In the time domain, the cross-correlation between two $(p-1)$-periodic sequences $f[n]$ and $g[n]$ is given by $(f \star g)[n] \doteq \sum_{m=0}^{p-2} f^*[m]\, g[n+m]$ where $f^*[m]$ denotes the complex conjugation of $f[m]$. In the frequency domain, the cross-correlation satisfies $\mathscr{F}\{f \star g\} = (\mathscr{F}\{f\})^* \mathscr{F}\{g\}$, where $\mathscr{F}\{f\}$ denotes the *Discrete Fourier Transform* (DFT) of a sequence f of period $p-1$, which is defined by

$$\mathscr{F}\{f\}(t) \doteq \frac{1}{p-1} \sum_{n=0}^{p-2} f[n]\, e^{\frac{-2\pi j}{p-1} tn}, \quad t = 0,\ldots,p-2,$$

with $j = \sqrt{-1}$.

For the DFT of a $(\mathscr{P},\alpha)$-sequence the following proposition is the analogous to Proposition 85.

PROPOSITION 87. *For M-ary $(\mathscr{P},\alpha)$-sequences of length $p-1$ and $k \in M_{p-1}$, we have*

$$\mathscr{F}\{f_{\alpha^k}\} = \mathscr{F}\{\sigma_{k^{-1}} \circ f_\alpha\} \circ \pi_{k^{-1}}.$$

In particular, the DFT of the set $S_{\mathscr{P},\alpha}$ is equal to

$$\{\mathscr{F}\{\sigma_{k^{-1}} \circ f_\alpha\} \circ \pi_{k^{-1}} : \quad k \in M_{p-1}\}.$$

PROOF. Using Proposition 85 and taking into account that the product by k of the subscripts $0,\ldots,p-2$ induces a permutation on this set of subscripts, we get

$$\begin{aligned}
\mathscr{F}\{f_{\alpha^k}\}(t) &= \frac{1}{p-1}\sum_{n=0}^{p-2} f_{\alpha^k}[n]\, e^{\frac{-2\pi j}{p-1}tn} \\
&= \frac{1}{p-1}\sum_{n=0}^{p-2} (k^{-1} f_\alpha[kn]) e^{\frac{-2\pi j}{p-1}(k^{-1}t)(kn)} \\
&= \frac{1}{p-1}\sum_{r=0}^{p-2} (\sigma_{k^{-1}} \circ f_\alpha)[r]\, e^{\frac{-2\pi j}{p-1}(k^{-1}t)r} \\
&= \mathscr{F}\{\sigma_{k^{-1}} \circ f_\alpha\}(k^{-1}t).
\end{aligned}$$

□

Remark that if $M = 2$ and $k \in M_{p-1}$, we get the identity permutation $\sigma_k = id$, since the elements of M_{p-1} are odd integers and thus congruent to 1 modulo 2. This fact along with Proposition 85 and Proposition 87 implies immediately the following result.

COROLLARY 88. *For $M = 2$, a complete set of $(\mathscr{P},\alpha)$-sequences can be generated, in the time domain, from the initial sequence f_α by suitable compositions of the permutations $\pi_{a_1},\ldots,\pi_{a_s}$. In the frequency domain, the spectra of this set can be generated from $\mathscr{F}\{f_\alpha\}$ by suitable compositions of the permutations $\pi_{a_1^{-1}},\ldots,\pi_{a_s^{-1}}$.*

In the next subsection we explain in detail how to apply Corollary 88 to compute the complete set of $(\mathscr{P},\alpha)$-sequences, when p is a sub-Pythagorean prime.

5.6.1. The case of sub-Pythagorean primes. When p is a sub-Pythagorean prime, $M_{p-1} \simeq C_2 \times C_{\lambda(p-1)}$ with $\lambda(p-1) = \frac{\phi(p-1)}{2}$ and there are elements $a_1, a_2 \in M_{p-1}$ such that $\mathrm{ord}_{p-1} a_1 =$

Table 5.3. Generators a_1 and a_2 for permutations of order 2 and $\lambda(p-1)$, respectively. The value a_i^{-1} corresponds to the modular multiplicative inverse of a_i.

$p = 4k+1$	α	$\simeq M_{p-1}$	$\phi(p-1)$	$a_1 = a_1^{-1}$	a_2, a_2^{-1}
5	2	C_2	2	–	3, 3
13	2	$C_2 \times C_2$	4	5	11, 11
17	3	$C_2 \times C_4$	8	15	5, 13
29	2	$C_2 \times C_6$	12	13	3, 19
37	2	$C_2 \times C_6$	12	17	11, 23
53	2	$C_2 \times C_{12}$	24	51	15, 7
101	2	$C_2 \times C_{20}$	40	99	27, 63
109	6	$C_2 \times C_{18}$	36	53	83, 95
149	2	$C_2 \times C_{36}$	72	147	39, 19
197	2	$C_2 \times C_{42}$	84	97	3, 131
257	3	$C_2 \times C_{64}$	128	255	5, 205
317	2	$C_2 \times C_{78}$	156	157	3, 211
509	2	$C_2 \times C_{126}$	252	253	3, 339
557	2	$C_2 \times C_{138}$	276	277	419, 487
797	2	$C_2 \times C_{198}$	396	397	3, 531
1109	2	$C_2 \times C_{276}$	552	1107	559, 111
65537	3	$C_2 \times C_{16384}$	32768	65535	5, 52429

2, $\mathrm{ord}_{p-1} a_2 = \lambda(p-1)$ with $(a_1) \cap (a_2) = (1)$. Moreover, since $\mathrm{ord}_{p-1} a_1 = 2$, $a_1^{-1} = a_1$. Thus we get immediately the following consequence of Corollary 88.

COROLLARY 89. *For p a sub-Pythagorean number and $M = 2$ the set of $(\mathscr{P}, \alpha)$-sequences corresponds to the set of sequences*

$$f_\alpha \circ \pi_{a_2^i}, \quad f_\alpha \circ \pi_{a_1} \circ \pi_{a_2^i},$$

and its spectra to the set of sequences

$$\mathscr{F}\{f_\alpha\} \circ \pi_{a_2^{-i}}, \quad \mathscr{F}\{f_\alpha\} \circ \pi_{a_1} \circ \pi_{a_2^{-i}},$$

where i ranges through the set of integers $0, \ldots, \lambda(p-1)-1$.

The generators a_1, a_2 are presented in Table 5.3 for some chosen sub-Pythagorean primes. Note that when p is a Fermat prime greater than 5, $\lambda(p-1) = \frac{p-1}{2} = 2^t$ with $t \geq 3$ and we can choose $a_1 = p-2$ and $a_2 = 5$. Actually $(p-2)^2 = (-1)^2 = 1 \pmod{(p-1)}$ and it is a well known result that $\mathrm{ord}_{p-1} 5 = \mathrm{ord}_{2^t} 5 = \phi(2^t)/2$.

5.7. Application to Sidel'nikov sequences

For M dividing p^n-1, where p is a prime number and n is a positive integer, Sidel'nikov [**57**] introduced a class of M-ary sequences (called Sidel'nikov sequences or SN-sequences) of period p^n-1, for which the out-of-phase auto-correlation magnitude is upper-bounded by 4. A recent example of a low complexity implementation of M-ary SN-sequences of period $p^{2n}-1$ is presented in [**58**]. In this section we address the fast generation of SN-sequences of period $p-1$.

Later, Lempel, Cohn and Eastman [**59**] rediscovered the binary Sidel'nikov sequences of period p^n-1. Lempel/Cohn/Eastman construction results in the Boolean-wise negation of the corresponding Sidel'nikov sequence [**60**]. More recently, Tomlinson, Cercas, and Hughes [**5**], presented a class of binary sequences (called TCH sequences) of period $p-1$ with p a Fermat prime. We show in a following section that TCH sequences are a small subset of binary SN-sequences.

We keep the notations of the previous section.

Let $M \geq 2$ be an integer which divides $q-1$ with $q=p^n$. For each primitive element α of $\mathbb{F}_q$, consider the partition of $\mathbb{F}_q^*$, $\mathscr{S}^{(\alpha)} = \left(S_\ell^{(\alpha)}\right)$, defined by

$$\begin{aligned} S_\ell^{(\alpha)} &= \left\{ \alpha^{Mi+\ell} - 1 \mid 0 \leq i < \frac{q-1}{M} \right\}, 1 \leq \ell \leq M-1, \\ S_0^{(\alpha)} &= \left\{ \alpha^{Mi} - 1 \mid 0 < i < \frac{q-1}{M} \right\} \cup \left\{ \alpha^{\frac{q-1}{2}} \right\}. \end{aligned}$$

The M-ary *Sidel'nikov (SN) sequence* of period $q-1$ associated to the primitive element α, is the sequence $g_\alpha = (g_\alpha(t))$, $t \in \{0,\ldots,q-2\}$, defined by

$$g_\alpha(t) = \ell \qquad \text{if} \qquad \alpha^t \in S_\ell^{(\alpha)}.$$

In the sequel we assume $q=p$ and we consider a fixed primitive element α of $\mathbb{F}_p$. Let $\mathscr{P} = (P_\ell)$ be the ordered set partition of $\mathbb{F}_p^*$ given by $P_\ell = S_\ell^{(\alpha)}$, $\ell \in \{0,\ldots M-1\}$.

PROPOSITION 90. *For each $k \in M_{p-1}$, the SN-sequence g_{α^k}, associated to the primitive element α^k is a $(\mathscr{P},\alpha)$-sequence.*

PROOF. By definition $g_{\alpha^k}(t) = \ell$ if $\alpha^{kt} \in S_\ell^{(\alpha^k)}$. It remains to be proved that for every $k \in M_{p-1}$ and $\ell \in \{0,\ldots M-1\}$,

$$S_\ell^{(\alpha^k)} = P_{k\ell}. \tag{5.24}$$

Assume in the first place that $\ell \neq 0$. Since k and M are co-prime integers, the multiplication by k (mod $(p-1)$) gives rise to a permutation of the set $\{Mi : i = 0,\ldots,\frac{p-1}{M}-1\}$ and thus we

obtain

$$\begin{aligned} S_\ell^{(\alpha^k)} + 1 &= \left\{ \alpha^{(Mi+\ell)k} \mid i = 0, \ldots, \frac{p-1}{M} - 1 \right\} \\ &= \left\{ \alpha^{Mik} \cdot \alpha^{\ell k} \mid i = 0, \ldots, \frac{p-1}{M} - 1 \right\} \\ &= \left\{ \alpha^{Mik} \mid i = 0, \ldots, \frac{p-1}{M} - 1 \right\} \cdot \alpha^{\ell k} \\ &= \left\{ \alpha^{Mi} \mid i = 0, \ldots, \frac{p-1}{M} - 1 \right\} \cdot \alpha^{\ell k} \\ &= \left\{ \alpha^{Mi+k\ell} \mid i = 0, \ldots, \frac{p-1}{M} - 1 \right\} \\ &= S_{k\ell}^{(\alpha)} + 1. \end{aligned}$$

If ℓ vanishes, reminding that the multiplication by $k \pmod{(p-1)}$ gives rise to a permutation of the set $\{Mi : i = 1, \ldots, \frac{p-1}{M} - 1\}$ and $\alpha^{\frac{p-1}{2}}$ is congruent to $-1 \pmod{p}$, we have

$$\begin{aligned} S_0^{(\alpha^k)} + 1 &= \left\{ \alpha^{Mik} \mid i = 1, \ldots, \frac{p-1}{M} - 1 \right\} \cup \{0\} \\ &= \left\{ \alpha^{Mi} \mid i = 1, \ldots, \frac{p-1}{M} - 1 \right\} \cup \{0\} \\ &= S_0^{(\alpha)} + 1. \end{aligned}$$

Hence $S_\ell^{(\alpha^k)} = S_{k\ell}^{(\alpha)} = P_{k\ell}$ for every $\ell = 0, \ldots, M-1$ and $k \in M_{p-1}$.

□

5.8. TCH codewords are a subset of SN sequences

TCH sequences are balanced binary sequences of period $p-1$, with p a Fermat prime. This class of sequences was introduced in [**5**]. For each primitive element α of $\mathbb{F}_p$, the TCH sequence associated to α was defined, in polynomial form, by

$$h_\alpha(x) = \sum_{i=0}^{\frac{p-3}{2}} x^{K_i} \in \mathbb{Z}_2[x], \tag{5.25}$$

where the exponents K_i satisfy the *generating equation*,

$$\alpha^{K_i} \equiv 1 + \alpha^{2i+1} \pmod{p} \quad i = 0, 1, \ldots, \frac{p-1}{2} - 1.$$

PROPOSITION 91. *The TCH sequence associated to a primitive element of $\mathbb{F}_p$ is a half-period rotation of the binary SN-sequence associated to the same primitive element.*

PROOF. Let QR be the set of *quadratic residues* of $\mathbb{F}_p$ and $\mathrm{NQR} = \mathbb{F}_p^* \setminus \mathrm{QR}$ the set of *quadratic non-residues*. It is well known that

$$\mathrm{QR} = \{\alpha^0, \alpha^2, \ldots, \alpha^{p-3}\},$$

and

$$\mathrm{NQR} = \{\alpha^1, \alpha^3, \ldots \alpha^{p-2}\}.$$

Remarking that the TCH sequence (5.25) can also be defined, in the binary form, as

$$h_\alpha(t) = \begin{cases} 1, & \text{if } \alpha^t \in 1 + \mathrm{NQR} \\ 0, & \text{if } \alpha^t \notin 1 + \mathrm{NQR} \end{cases} \qquad 0 \le t \le p-2,$$

and reminding that $\alpha^{\frac{p-1}{2}} \equiv -1 \pmod{p}$ with $\frac{p-1}{2}$ an even integer, we obtain

$$\begin{aligned} h_\alpha(t) = 1 &\Leftrightarrow \exists i,\ \alpha^t = \alpha^{2i+1} + 1 \\ &\Leftrightarrow \exists i,\ \alpha^t = \alpha^{2i+1} + \alpha^{p-1} \\ &\Leftrightarrow \exists i,\ \alpha^t = \alpha^{\frac{p-1}{2}}(\alpha^{2i+1-\frac{p-1}{2}} - 1) \\ &\Leftrightarrow \exists j,\ \alpha^{t+\frac{1-p}{2}} = \alpha^{2j+1} - 1 \\ &\Leftrightarrow \alpha^{t+\frac{1-p}{2}} \in S_1^{(\alpha)}. \end{aligned}$$

□

5.9. Demonstration examples

In this section we apply the results of the previous sections to generate the SN-sequences associated to some sub-Pythagorean primes. The examples considered in this section may nonetheless be easily extended to generate the SN-sequences associated to an arbitrary sub-Pythagorean prime.

By Proposition 90 the set of SN-sequences is a complete set of $(\mathscr{P}, \alpha)$-sequences for a fixed, but arbitrary, primitive element $\alpha \in \mathbb{F}_p$. By taking α as the smallest primitive element we consider $\alpha = 3$ if p is equal to 17, 257 or 65537, $\alpha = 6$ if $p = 109$ and $\alpha = 2$ for the remaining sub-Pythagorean primes p listed in Table 5.3.

5.9.1. Binary SN-sequences associated to the Fermat primes 17, 257 and 65537. For coding applications the best code rates (proportional to $\phi(p-1)/(p-1)$) are obtained when $p = 2^{2^m} + 1$, that is, when p is a *Fermat prime*. The list of known Fermat primes is $3, 5, 17, 257, 65537$. This class of sequences was independently presented in [**5, 6**] as *TCH-sequences*.

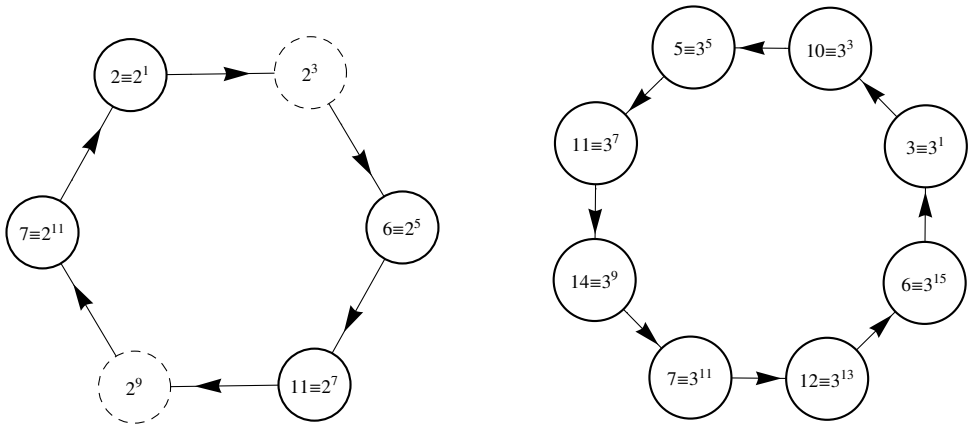


Figure 5.3. The circular structure relating the k^{th} powers of a primitive element α_0 with $\alpha = 2$, $p = 13$ (left) and $\alpha = 3$, $p = 17$ (right). Note that in the figure on the left $3, 9 \notin M_{12}$ while in the figure on the right (corresponding to a Fermat prime case) all odd powers of α are primitive elements.

We start with $p = 17$ and $\alpha = 3$. Since p is a Fermat prime each odd power of α gives rise to a primitive element of $\mathbb{F}_p$ (See Figure 5.3). Moreover, the set of all such primitive elements correspond to the set of *quadratic non-residues*, NQR $= \{3^1, 3^3, 3^5, 3^7, 3^9, 3^{11}, 3^{13}, 3^{15}\}$ $= \{3, 10, 5, 11, 14, 7, 12, 6\}$.

The initial primitive element $\alpha_0 = \alpha = 3$ gives rise to the binary sequence $f_3 = 1101\,0000\,1011\,1100$ (with the least significant bit at the rightmost position). In hexadecimal notation we shall write 0xD0BC. The permutation $\pi_5(j) = 5j \pmod{16}$ has order 4 since $\text{ord}_{16}(5) = 4$. As illustrated in Figure 5.4, by successive application of the permutation π_5, we obtain from f_3 the sequences associated to the primitive elements $\alpha_1 = \alpha_0^5$, $\alpha_2 = \alpha_1^5$, and $\alpha_3 = \alpha_2^5$, i.e., the sequences $f_5 = 0$x1CDA, $f_{14} = 0$xF894 and $f_{12} = 0$x9E58, respectively. To obtain the remaining 4 sequences we apply the permutation π_{15} to the sequence f_3, which yields $f_6 = 0$x7A16, followed by the permutations π_5^i, $i = 1, 2, 3$, which yield the sequences $f_7 = 0$xB670, $f_{11} = 0$x523E and $f_{10} = 0$x34F2.

To get the spectra of this set we proceed as above with the sequence $\mathscr{F}\{f_3\}$ in the place of f_3 and we use the permutation $\pi_{13} = \pi_{5^{-1}}$ instead of the permutation π_5.

For the remaining cases $p = 257$ and $p = 65537$ we can also choose $\alpha = 3$, $a_1 = p - 1$ and $a_2 = 5$ and proceed as above. Note that the structure of the prime residue group M_{p-1} for these two primes is similar to the structure depicted in Figure 5.4, corresponding to $C_2 \times C_{64}$ in the former case and to $C_2 \times C_{16384}$ in the later case.

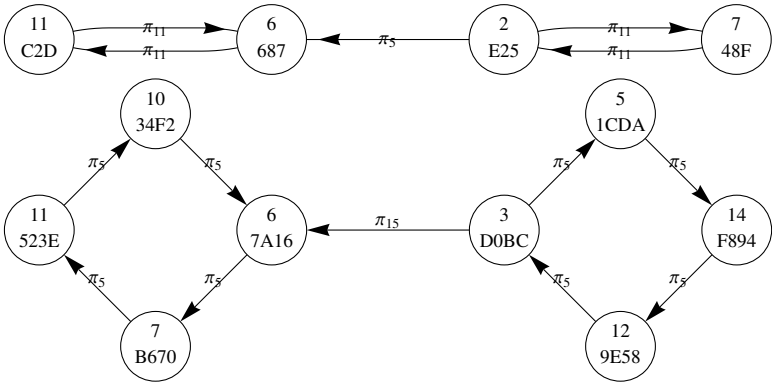


Figure 5.4. Digraph of all binary SN-sequences for the sub-Pythagorean primes 13 and 17. Each node contains a primitive element α^k and the corresponding SN-sequence $f_{\alpha^k}(t)$ (written in hexadecimal base). The edge labels indicate the permutations used to transform one sequence into another. In the top figure we have the $C_2 \times C_2$ structure for $p = 13$, while in the bottom figure we have the $C_2 \times C_4$ structure for $p = 17$. We consider the initial sequences to be associated with the lowest primitive elements ($f_2 = 0$xE25 in the top case and $f_3 = 0$xD0BC in the bottom case).

5.9.2. Binary SN-sequences for the sub-Pythagorean prime 13. For $p = 13$ and $\alpha = 2$ we obtain the binary sequences, written in the hexadecimal base,

$$f_2 = 0\text{xE25},$$
$$f_7 = f_2 \circ \pi_{11} = 0\text{x48F},$$
$$f_6 = f_2 \circ \pi_5 = 0\text{x687},$$
$$f_{11} = f_2 \circ \pi_5 \circ \pi_{11} = 0\text{xC2D}.$$

(See figures 5.3 and 5.4).

5.9.3. M-ary SN-sequences for the sub-Pythagorean prime 13. For a general prime p let

$$D_\ell^{(\alpha)} = \left\{ \alpha^{Mi+\ell} \mid 0 \le i \le \frac{p-1}{M} - 1 \right\}, \quad 0 \le \ell \le M-1,$$

be the cyclotomic classes of $\mathbb{F}_p$ of order M. It is easy to prove that $D_\ell^{(\alpha^k)} = \alpha^{k\ell} D_0^{(\alpha)}$ holds for every $k \in M_{p-1}$. Therefore, once $D_0^{(\alpha)}$ is determined, all the remaining cyclotomic classes are easily computed. In terms of the SN-partition $\mathscr{S}^{(\alpha)}$, since $S_0^{(\alpha)}$ corresponds to the set $D_0^{(\alpha)} - 1$ with the element 0 replaced by -1 and $S_\ell^{(\alpha)} = D_\ell^{(\alpha)} - 1$ if $\ell \neq 0$, we conclude that $\mathscr{S}^{(\alpha)}$ is essentially determined by the set $S_0^{(\alpha)}$.

Applying these considerations when $p = 13$, $M = 3$ and $\alpha = 2$, we get $D_0^{(2)} = \{1,8,12,5\}$, $D_1^{(2)} = 2D_0^{(2)}$ and $D_2 = 2^2 D_0^{(2)}$, which yields $S_0^{(2)} = \{12,7,11,4\}$, $S_1^{(2)} = \{1,2,10,9\}$ and $S_2^{(2)} = \{3,5,8,6\}$. This partition gives rise to the ternary sequence $f_2 = 012100222011$.

Applying Proposition 85 to the initial sequence f_2 and taking into account that 5^{-1} and 11^{-1} are both congruent to 2 modulo 3, we obtain the remaining ternary sequences (here represented in base 27 for convenience),

$$\begin{aligned} f_2 &= 59\mathrm{Q}4_{27}, \\ f_6 &= 5^{-1} f_2 \circ \pi_5 = 2 f_2 \circ \pi_5 = 19\mathrm{PN}_{27}, \\ f_7 &= 11^{-1} f_2 \circ \pi_{11} = 2 f_2 \circ \pi_{11} = \mathrm{JC}74_{27}, \\ f_{11} &= 11^{-1} f_6 \circ \pi_{11} = f_2 \circ \pi_5 \circ \pi_{11} = \mathrm{NC}81_{27}. \end{aligned}$$

5.9.4. Composite sequences for non sub-Pythagorean primes. The construction of SN-sequences generates words of period $n = p-1$ with p a prime. However, for several applications sequences whose period is equal to 32, 64, 128, 512, 1024, (which are a power of two but their sum with 1 is not a prime number), are also of interest. In order to obtain sequences of such periods, one can use sequences of a larger period and delete the excess coordinates *a posteriori*, or one can compose two sequences, say $\mathscr{C}_1$ of period $n_1 = p_1 - 1$ and $\mathscr{C}_2$ of period $n_2 = p_2 - 1$ to obtain a sequence of length $n = n_1 + n_2$. Two standard constructions [**61**] are often used: the *direct sum construction*,

$$\mathscr{C}_1 \oplus \mathscr{C}_2 = \{\mathbf{xy} \mid \mathbf{x} \in \mathscr{C}_1, \mathbf{y} \in \mathscr{C}_2\}, n_1 \neq n_2$$

and the $(u, u+v)$-*construction*,

$$\{\mathbf{x}(\mathbf{x}+\mathbf{y}) \mid \mathbf{x} \in \mathscr{C}_1, \mathbf{y} \in \mathscr{C}_2\}, n_1 = n_2.$$

We depict in the graph of Figure 5.5 all pairs $(p_1,\ p_2)$ of sub-Pythagorean primes p_i, $i = 1,2$, for which $n = p_1 + p_2 - 2$ is a power of two not greater than 512. There are a lot more prime pairs involved if the desired value of n is 1024.

5.10. Extension to $q = p^n$

The results presented in section 5.7 can be extended to more general primes. From here on, p is an odd prime number, n is a positive integer, $q = p^n$, $N = q-1$, M is a positive divisor of N and $\mathbb{F}_q = \mathrm{GF}(q)$ denotes the Galois field of order q. A primitive (field) element of $\mathbb{F}_q$ is a generator of the cyclic group $\mathbb{F}_q^*$. It is well known that the set of primitive elements of $\mathbb{F}_q$ consists of the elements of the form α^k, where α is a fixed primitive element of $\mathbb{F}_q$ and k runs through the prime residue group M_N associated to N.

Let S_0 be the set of M-th power residues modulo q. The Sidel'nikov (SN) sequence of length N associated to a primitive element α, called Sidel'nikov sequence of type 2 in [**62**], is defined

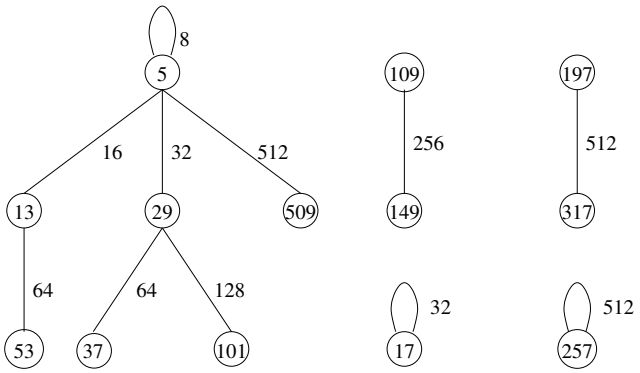


Figure 5.5. Undirected weighted graph where each vertex represents a sub-Pythagorean prime. A vertex p_1 is connected to a vertex p_2 by an edge of weight $n = p_1 + p_2 - 2$ such that the composite sequence length n is a power of 2 not exceeding 512. Note that in this setting the Fermat primes correspond to the vertices with self-loop edges.

as

$$g_\alpha(t) = \begin{cases} 0 & \text{if} \quad t = \frac{N}{2}, \\ \ell & \text{if} \quad \alpha^t + 1 \in \alpha^\ell S_0, \end{cases} \qquad t = 0, \ldots N-1.$$

The SN sequences are related with each other via permutations by

$$g_{\alpha^k} = \sigma^M_{k^{-1}} \circ g_\alpha \circ \sigma^N_k, \quad k \in M_N \tag{5.26}$$

where σ^r_k denotes the permutation of the set $\{0, \ldots, r-1\}$ given by multiplication by k modulo r assuming r and k coprime integers. Actually we have for every $k \in M_N$ and $t \neq \frac{N}{2}$,

$$g_{\alpha^k}(t) = \ell \text{ if } (\alpha^k)^t + 1 \in (\alpha^k)^\ell S_0 \qquad \Leftrightarrow \qquad g_\alpha(kt) = k\ell \text{ if } \alpha^{kt} + 1 \in \alpha^{k\ell} S_0.$$

Moreover, since M_N is a finite abelian group, M_N is isomorphic to a direct product of finite cyclic groups and we can find a minimal set of pairwise commuting elements $a_1, \ldots, a_s \in M_N$, which generate the group M_N. From this fact along with (5.26) and taking into account that $\sigma^r_k \sigma^r_s = \sigma^r_s \sigma^r_k$ for all k, s coprime to r we obtain the complete set of M-ary Sidel'nikov sequences from the seed sequence g_α by considering the compositions,

$$\sigma^M_{a_s^{-i_s}} \circ \cdots \circ \sigma^M_{a_1^{-i_1}} \circ g_\alpha \circ \sigma^N_{a_1^{i_1}} \circ \cdots \circ \sigma^N_{a_s^{i_s}}, \quad i_k = 0, \ldots, m_k - 1, \; k = 1, \ldots, s, \tag{5.27}$$

where m_k denotes the order of the element a_k in M_N and $\circ$ the usual composition of maps. In particular, for the case of binary sequences ($M = 2$), $\sigma^2_{k^{-1}} = \mathrm{id}$ for every $k \in M_N$, and the complete set of binary Sidel'nikov sequences is given by

$$g_\alpha \circ \sigma^N_{a_1^{i_1}} \circ \cdots \circ \sigma^N_{a_s^{i_s}}, \quad i_k = 0, \ldots, m_k - 1, \; k = 1, \ldots, s. \tag{5.28}$$

Let $\mathscr{F}\{g\}$ denote the DFT of the sequence g. Using (5.26) and taking into account that the multiplication by k modulo N induces a permutation on the set $\{0,\ldots,N-1\}$, we get

$$\begin{aligned}
\mathscr{F}\{g_{\alpha^k}\}(t) &= \frac{1}{N}\sum_{n=0}^{N-1} g_{\alpha^k}(n)e^{\frac{-2\pi j}{N}tn} \\
&= \frac{1}{N}\sum_{n=0}^{N-1} (k^{-1}g_\alpha(kn))e^{\frac{-2\pi j}{N}(k^{-1}t)(kn)} \\
&= \frac{1}{N}\sum_{r=0}^{N-1} (\sigma_{k^{-1}}^M \circ g_\alpha)(r)e^{\frac{-2\pi j}{N}(k^{-1}t)r} \\
&= \mathscr{F}\{\sigma_{k^{-1}}^M \circ g_\alpha\}\circ\sigma_{k^{-1}}^N(t).
\end{aligned}$$

The analogue of (5.26) in frequency domain is therefore given by

$$\mathscr{F}\{g_{\alpha^k}\} = \mathscr{F}\{\sigma_{k^{-1}}^M \circ g_\alpha\}\circ\sigma_{k^{-1}}^N, \quad k \in M_N \tag{5.29}$$

and obvious analogues of (5.27) and (5.28) also hold in the frequency domain.

The sequences spectra is particularly useful for the implementation of a maximum likelihood sequence detector exploiting cyclic correlations computed in the frequency domain. The time-domain cross-correlation between two sequences u and v, i.e., $(u\star v)$, can be computed by complex vector multiplication in the frequency domain, i.e., $N\overline{\mathscr{F}\{u\}}\mathscr{F}\{v\}$, followed by an inverse DFT (here the bar denotes complex conjugation). Assuming the input sequence u to be a noisy and circular shifted version of g_α and v to be an instance of g_{α^k}, all the necessary correlations for estimating u can be obtained via the computation $N\overline{\mathscr{F}\{u\}}(\mathscr{F}\{\sigma_{k^{-1}}^M \circ g_\alpha\}\circ\sigma_{k^{-1}}^N)$ for appropriate values of k.

5.11. Sequence detector

In some of the examples given previously, we worked with a subset of binary SN-sequences for which q is a Pythagorean prime of the form $4s^t+1$ for some positive integer t and prime number s. In this case, the group M_N is isomorphic to the direct product of cyclic groups $C_2 \times C_{\lambda(N)}$ where $\lambda(\cdot)$ is the Carmichael function [**37**].

This structure is used advantageously in the implementation of a maximum likelihood sequence detector as illustrated in Figure 5.6. A N-point complex FFT is used to obtain the spectra of two real N-point sequences, the seed sequence g_α which resides in memory and the input sequence u which we want to estimate. As previously described, the complete spectra set is obtained via permutations, iterating $r = 0,\ldots,\frac{N}{4}-1$. The real part of the IFFT block output contains the correlation of u with an instance of g_{α^k} and the imaginary part contains the

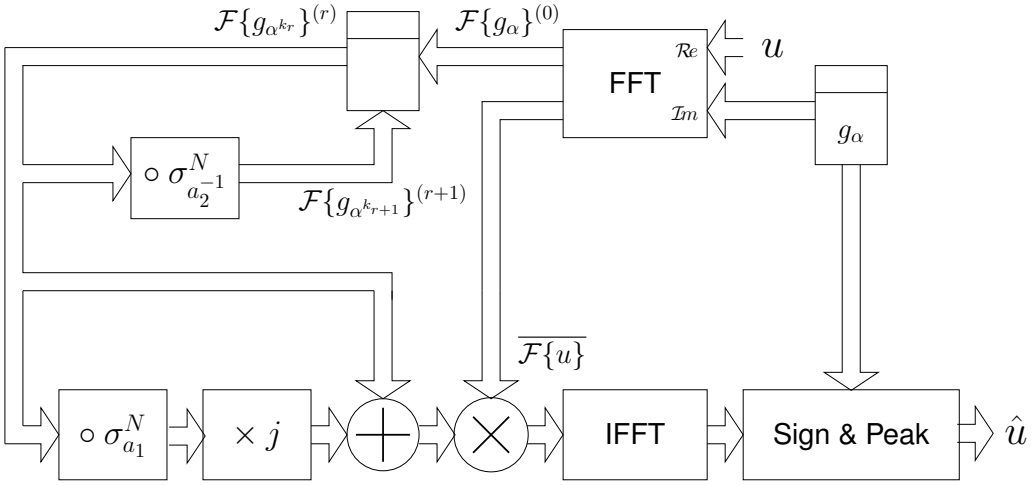


Figure 5.6. Optimized frequency-based maximum likelihood sequence detector.

correlation of u with the reflected instance, i.e. with $g_{\alpha^k} \circ \sigma^N_{N-1}$. The Sign & Peak block then determines which correlation produces the maximum peak value (the winning correlation) and the sign of the correlation peak indicates if a sequence or its logical complement was used, thus producing the estimate $\hat{u}$. A further decoding step can be done with the knowledge of the peak position in the winning correlation. With regard to the detector presented in [**7**] our proposal uses 3 times less FFT's and needs significantly less memory since only one seed sequence is required. This work was published in [**20**].

5.12. Conclusion

Pseudo-random sequences are important in synchronization applications and in multi-access techniques either to differentiate users or communication channels. One of the best classes of pseudo-random sequences is known as the class of Sidel'nikov sequences (over time also published under different names). When the generation of a large number of sequences is involved, an efficient way to obtain these sequences is important from an implementation point of view. In addition, the computation of the correlation between sequences is also necessary. However, if the length of these sequences is large this correlation is computationally more efficient if performed in the frequency domain rather than in the time domain, stressing the importance in obtaining also the spectra of the sequences. In this chapter we presented a fast and efficient method, based on group theory, to compute the set of all sequences and their spectra from a single sequence. For binary sequences associated to sub-Pythagorean primes the method only requires the repeated application of 3 permutations (two for the time domain and one extra for the frequency domain) and a DFT operation, thus saving memory space and processing time.

For general M-ary sequences the procedure may require, at most, M additional permutations. A set of demonstration examples and an engineering application are also included. The work of this chapter was published in [**20**] and accepted for publication in [**22**].

CHAPTER 6

APPLICATION OF SEQUENCES TO COMMUNICATIONS

In this chapter we describe three engineering applications where the TCH-type sequences are employed. In section 6.1 we simulate the bit error rate performance of some codes using both AWGN and Rayleigh channel models. In section 6.2 we proceed with a common problem of wireless communication systems: that of frame synchronization. We present the important variables in the context of CDMA systems, discuss the signal model and receptor types and then introduce the proposed frequency domain correlation technique as well the corresponding simulation results. A second application involves a spectra modified version of a TCH sequence exploited for channel estimation which we present in section 6.3. In this case the context is that of OFDM systems with frequency domain channel estimators. Finally, in section 6.4, we address the use of TCH sequences in Ultra Wide Band (UWB) systems. TCH codes are used as time-hopping codes in TH-PPM systems and as spreading codes in DS-PAM systems.

6.1. Code performance results

In this section the performance results are obtained for codes generated by the method explained in chapter 4. As an example it was considered codes with $R_c = 8/32$, $R_c = 10/64$ and $R_c = 9/128$.

On the other hand, it was assessed the performance of codes with $R_c = 7/64$, obtained by the method explained in chapter 5 comparing these results with the same code rate generated by [**6**].

The performance of these codes was obtained by simulation taking into consideration an AWGN channel [**63**] and a more realistic channel using the Rayleigh probability density functions (PDF) to model the occurrence of fading [**63**]. The rural and urban environments were used with the average depth fading, $E[\alpha] = 0$ dB and $E[\alpha] = -20$ dB , respectively. The Rayleigh

pdf is used in multipath environments. Some authors use this distribution to model the propagation channel, e.g., [**64, 65, 66, 67, 68, 69**].

For the simulation, the binary phase shift keying (BPSK) was adopted in the AWGN and Rayleigh channels where the error probability of channel, P_e is given by [**63**]:

$$P_e = \frac{1}{2}\text{erfc}\left(\sqrt{\frac{E_b}{N_o}}\right) \tag{6.1}$$

and

$$P_e = \frac{1}{2}\left(1 - \sqrt{\frac{\overline{\gamma_b}}{1+\overline{\gamma_b}}}\right) \tag{6.2}$$

respectively. The erfc $(\cdot)$ is the complementary error function [**70**] and E_b and N_o is the energy of bit and the spectral density of noise, respectively, and

$$\overline{\gamma_b} = \frac{E_b}{N_o} E\left[\alpha^2\right] \tag{6.3}$$

The random variable α^2 has the chi-squared PDF and its average is given by $E\left[\alpha^2\right]$.

The soft decision (SD) decoding was used to obtain the simulation results of bit error ratio (*BER*). Concerning to hard decision (HD) we only present some *BER* upper bound results obtained by [**63**]:

$$BER \leq \sum_{m=2}^{M} \left(4P_e \cdot (1-P_e)\right)^{w_m/2} \tag{6.4}$$

where $M = 2^k$ is the total number of code words with weigh w_m and k is the number of information bits in each information word.

Concerning the precision of results, in all simulations the number of information bits, *NIB* , was simulated considering a given confidence level, C_l [**71**]:

$$NIB \geq 2\left(\frac{1-BER}{BER}\right)\left[\text{erf}^{-1}(C_l)\right]^2 \tag{6.5}$$

where $\text{erf}^{-1}(\cdot)$ is the inverse error function. As an example we considered $C_l = 99.9\%$. The performance results are depicted in figures 6.1, 6.2 and 6.3 for AWGN, and for fading with an average depth fading of 0dB and -20dB, respectively.

The Figure 6.4 illustrates the performance of TCH codes (64,7,13) obtained by [**6**] and the method explained in chapter 5 (using the codeword CCE03C51DBED2A4C). As it can be observed, once Eb/N_0 approaches the average fading level, the results present a small improvement for codes generated by the method of chapter 5. This advantage is more evident for lower BER. Due to the greater minimum distance of the SN code (resulting in a number of information

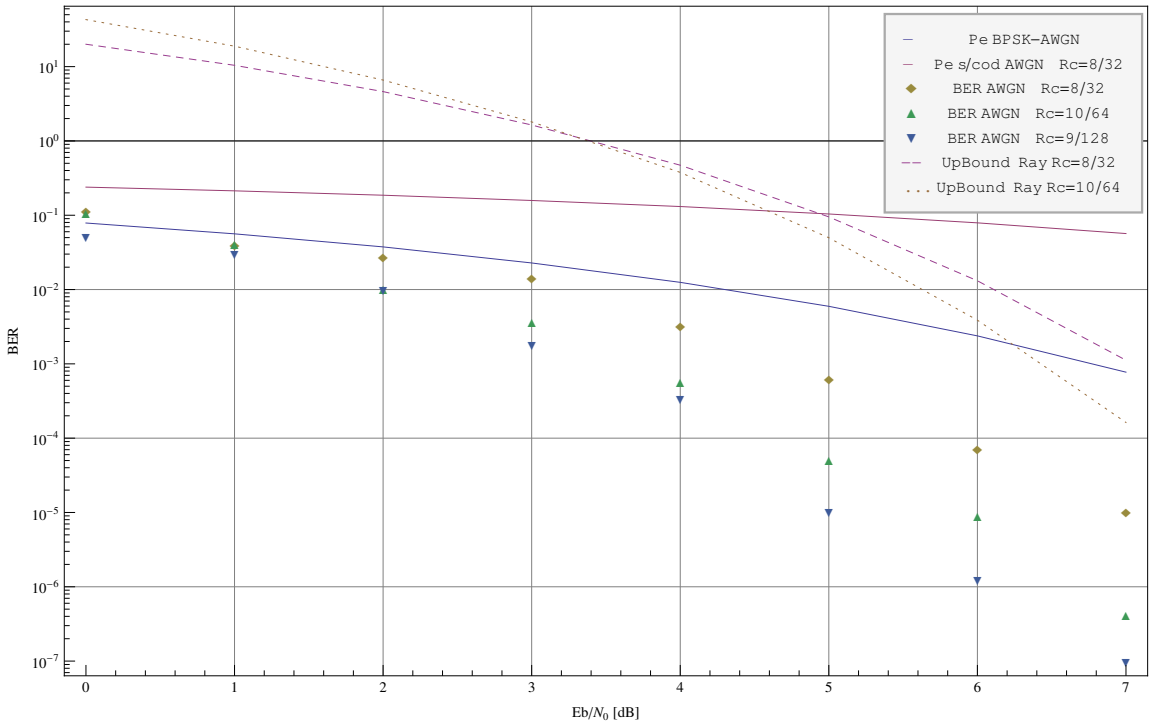


Figure 6.1. BER performance of three different codes for an AWGN channel

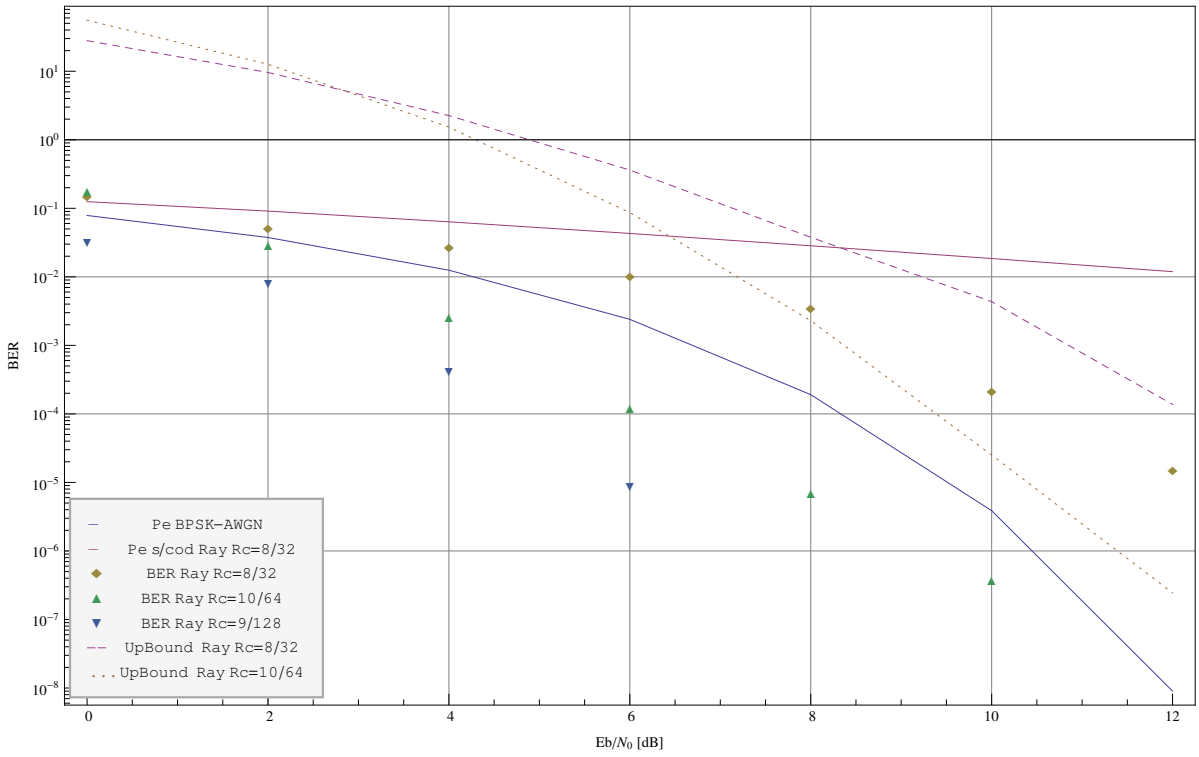


Figure 6.2. BER performance of three different codes for a Rayleigh channel with 0 dB fading

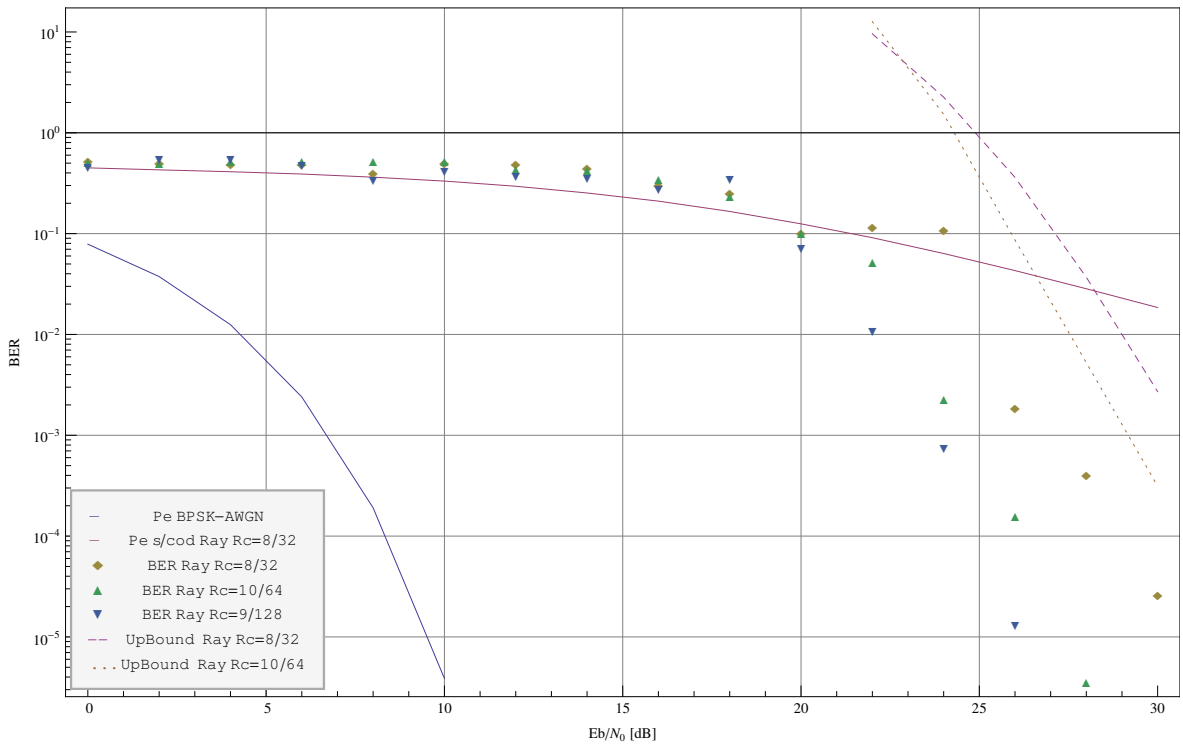


Figure 6.3. BER performance of three different codes for a Rayleigh channel with -20 dB fading

bits of 7.5) there is a better error correction ability. In addition, as observed from the graph at BER $= 1\text{E}^{-6}$ the advantage of the SN code vs the TCH code is approximately $\frac{3}{10}$ dB.

6.2. Synchronization

6.2.1. Spread spectrum systems and code acquisition. Synchronization involves the generation of a concurrent system of reference such that signal alignment in some particular domain is attained. Typically but not exclusively, synchronization takes place in the temporal and/or frequency domains. Synchronization can also be seen as an estimation problem where one or more parameters have to be determined from a given signal. Different levels of synchronization can be defined, like carrier, code, bit, symbol, frame and network synchronization.

Coherent reception assumes that the receiver knows (or by some means estimates) the carrier phase and frequency. It is important to understand that a receiver must also detect the incoming carrier signal by replicating the carrier frequency plus Doppler. Thus it can be stated that signal acquisition and tracking process is a two dimensional (*pseudo-noise* (PN) sequence phase and carrier frequency) signal replication process. The time/frequency uncertainty region composed by unitary search cells (Fig. 6.5) is defined by system and receiver characteristics. To represent the time-frequency uncertainty range a two-dimensional state matrix is used. The

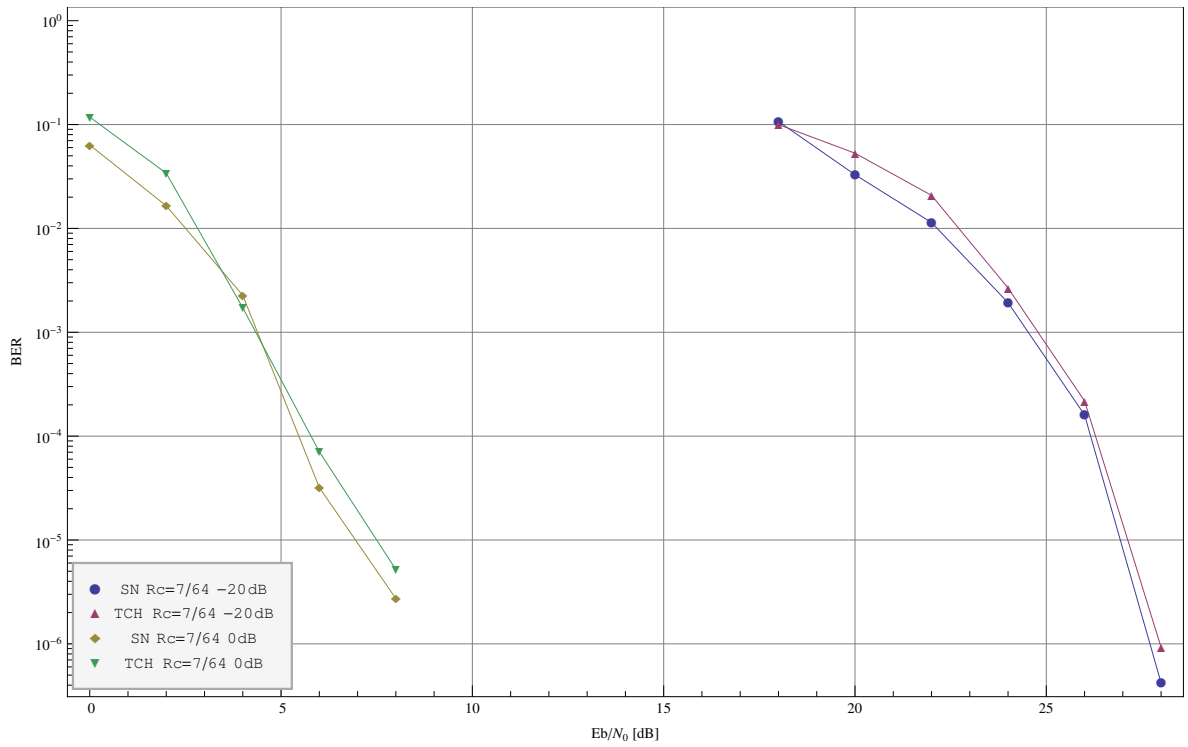


Figure 6.4. BER comparison of two 64-bit codes for a Rayleigh channel with 0 dB and -20 dB fading

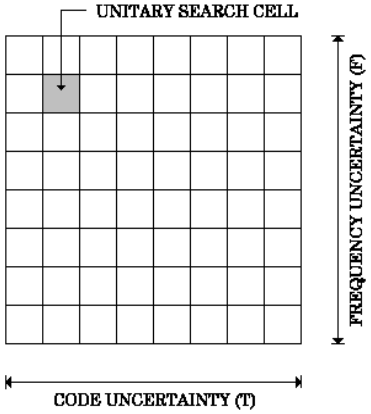


Figure 6.5. Time/frequency uncertainty region

matrix represents the quantization of the uncertainty range, in the PN code-phase axis and in the PN frequency-offset axis (due to oscillator drifts and Doppler effects). The region to search is given by *m* code phase hypotheses and *n* carrier frequency offset hypotheses. Therefore there are *m* by *n* cells to be tested by the acquisition part of the receiver.

Carrier synchronization is the procedure by which the receiver replicates the local carrier with the same frequency and phase than that found in the received carrier. Only after that, is coherent operation (i.e., detection) possible. In *Spread-Spectrum* (SS) systems the transmitted signal is spread by a spreading code according to *Direct-Sequence* (DS) or *Frequency Hopping* (FH) modulation schemes. In both cases the receiver has to align its locally generated spreading sequence to the corresponding received one to allow signal de-spreading and further detection. This operation is known as *code synchronization*. In order to make a decision about a received symbol the epochs in which the symbol starts and ends has to be available to the receiver. The estimation of these time instants is referred to as *symbol synchronization*. At a higher level, when signaling is highly structured, periodical timing is required to indicate the starting of a certain frame containing a number of separable signals. *Frame synchronization* procedures are typical in multiple-access systems based on *Time Division Multiple Access* (TDMA), where the periodical signaling scheme is repeated after an initial reference frame. At the top of the synchronization hierarchy is *network synchronization*, which encompasses methods and techniques for creating and distributing a common timing reference to a number of nodes defining a network.

An initial coarse synchronization process known as a *code acquisition* is followed by a fine synchronization process known as *code tracking* [**72**].

The most commonly employed modulation techniques in spread spectrum communications are direct-sequence and frequency-hopping spread spectrum. *Time Hopping* (TH) systems and hybrid combinations of the mentioned approaches are also possible. In general the spectrum occupied by the transmitted signal largely exceeds the corresponding bandwidth of the original signal to be transmitted. In DS-SS the spectral expansion is achieved by modulating each unit of information to be transmitted onto a random-like code sequence of pulses (e.g., signature sequence), where each pulse is denominated as a chip. In FH-SS the carrier frequency of the transmitted signal is changed on a regular interval basis following a predetermined hopping pattern. In the receiver DS signals are demodulated by multiplying the received signal by an aligned replica of the signature used in the transmitter. The alignment condition is essential for a successful demodulation since only when the codes are perfectly synchronized the received signal is reverted to its original format. Equally, in FH systems the receiver has to use the same hopping pattern, properly aligned, in order to demodulate. It is obvious that a mechanism for establishing a temporal synchronization between receiver and transmitter is fundamental to exploit the FH principle. In the synchronization process the receiver accommodates its own timing according to the timing of the received signal to attain a common temporal reference.

In DS-SS systems the signature sequence used for spreading and de-spreading the signals is commonly referred to as the spreading code and the process of sequence alignment is known as

code synchronization, as was stated previously. Typically, code synchronization in DS systems is carried out in two phases, an initial code acquisition followed by code tracking. Code acquisition is a coarse synchronization process by which the received and locally generated codes are brought into phase with a residual error of a fractional part of a chip. Once the codes are roughly aligned the remaining phase difference is reduced and kept to zero by a code tracking procedure. Spread-spectrum systems are characterized by a *processing gain* (PG), a parameter measuring the bandwidth expansion achieved by the modulation. It is easy to show that the effect of the interference on the desired signal is attenuated also by a factor equal to PG. In general the performance of an SS system can be related to the PG of the system. Thus, noise plus interference rejection, low probability of detection capabilities, link performance and system capacity in CDMA networks, among others performance figures of merit, are very dependent on the PG. However, during the code acquisition process of any SS system, the received signal remains basically as a wideband signal detrimentally combined with interference and noise. In particular, since PG cannot be exploited at this initial stage the effects of interference and noise cannot be precluded. Signal (or user) separation obtained by mutually orthogonal spreading sequences in CDMA networks is not available during code acquisition, a target signal can only be separated from the composite CDMA sum signal after the synchronization and de-spreading process takes place. Here lies a major difficulty during the initial synchronization phase.

Model for code acquisition. We consider a general direct sequence spread spectrum (DS-SS) modulation system described as follows: The data waveform is given by

$$d(t) = d_n,\ nT_S \leq t \leq (n+1)\,T_S \tag{6.6}$$

where $\{d_n\}$ is the binary data symbol with values in $\{-1,1\}$ and n an integer.

The spreading sequence is

$$c(t) = c_k,\ kT_C \leq t \leq (k+1)\,T_C \tag{6.7}$$

where$\{c_k\}$ is the code chip with values in $\{-1,1\}$ and k an integer. The spreading factor L is given by $L = T_S/T_C$ with T_S the symbol time and T_C the chip time duration. The chip function shaping is $p(t)$ and the spread signal (for a single user) is

$$s(t) = \sum_k d_n c_k p(t - kT_C), \quad n = \lfloor k/L \rfloor \tag{6.8}$$

The formulation of the code acquisition problem can be stated by first defining the DS received signal of the form

$$r(t) = s(t-\tau) + n(t) \tag{6.9}$$

where the transmitted signal $s(t)$ is spread by a code sequence $c(t)$, τ is the delay associated with the transmission and $n(t)$ represents additive noise plus interference. The code sequence

(or spreading code) consists of a pseudorandom succession of m pulses. Each pulse is denominated by a chip with length T_C. Due to the unknown delay imposed by the radio channel the correct temporal position of the code sequence is not known to the receiver, resulting in the received code sequence $c(t-\tau)$. The receiver generates a replica of the spreading code sequence with a controllable delay τ_{local}, such that this local sequence results in $c(t-\tau_{\text{local}})$, $0 \leq \tau_{\text{local}} \leq B \cdot T_C$ with $B \leq m$. The task of code synchronization is to align these two sequences in the time domain, or in other words to provide an estimate $\hat{\tau}$ of the delay τ such that the local delay is set according to $\tau_{\text{local}} = \hat{\tau}$. The synchronization evaluation can then be stated as: Let τ be the time delay in the received sequence and the locally generated reference sequence. If T_C is the chip duration and L the sequence length, then $\tau \in [0, L \cdot T_C]$ and $\hat{\tau}$ is its estimate. Synchronism is acquired if the following condition is satisfied:

$$0 \leq |\tau - \hat{\tau}| \leq \delta \cdot T_C \vee (L-\delta) \cdot T_C \leq |\tau - \hat{\tau}| \leq L \cdot T_C \tag{6.10}$$

Usually $\delta = \frac{1}{2}$, i.e. the acceptable timing error is half a chip. Though τ is a continuous variable, practical implementation requires the discretization of estimate values range. In this work the evaluation of the receiver architecture is based on the first attempt acquisition success probability, p_D (or alternatively on the acquisition error probability, $1-p_D$) in a finite time instead of the traditional mean acquisition time. Here p_D is the probability of estimating correctly the code phase of the incoming sequence (the first time it is tested).

Approach. The essential operative constituents of code acquisition are a plan of action to achieve the acquisition state and a function to identify the presence or not of alignment. The former is known as the *search strategy* while the latter corresponds to the *detector structure* employed by the receiver. These are the most important functions of the acquisition process.

6.2.2. Search strategies. Given the received signal $r(t)$ and the locally generated code replica $c(t)$ the receiver will apply a given procedure to determine the position in which code alignment occurs. Each relative position between the codes is called a *cell*, or strictly speaking a delay cell, to differentiate it from frequency and angular cells. The *uncertainty region* is defined as the total number of cells to be searched. In practice the length of the uncertainty region is kept to a manageable (low) number by dividing (i.e., quantizing) the uncertainty region into a finite number of cells. The procedure followed to explore the uncertainty region is referred to as a search strategy. Cells are tested by correlating the received and locally generated codes over a dwell time τ_d. A detector is employed to carry out the testing operation. The position in which code sequences are in-phase, henceforth leading then to the acquisition state (ACQ), is referred to as a *synch cell*. Note that there could be as many synch cells as the number of signal replicas resolved by the receiver. The remaining out-of-phase positions between codes correspond to *nonsynch cells*. The evaluation of each cell is modeled by the conventional hypothesis testing

tool H_i, with $i = 1$ for synch cells (sync hypothesis H_1) and $i = 0$ for nonsynch cells (out-of-sync hypothesis H_0). Typically, it is accepted that the acquisition process comes to its end when one synch cell has been detected. However, the definition can be extended as to consider that the acquisition process is finished upon detection of $L_{\text{acq}} = L$ paths.

Maximum likelihood. The *maximum likelihood* (ML) approach to code acquisition can be seen as a method in which the timing information is obtained from the received signal by a concurrent testing of all possible cell positions. This requires a massive use of parallel detectors to simultaneously examine all the cells defining the uncertainty region. A detector performs simultaneous correlation between the received signal and each of the locally generated realizations of the code sequence. Upon a single observation of the received signal $r(t)$ the ML estimate of the delay associated with the i$^{\text{th}}$ component (e.g., the i$^{\text{th}}$ synch cell) is given by $\hat{\tau}_i = \arg\max(r|\tau_i)$, or, in other words, the ML estimates $\hat{\tau}_i$ corresponds to the detector yielding the maximum output τ_i.

Serial search. The most common approach to code acquisition is to progressively shift the phase (delay) of the local code sequence in a serial fashion by steps, starting from an arbitrary initial cell. At each shift position the relative phases of the codes are compared and the process is serially repeated until a correct phase alignment is detected. This simple procedure, known as straight-line *serial-search* code acquisition, is used when no *a priori* information about the most likely alignment positions is available. In that case the probability density function (pdf) of the synch cell is assumed to be uniformly distributed within the uncertainty region. In order to align the codes the local sequence is shifted in fixed steps of length δT_C where typically $\delta^{-1} = 1, 2, 4$. The uncertainty region can be seen as the combined result of range ambiguity and relative movement between the k$^{\text{th}}$ user and receiver, as well as due to clock instabilities, lack of synchronization between transmitting and receiving clock frequencies, clock drifts, etc. In practice the length of the uncertainty region ranges from a few cells to the total number of available cells m/δ. In many applications range ambiguity can be considered as the principal contributor to delay domain uncertainty. The serial-search approach assumes that the channel remains unchanged during the whole examination period. A structured classification of serial-search strategies and their analysis, including z-search and expanding window approaches, were studied by [**73, 74, 75**].

Parallel search. In the search techniques described in the previous section only one correlating element or detector was used, hence the serial nature of the search. Parallel search makes use of a larger number of correlating elements. In one extreme the receiver could use m correlating elements to simultaneously search the m cells composing the uncertainty region. This will largely reduce the acquisition time, but on the other hand, implementation complexity of such a receiver will increase with m, being unpractical for long spreading codes. If $n < m$

detectors are available, each detector could search in an uncertainty region of reduced length, that is m/n cells.

6.2.3. Detector structures. In order to determine whether a cell corresponds to the synchronized position or not, the received signal $r(t)$ containing the spreading code is correlated with the locally generated delay-controllable version of the same code. The operation is denoted as

$$\int_0^{\tau_d} r(t)\, c(t-\tau)\, dt \tag{6.11}$$

An energy detector is the structure used to carry out the correlating operation defined in Eq.6.11. The detector plays a fundamental role in the performance of the acquisition process and its task is to detect with a high degree of reliability the presence of synchronized or non-synchronized cells. The signal correlation is computed over a finite period of time td, known as the *integration time*, *dwell time* or *observation time*. In principle, two basic approaches are possible here, namely *coherent* or *non-coherent detection*. In general coherent detection is not used in the context of code acquisition due to the requirement of carrier phase information for the operation of coherent correlation. Indeed, code acquisition takes place before the carrier phase tracking loop is activated due to the fact that estimation of carrier phase from a wide-band, low-spectral-density signal (e.g., previous to de-spreading) is difficult, if not impossible, particularly in scenarios with low SNR.

At a given cell the detector output (or decision variable) y is compared to a threshold T_h to make a decision about that cell. When the codes are actually in phase (hypothesis H_1) the synchronized cell position will be detected with a probability of detection p_D and missed with $p_M = 1 - p_D$. When the local code sequence is shifted in steps smaller that the chip duration as well as in cases of resolvable multipath propagation, more than one synchronized position can be found in the uncertainty region. Thus, at the synchronized cell position the detector will declare that the code sequences are in phase whenever the detector output exceeds the threshold reference value.

In any of the out-of-phase positions (hypothesis H_0) a synchronized cell could be wrongly declared, with a probability of false alarm p_{FA} or the non-synchronized cell could be correctly detected with a probability $1 - p_{FA}$. The probabilities p_D and p_{FA} are very much dependent on the SNR prevailing during the code correlating operation as well as on some detector parameters, and thus, they have a major impact on acquisition performance. In general a false alarm leads the acquisition process to a time-consuming state. In fact, the tracking operation will be activated but, since the signal is out of the pull-in range of the tracking loop, the system will return to the acquisition process to resume the search. The time required by the system to detect the false alarm state and return to the acquisition task (known also as the penalty time) is

a random variable but in most theoretical approaches it is modeled as fixed (e.g., a multiple k of the dwell time τ_d).

The correlation between received and local codes can be performed sequentially or concurrently. In the first case an *active correlator* computes the correlation on a chip-by-chip basis (i.e., serially) while in the second case a *passive code-matched filter* correlates a number of chips in parallel. Though both active and passive correlating units materialize the correlation operation of Eq.6.11, some practical differences between the two methods can be pointed out. These approaches can be classified according to the speed required to form the decision variable y. Active correlation of N spreading code chips requires $N \cdot T_C$ seconds while the same operation with a MF based passive correlator of length N (i.e., acquisition window length N) is carried out in T_C seconds. In terms of speed, the superiority of the MF scheme is clear for large values of N, but on the other hand, its inherent complexity makes its implementation feasible only for low to moderate values of N. The active correlator can be seen as a minimum-complexity approach where only a single and simple correlating unit is employed. The basic active correlator and passive MF approaches are studied in detail by [**74**].

As an example application of an MF in practical synchronization systems, the problem of initial synchronization in *Wideband CDMA* (WCDMA) systems is considered. In this system the acquisition problem consists in determining the timing and identity of the received pseudo-noise (PN) sequence. A multistage procedure is applied, where the initial timing is obtained from the *Primary Synchronization Code* (PSC) contained in a *Synchronization Channel* (SCH), available during one tenth of the time slot. An MF matched to the PSC is employed to acquire the slot timing of the strongest cell. A *Secondary Synchronization Code* (SSC), orthogonal to the PSC, conveys 16 short codes used for frame synchronization. An equal number of MFs, each matched to a particular short code, can be used for that purpose in a maximum-likelihood approach. Initial synchronization for WCDMA has been studied by [**76, 77, 78**].

Single and multiple-dwell detectors. The above-described simple correlating approach is not effective in terms of the time required to reach a synch cell. In a typical system there are far more nonsynch cells than synch cells. Thus, most of the time is spent in testing cells corresponding to nonsynch positions. In addition, since a false alarm state is associated with every nonsynch cell, the time to acquire could be excessively long. Different approaches based on repeated observation of the cells have been developed to reduce the acquisition time. The methods considered so far make a cell decision based on a single-dwell or integration. A second dwell, usually characterized by longer integration time, could be used upon synch cell detection by the first dwell, to verify the correctness of the first (or tentative) decision, thus avoiding occurrences of false alarms. Generalization to multiple-dwell detectors, that is, consecutive

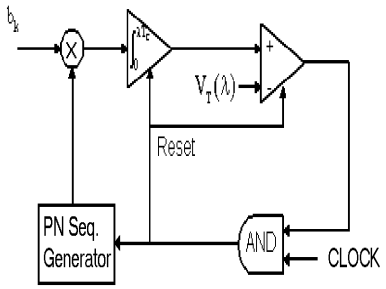


Figure 6.6. Classical correlation technique

tests to the same cell with successively increased dwell times, is straightforward. The synch cell is declared only after all the stages result in synch cell detection.

Another characterization of a detector can be done according to the nature of the dwell time. In *fixed-dwell time* detectors the integration time remains constant regardless of the cell being visited. The counterparts are detectors based on *variable-dwell time* or also known as *sequential detectors*. The argument favoring the latter approach is the fact that with fixed dwell-time schemes the detector spends the same time rejecting nonsynch cells than accepting synch cells.

Another classification of detectors accounts for the span of the correlation period considered, leading to *full-period correlation* when the spreading codes are correlated over the complete extension of the sequence, or *partial-period correlation* otherwise. When considerably long codes are used the detector makes a decision out of a partial correlation outcome, computing full-period correlations is practical in cases of short codes.

6.2.4. Other approaches to code acquisition. As referred previously code acquisition can be addressed in the frequency domain as opposed to time-domain. The conventional serial search scheme is normally simple in hardware, but the acquisition time is very long for long-duration PN sequences because its mean acquisition time is directly proportional to the period of the PN sequence employed, since several correlations must be performed [**6**] (one for each code phase tested). An alternative is to trade the time needed for performing correlations in the time domain with the slighter more complex algorithm (but taking a lesser processing time) of doing the same operation in the frequency domain.

Classical vs proposed synchronization. There are several well documented acquisition techniques in literature [**77, 79**]. The classical sliding window correlation method (Figure 6.6) is based on the comparison between the received PN sequence and a locally generated replica.

If the correlation level doesn't exceed a certain threshold, the receiver will increment the offset between the two sequences by half a chip and repeats the correlation procedure. When

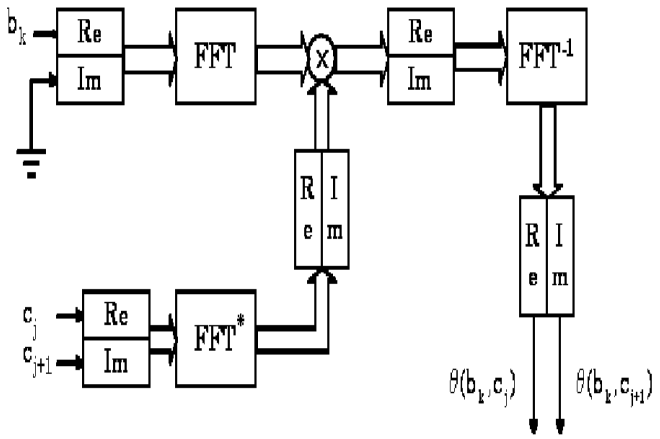


Figure 6.7. Frequency domain correlation technique

the given threshold is exceeded the signal acquisition is completed. Complexity reduction accomplished by using this method is set back by a considerable increase of the acquisition period. It can be easily seen that in a worse case scenario, the receiver will need to perform the correlation procedure $2L-1$ times until it reaches signal acquisition (L being the sequence period). Ideally, the minimum acquisition time of any sequence is proportional to the sequence period. One possible implementation strategy is to use $2L$ parallel correlation circuits (considering half chip offsets). Although with optimal performance, it is not feasible or practical when dealing with long period sequences.

Recent technological breakthrough has made possible the development of digital signal processing (DSP) with high performance and low costs. The use of DSP technology offers the possibility of producing optimal acquisition circuit as an alternative to classical techniques.

One example is illustrated in Figure 6.7 where data processing is done in the frequency domain rather than in the time domain where $\theta\left(b_k, c_j\right)$ represents the resulting correlation values between input sequence b_k and the local stored sequence c_j.

The frequency domain technique is described in [**8**], where the equivalence between time domain techniques and the frequency domain technique is exploited [**80**].

For two time domain sequences r and s the correlation is given by

$$\sum_{i=0}^{L-1} r_{(i+l)} s_i^* \longleftrightarrow R[m] S^*[m] \tag{6.12}$$

where R and S are the discrete Fourier transforms of r and s given respectively by

$$R[m] = \sum_{w=0}^{L-1} r_w e^{-j\frac{2\pi m}{L} w} \tag{6.13}$$

$$S^*[m] = \sum_{w=0}^{L-1} s_w^* e^{-j\frac{2\pi m}{L}w} \tag{6.14}$$

This is equivalent to say that the circular convolution of L-length sequences r_n and s_n can be computed through the multiplication of their corresponding DFT's, $R[m]$ and $S[m]$. The cross-correlation between y_n and x_n can be calculated by the convolution of y_n with x_{-n}. The correlation coefficients can be obtained by applying inverse discrete Fourier transform:

$$\theta_{r,s}(l) = \sum_{i=0}^{L-1} r_{(i+l)} s_i^* = \frac{1}{L} \sum_{m=0}^{L-1} R[m] S^*[m] e^{j\frac{2\pi l}{L}m} \tag{6.15}$$

This method has also the advantage of being able to test simultaneously two independent local sequences (c_j and c_{j+1} in Figure 6.7).

The calculus of a FFT or IFFT with L samples requires $L \cdot \mathrm{Log}_2 L$ complex multiplications and $L \cdot \mathrm{Log}_2 L$ complex additions.

Considering that the resulting FFTs of the local replica signals can be held in memory blocks, the number of basic operations is:

$$\begin{cases} L(2\log_2 L + 1) & \text{Complex multiplications} \\ 2L\log_2 L & \text{Complex additions} \end{cases} \tag{6.16}$$

The number of operations needed to calculate a single correlation point using the classical technique (with an offset of half a chip) is $2L$ algebraic multiplications and $2L-1$ algebraic additions. Knowing that the mean offset value between the incoming signal and the local replica is $L/2$, it can be concluded that the average equivalent number of basic operations regarding the classical method is given by:

$$\begin{cases} 2L \cdot L & \text{Complex multiplications} \\ (2L-1)L & \text{Complex additions} \end{cases} \tag{6.17}$$

Based on the evidence that a complex multiplication equals four algebraic multiplications and two algebraic additions and that the additions can be ignored in terms of calculus requirements, it can be defined a calculus reduction ratio (CRR) as the quotient of the basic operations between the two considered techniques:

$$\mathrm{CRR} = \frac{L}{2(2\log_2 L + 1)} \tag{6.18}$$

This gain factor is 4 for 16 chip sequences and over 20 for 256 chip sequences. The CRR should not be interpreted nor be a reference to infer on the relation between acquisition time which depends on the electronic implementation of both techniques. The undeniable advantage of the FFT based technique rest in the possibility of acquire a certain received SS signal with

only a single sequence period sample. This equals the improbable best case scenario with time domain techniques. The expected (mean) acquisition time value for time domain techniques equals L sequence periods (for half chip offset resolution).

6.2.5. Multi-rate receiver using cyclic sequences. We consider a general direct sequence spread spectrum (DS-SS) modulation system described as follows:

The data waveform is given by $d(t) = d_n$, $nT_s \leq t < (n+1)T_s$ where $\{d_n\}$ is the binary data symbol with values in {-1,1} and n an integer. The spreading sequence is $c(t) = c_k$, $kT_c \leq t < (k+1)T_c$ where $\{c_k\}$ is the code chip with values in {-1,1} and k an integer. The spreading factor S is given by $S = T_s/T_c$ with T_s the symbol time and T_c the chip time interval. The chip shaping is $p(t)$ and the spread signal (for a single user) is

$$s(t) = \sum_k d_n c_k p(t - kT_c),\ n = \lfloor k/S \rfloor \tag{6.19}$$

Root raised cosine filters split between the transmitting and receiving sections are used for $p(t)$. The multi-user case is implemented with code-division multiple access (CDMA) where each user is assigned a different spreading code.

In the frequency based decoder we use an N-point DFT for each N-chip length code. It turns out that an N-point DFT can be partitioned into M smaller L-point DFT's where $N = LM$.

Assume a divide and combine approach where for a sequence $x[n]$ of length N we divide the sequence into M smaller sub-sequences of length L. Using iterators l and m,

$$n = Ml + m, \quad 0 \leq l \leq L-1, \quad 0 \leq m \leq M-1 \tag{6.20}$$

Similarly,

$$k = p + Lq, \quad 0 \leq p \leq L-1, \quad 0 \leq q \leq M-1 \tag{6.21}$$

and therefore we can write sequences $x[n]$ and $X(k)$, the DFT (Discrete Fourier Transform) of $x[n]$, as arrays $x[l,m]$ and $X(p,q)$ each with L rows and M columns. Then, the DFT of $x[n]$,

$$X(k) = \sum_{n=0}^{N-1} x[n] \cdot W_N^{nk}, \quad 0 \leq k \leq N-1 \tag{6.22}$$

can be written as,

$$\begin{aligned} X(p,q) = & \sum_{m=0}^{M-1} \sum_{l=0}^{L-1} x[l,m] \cdot W_N^{(Ml+m)(p+Lq)} \\ = & \sum_{m=0}^{M-1} \left\{ W_N^{mp} \sum_{l=0}^{L-1} x[l,m] \cdot W_N^{Mlp} \right\} \cdot W_N^{Lmq} \\ = & \sum_{m=0}^{M-1} \left\{ W_N^{mp} \sum_{l=0}^{L-1} x[l,m] \cdot W_L^{lp} \right\} \cdot W_M^{mq} \end{aligned} \tag{6.23}$$

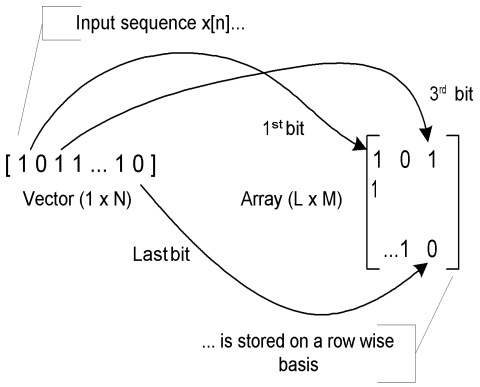


Figure 6.8. Sequence re-ordering when stored on array $x[l,m]$

where we used the periodicity property of the *twiddle* factor $W_N = e^{-j2\pi/N}$. From this result we can see that a N-point DFT can be computed by taking M smaller L-point DFT's and then combining these into a larger DFT using L smaller M-point DFT's. From the computation point of view this result can be implemented in a three-step procedure:

For each of the columns $m = 0,...,M-1$, compute an L-point DFT and store the result in array $F(p,m)$.

$$F(p,m) = \sum_{l=0}^{L-1} x[l,m] \cdot W_L^{lp}, \quad 0 \leq p \leq L-1 \tag{6.24}$$

Note that this corresponds to the inner sum of (6.23). Moreover, note that the DFT are computed for sequences stored in the columns of $x[l,m]$ but due to partition of (6.20) these sequences were stored in a special order, i.e., not on a column basis but on a row first basis (see Fig.). Note that this sequence re-ordering is in fact a matrix interleaver.

Modify $F(p,m)$ to obtain another array $G(p,m)$ using the *twiddle* factor W_N^{pm},

$$G(p,m) = W_N^{pm} \cdot F(p,m), \quad \begin{cases} 0 \leq p \leq L-1 \end{cases} \tag{6.25}$$

For each of the rows $p = 0,...,L-1$ of $G(p,m)$ compute the M-point DFT. The N-point DFT will be present in $X(p,q)$ reading on a row wise basis.

$$X(p,q) = \sum_{m=0}^{M-1} G(p,m) \cdot W_M^{mq}, \quad 0 \leq q \leq M-1 \tag{6.26}$$

Consider the case where in the process of calculating a 256-point DFT we also compute 16 DFT's of 16 points. This is an elegant result where the same algorithm is used to process a 256-chip code or 16 times 16-chip codes.

Matrix interleaver. A block interleaver formats the encoded data into a rectangular array of N_R rows and N_C columns, interleaving $N_C N_R$ bits (or chips) at a time as indicated in Table 6.1.

Table 6.1. Sample re-arrangement of a matrix interleaver (of depth N_R)

1	N_R+1		
2	N_R+2		
$\vdots$	$\vdots$		
N_R	$2N_R$	$\cdots$	N_CN_R

$\uparrow$ N_R Rows $\downarrow$

$\leftarrow N_C$ Cols $\rightarrow$

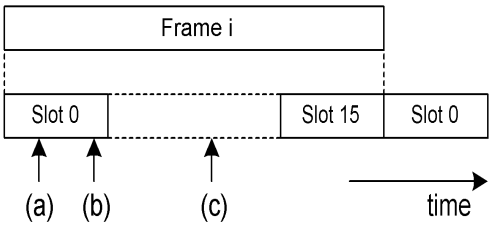


Figure 6.9. Initial time instant possibilities for synchronization search

Usually, each row contains a word of source data having N_C bits. An interleaver of degree N_R (or depth N_R) consists of N_R rows.

The structure of a block interleaver is such that source bits are placed into the interleaver by sequentially increasing the row number for each successive bit, and filling the columns. The interleaved source data is then read out row-wise and transmitted over the channel. This has the effect of separating the original source bits by N_R bit periods. At the receiver, the de-interleaver stores the received data by sequentially increasing the row number of each successive bit, and then clocks out the data row-wise, one word (row) at a time. There is an inherent delay associated with an interleaver since the received message block cannot be fully decoded until all of the N_RN_C bits arrive at the receiver and are de-interleaved.

6.2.6. Performance results. Let's assume that a receiver is turned on. Although there is a frame format predefined for the data communication the receiver doesn't know where one frame starts or ends. Therefore, one possible solution is to force a transmitter to periodically broadcast a certain fixed code that the receiver searches and try to lock on. If this code is identified, the rest of the data can be understood correctly. In Figure 6.9 we assume a radio frame of TF seconds with 16 slots in each frame. For a SS-CDMA system TF is usually around 10 ms. We also assume that each slot is used to transmit 1024 chips. The first slot in each frame is used to transmit a synchronizing code. This is different from the usual approach of repeatedly transmitting a synchronizing code with a period of one slot.

In the context of synchronization we can anticipate several possible time instants for the receiver initial search:

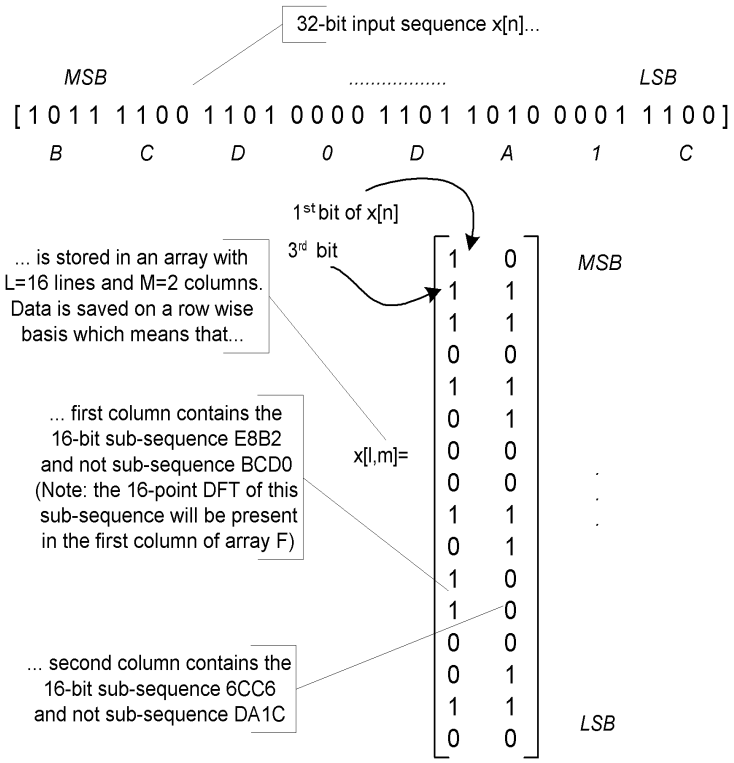


Figure 6.10. Example of a 32 bit sequence partitioned into two 16 bit sub-sequences

a) the receiver starts to load a sequence in the slot reserved for synchronization. At least one full sync code will be loaded and processed,

b) the receiver starts to load at the end of the sync slot such that it will not catch one full sync code but only a portion of it,

c) the receiver starts to load a sequence in the middle of the frame and therefore must wait until the next sync slot is transmitted.

The decoding/de-spreading operation which is also used for synchronization acquisition was simulated in Matlab. We start by defining an input sequence r which is then cut (stripping bits) at the head or tail to simulate the three possible initial time instants of Figure 6.9. Afterwards, sequence r is reordered due to the DFT partition. This is illustrated in Figure 6.10 for a smaller length sequence. Then the three step procedure for de-spreading is applied and then we analyze the correlation peaks to check for sync codes existence. To start the synchronization procedure the receiver loads a sequence of 512 bits into a FIFO memory (First-in first-out). This sequence is partitioned into 2 sub-sequences of 256 bits each (Figure 6.11). This example corresponds to case (a) of Figure 6.9. By changing *offset* other time instants can be simulated.

Each sub-sequence will have to be correlated with the known code words used for synchronization purposes. If a match is found the position of the correlation peak can report an offset

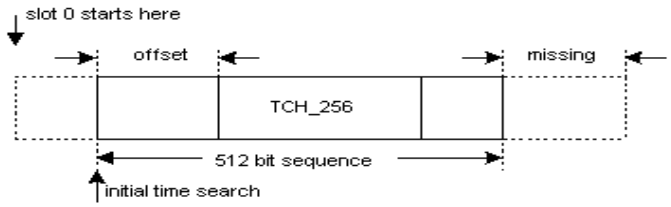


Figure 6.11. Example of a 512 bit data sequence including the sync codes

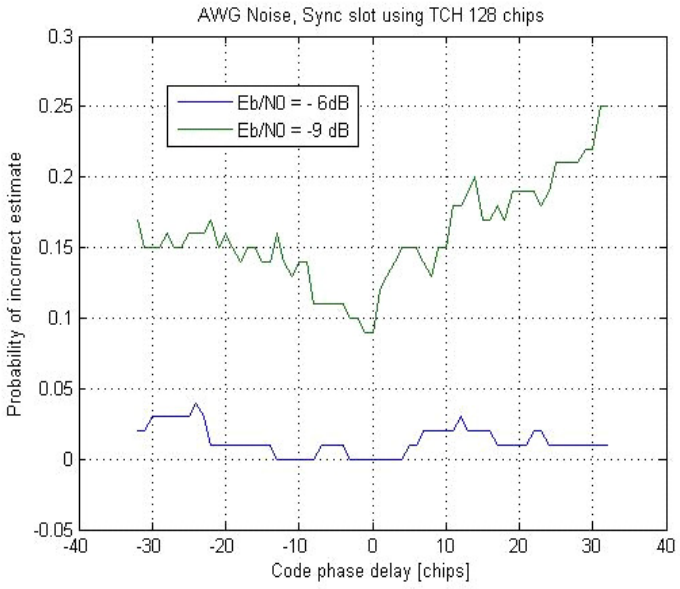


Figure 6.12. Probability of making an error in the estimation of the code delay for a misalignment between the local and received code sequence.

as to where the code starts. If a match is not found a new group of 512 bits is loaded and the correlation process starts over.

Frame synchronization. For a simple frame synchronization simulation we used a Monte Carlo analysis with better than 100 iterations depending on simulation time. The noise was specified by values of Eb/N0 ranging from -15 dB to 0 dB in steps of 3 dB. We used one sample per chip, and rectangular pulses and the probability of wrong code delay estimation is given in Figure 6.12. In this case a delay between -32 and 32 chips is depicted.

Multi-user case. For simulation of the multi-user case we used a Monte Carlo analysis with 1000 iterations. For each point we estimate the chip phase delay for misalignments between the received and local sequences of -8, 0 and 8 chips and average the probability of incorrect estimation over all iterations. The channel noise was specified by values of signal to noise ratio, SNR/chip and then converted to the metric Eb/No ranging from -12 dB to +12 dB as appropriate. Two cases were considered. The first case considers one single user. The probability

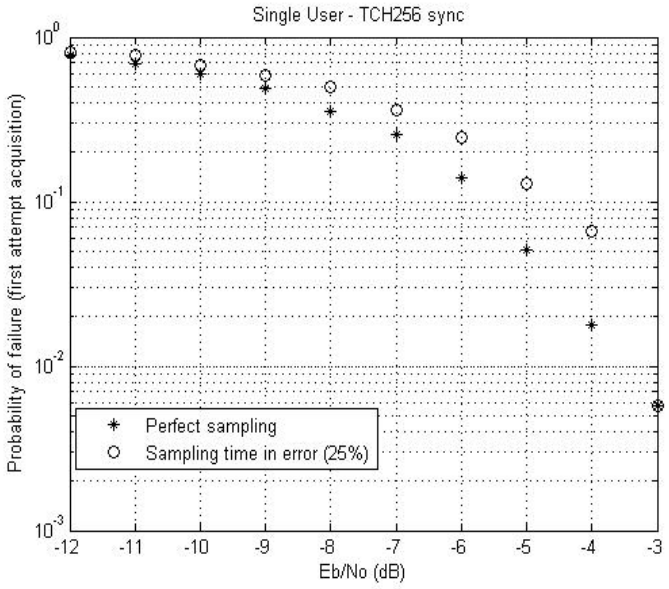


Figure 6.13. Probability of making an incorrect estimation of the code delay for a misalignment between the local and received code sequence for a single user.

(considering only the first attempt) of estimating correctly the code phase delay is illustrated in Figure 6.13. Both ideal and non-ideal sampling were simulated. The ideal instant corresponds to sampling the chip waveform when it reaches the peak. The non-ideal sampling corresponds to sampling the chip waveform one sample after the ideal instant case (four samples per chip were considered). In this case each point represents the average for delays between -8 and 8 chips. Degradation from non-ideal sampling as illustrated is only significant for low values of Eb/N_0.

The second case considers 2 and 4 simultaneous (asynchronous) users and results are presented in Figure 6.14.

For both cases we used Root Raised Cosine filters, half in the transmit section, half in the receiver section with a roll-off factor of 0.22. The TCH code used for synchronization has a period length of 256 chips.

Variable rate code synchronization. In order to validate the possibility of using synchronization codes of different lengths we simulated the synchronization process for codes of 16 and 256 chips. The conclusion is that although it is possible to achieve synchronization with 16-chip typical Eb/No requires codes of much larger cardinality.

The work presented in this section was published in [**12, 23, 24**].

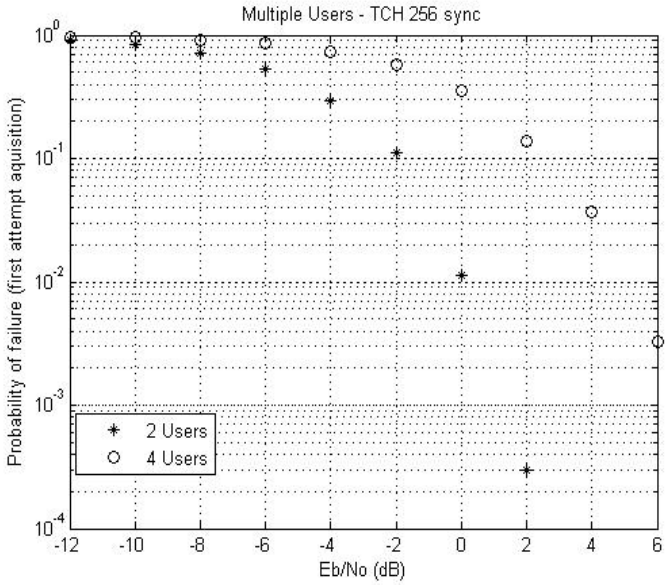


Figure 6.14. Probability of making an incorrect estimation of the code delay for a misalignment between the local and received code sequence considering 2 and 4 simultaneous (asynchronous) users.

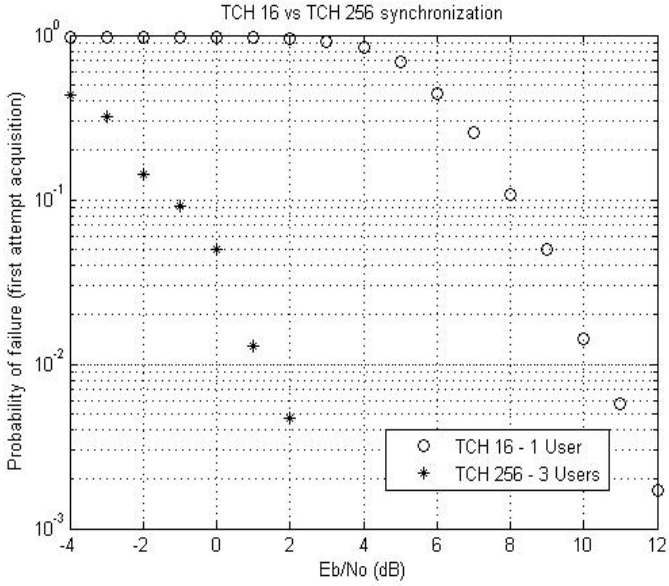


Figure 6.15. Probability of acquisition failure for 16 and 256 chip codes

6.3. Channel estimation

One of the most challenging problems in high data rate wireless transmission is to overcome the time dispersion caused by multipath propagation. High data rate mobile radio channels in

indoor or micro cellular environment can exhibit large relative time dispersions. The characteristics of this time-varying multipath propagation channel can be estimated using channel estimation methods with the help of pilot symbols and equalized using a suitable equalization scheme in the receiver.

6.3.1. Block Transmission Techniques. In block transmission techniques the data stream to be transmitted is split into several blocks. Each block has a preamble or cyclic extension (also called *cyclic prefix*) that makes the block appear periodic. The length of the cyclic extension is at least as long as the channel impulse response. This way the linear convolution becomes equivalent to the circular convolution and the latter can be computed in the frequency domain using the Discrete Fourier Transform. This type of signal processing is exploited in modulations like *Orthogonal Frequency Division Modulation* (OFDM) [**81**] and Single-Carrier (SC) combined with equalization in the frequency domain [**82**]. These two classes of modulation have been proposed as effective anti-multipath techniques since multipath is the dominant propagation impairment in broadband wireless channels. In these very dispersive channels *Inter-Symbol Interference* (ISI) is very significant and an equalizer is needed at the receiver for correct operation. To be as effective as possible the equalizer needs to know the channel, i.e. it needs to estimate the channel frequency response. Typically, these channel estimates are obtained with the help of pilots [**83**] or training sequences [**84**] that are multiplexed with data symbols. Typically, pilots and data symbols do not overlap so there is a rate loss. A promising technique to overcome this estimation overhead is to use implicit training or implicit pilots, where the training block overlap the data block instead of being multiplexed with it [**85**].

The training sequences used for channel estimation should, in principle, have ideal characteristics: they should be easy to generate, have a constant time envelope in order to allow efficient power amplification, have a signal spectrum with constant absolute value in order to prevent the noise enhancement effect. QPSK type sequences in the time domain are simple, have a constant envelope but generate deep fades in the frequency domain. On the other hand, QPSK type sequences in the frequency domain are also simple, have a signal in frequency with constant absolute value but the time envelope has many fluctuations (Gaussian-like as OFDM signals). An alternative is to use Chu sequences [**86**], which have constant envelope time samples and constant absolute value in the frequency domain. All of these sequences have an analog signal envelope with some fluctuations with zero crossings. A technique to reduce the fluctuations in both time and frequency was presented in [**87**] but the procedure is somewhat complex.

In this section we proposed to use modified TCH (Tomlinson, Cercas, Hughes) sequences to estimate the channel in block transmission schemes (either OFDM or SC-FDE). Recall that TCH codes are binary, nonlinear, non-systematic cyclic codes of length $N = 2^m$, m being any

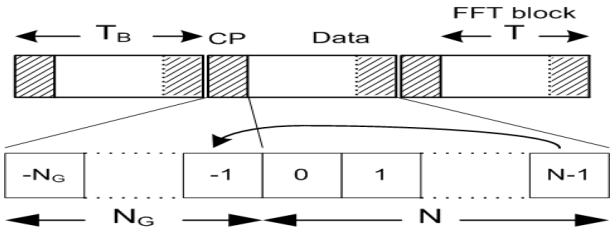


Figure 6.16. Block transmission frame format

positive integer. Since the length of TCH codes is a power of 2, correlations in the time domain can be done easily in the frequency domain with very fast FFT radix 2 operations. This is also useful since both OFDM and SC-FDE use FFT processing blocks thus sharing signal-processing resources. The best TCH codes known are derived from polynomials known as Basic TCH or B-TCH Polynomials. The first polynomial in a TCH code is generated by an analytical method and is then extended to increase the code set. A general and very important property of B-TCH polynomials is that their auto-correlation is always three-valued with the following non-normalized distribution: -4 for $N/4$ even shifts, 0 for $N/4-1$ even shifts and all odd shifts and finally N for no shift. As we discussed in the previous section TCH sequences have already been used in synchronization applications [**24**] due to their circular structure and correlation properties.

System Characterization. Block transmission systems typically employ the frame format depicted in Figure 6.16. The frame consists of N_T blocks each with duration T_B. In each block data is preceded with a cyclic prefix CP. The data payload part (also called useful block) has N samples. The last N_G samples of the block are copied to the beginning of the block. Since T is the duration of the FFT block, $T_S = T/N$ is the duration of a data symbol. The guard fraction G, is given by $G = T_G/T = N_G/N$ and therefore $T_B = T(N+N_G)/N$.

Transmitted signals for OFDM or SC-FDE. The transmitted signal associated with a frame is

$$x^{Tx}(t) = \sum_{m=1}^{N_T} x_m(t - mT_B) \tag{6.27}$$

where T_B is the duration of each block. The m$^{\text{th}}$ block has the form

$$x_m(t) = \sum_{n=-N_G}^{N-1} x_{n,m} h_T(t - nT_S) \tag{6.28}$$

with $x_{n,m}$ referring to the n$^{\text{th}}$ symbol of the m$^{\text{th}}$ block, N_G denoting the number of samples of the cyclic prefix, $h_T(t)$ the adopted pulse shaping filter and T_S the symbol duration.

In the following we will consider each block individually and as such we will drop the m subscript without loss of generality. Therefore x_n refers to the n$^{\text{th}}$ data symbol selected from

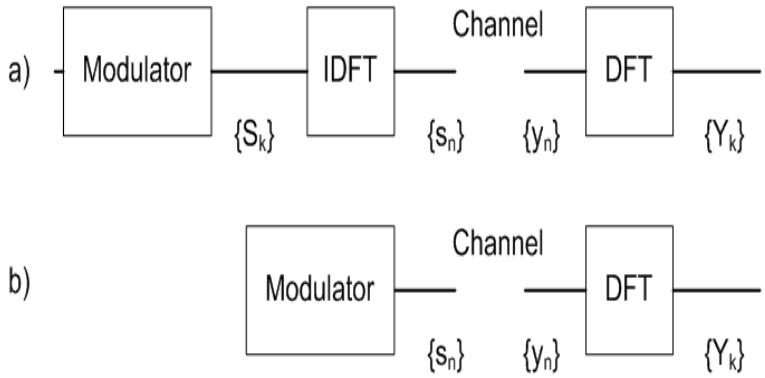


Figure 6.17. Signal notation for a) OFDM and b) SC-FDE

a given constellation (e.g. M-PSK) under an appropriate mapping rule (e.g. Gray coding). To refer a sequence, i.e. $x[n]$ we will use the more concise notation x_n, and for all data symbols forming a block we will use $\{x_n\}$, $n = 0,1,\ldots,N-1$. It is assumed that $x_{-n} = x_{N-n}$, $n = -N_G,-N_G+1,\ldots,-1$.

The signal x_n depends on the considered system, OFDM or SC-DFE, as illustrated in Figure 6.17.

Note that S_k is the frequency domain symbol to be transmitted on the k$^{\text{th}}$ carrier of a block and s_n is the n$^{\text{th}}$ time domain symbol in a transmitted block. The conversion between time and frequency domains is given by $S_k = \text{DFT}(s_n)$ and conversely, $s_n = \text{DFT}^{-1}(S_k)$.

The signal is transmitted over a time-dispersive channel and at the receiver side it is sampled and purged from the cyclic prefix leading to the signal y_n. The received signal is the result of the linear convolution between the channel response h_n and x_n. Due to the cyclic prefix, x_n appears periodic and therefore a circular convolution takes the place of the linear one. Circular convolutions in the time domain can be computed through DFT's in the frequency domain. Assuming that the cyclic prefix is longer than the channel impulse response the corresponding frequency domain block Y_k is obtained after a length-N DFT operation and given by

$$Y_k = X_k H_k + N_k \tag{6.29}$$

where H_k denotes the overall channel frequency response of the block in question and N_k denotes the corresponding channel noise. In general, the length of the channel impulse response h_n is less than N samples and therefore it is necessary to zero pad before applying the DFT in order to maintain the block length. We assume that the channel is almost invariant over the time frame, i.e. the product $T \cdot N_T$ must be less than the coherence time of the channel. Under this assumption it remains valid the drop of the subscript m referring to the m$^{\text{th}}$ block of a frame.

Channel Estimation. An equalizer can compensate the dispersive influence of the channel. This compensation requires an estimate of the channel. There are several alternatives for identification of the channel response in the form of training sequences or pilot symbols although it is also possible to use non-aided approaches called blind algorithms. In general, either the training sequences or pilots are isolated from the data but this reduces the throughput. Alternatively we can use implicit or superimposed pilots [**88, 89**]. Superimposed pilot sequences means that the known pilot sequence is overlaid on the data sequence and transmitted in the same frequency band and/or at the same time as the data sequence. In that case the transmitted signal contains both a known pilot part and an unknown data part.

In the conventional scheme, where there is no data overlapping the training or pilot block, an estimate of the channel can be obtained as follows,

$$\tilde{H}_k = \frac{Y_k}{X_k} = H_k + \frac{N_k}{X_k} = H_k + \varepsilon_k^H \tag{6.30}$$

where X_k is either S_k^P or S_k^{TS} depending on the system used, P for pilots and TS for training symbols.

The channel estimation ε_k^H error is Gaussian-distributed, with zero-mean and

$$\mathrm{E}\left[|\varepsilon_k^H|^2 \mid X_k\right] = \mathrm{E}\left[|N_k|^2\right] \cdot \mathrm{E}\left[\frac{1}{|X_k|^2}\right] \tag{6.31}$$

The power assigned to the known symbols is proportional to $\mathrm{E}\left[|X_k|^2\right]$ and $\mathrm{E}\left[\frac{1}{|X_k|^2}\right] \geq \frac{1}{\mathrm{E}\left[|X_k|^2\right]}$ with equality for $|X_k|$ constant.

To minimize the envelope fluctuations of the transmitted signal $|x_k|$ should also be constant. This can be achieved with Chu sequences [**86**], which have both $|s_n^{TS}|$ and $|S_k^{TS}|$ constant. In the next section we propose to use instead modified TCH sequences.

6.3.2. Channel Estimation using modified TCH sequences.

Modified TCH sequences. Let $c[n]$, $n = 0, 1, ..., N-1$, be a TCH codeword of length N. The DFT of $c[n]$ is $C[k]$ given by

$$C[k] = \sum_{n=0}^{N-1} c[n]\, e^{-j2\pi k \frac{n}{N}},\ k = 0, 1, \ldots, N-1 \tag{6.32}$$

It follows that $C[0] = \sum c[n]$ is the code weight and $C\left[\frac{N}{2}\right] = \sum_{n=0}^{N-1} c[n]\, e^{-j\pi n}$. Since $e^{-j\pi n}$ is $(-1)^n$ we can divide the range $[0, N-1]$ into even and odd n subranges and write

$$C\left[\frac{N}{2}\right] = \sum_{n=0}^{N/2-1} c[2n] - \sum_{n=0}^{N/2-1} c[2n+1] \tag{6.33}$$

the difference between the sum for even samples and the sum for odd samples. If we use sequences with polar values, i.e., a binary pair (0,1) is coded as (-1, 1), then $C[0] = C\left[\frac{N}{2}\right] = 0$. This is not attractive for channel estimation purposes because $1/C[0]$ and $1/C\left[\frac{N}{2}\right]$ would be infinity.

Let us define a new sequence $s[n]$ by copying first the spectrum of $c[n]$, i.e. $S[k] = C[k]$ and then modifying $S[k]$ such that $S[0] = \sqrt{N}$ and $S\left[\frac{N}{2}\right] = -\sqrt{N}$. Then, $s[n]$ is given by the inverse DFT of $S[k]$. This spectrum modification causes a magnitude deviation of the odd samples of $c[n]$ by

$$s[2n+1] = c[2n+1] + \Delta, \quad \Delta = \frac{2\sqrt{N}}{N} \tag{6.34}$$

If we had chosen to modify the spectrum by setting $S[0] = -\sqrt{N}$ and $S\left[\frac{N}{2}\right] = \sqrt{N}$ then the magnitude change would be in the opposite direction, i.e., $s[2n+1] = c[2n+1] - \Delta$. In the left column of Figure 6.18 we illustrate the first 32 samples of a TCH sequence length N=256 represented by $c[n]$, it's absolute value spectrum $|C[k]|$ with 2 zero points at $k = 0$ and $k = N/2 = 128$ and the IQ diagram for $C[k]$. In the right column of Figure 6.18 we illustrate the same results for the modified TCH sequence $s[n]$. Only the first 32 samples are shown in the time-domain sequence. The right column is for the modified TCH code. Note that $|s[n]|$ is still almost constant and that the absolute value of the spectrum $S[k]$ is also almost constant with an average value of $\sqrt{N}$. This modified sequence is thus appropriate for channel estimation purposes.

Assume that the first block of a frame consists of a spectra modified TCH code length 256. Due to the autocorrelation characteristics of this code it is also possible to use this block as a frame synchronization aid. In this section however we are concerned with its use in estimation. The procedure is as follows:

(1) Obtain a first estimate of the channel, $\widetilde{H}_k = Y_k/X_k$ where X_k is the frequency spectrum of the code.
(2) The estimate is then enhanced by truncating the channel impulse response $\tilde{h}_n$ in order to limit its duration to N_G samples, thus obtaining $\hat{h}_k = \tilde{h}_n w_n$ where w_n is a window defined as $w_n = 1$ if the n$^{\text{th}}$ sample is inside the cyclic prefix and $w_n = 0$ otherwise.

The sequence of operations is thus,

$$\widetilde{H}_k \rightarrow \tilde{h}_n \rightarrow \hat{h}_n \rightarrow \hat{H}_k$$

using the N-point DFT to go from the time domain to the frequency domain and the N-point inverse DFT vice-versa.

The performance of the estimation procedure can be computed using the mean square error (MSE), $|\widetilde{H}_k - H_k|^2$ and $|\hat{H}_k - H_k|^2$.

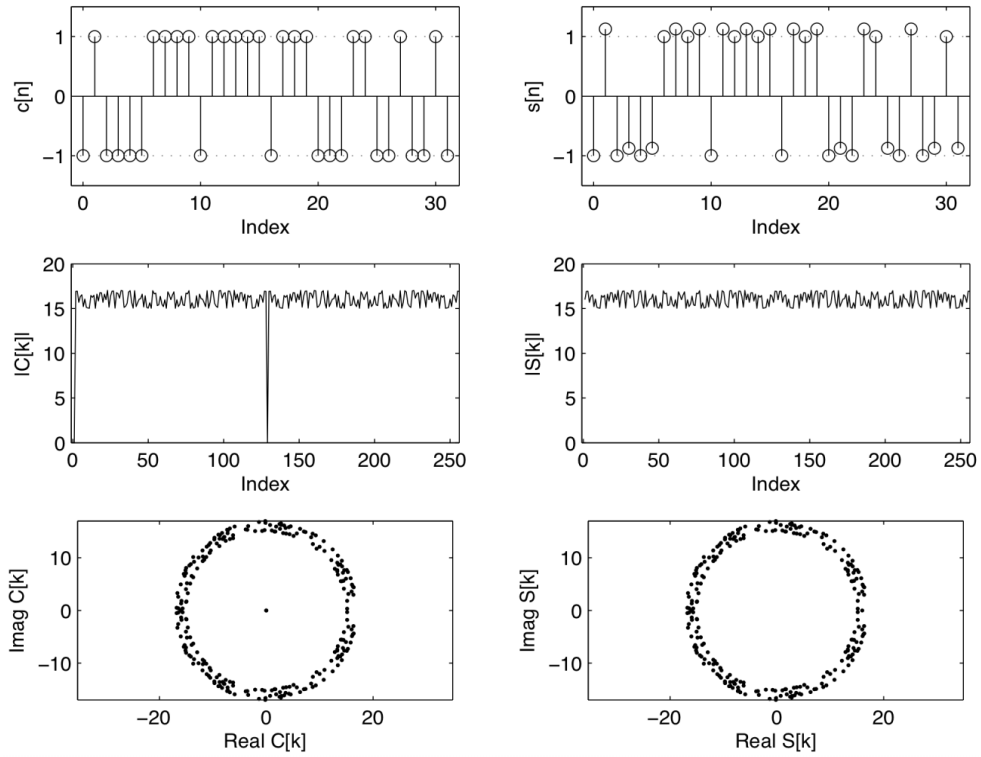


Figure 6.18. Time-domain sequence, absolute value spectrum and scatterplot of a length-256 TCH code (left column).

6.3.3. Simulation results. We start by evaluating the effect of applying the spectra modified TCH sequence in estimating a communication channel. In this case the standard parameters for a Hiperlan C channel were used. Figure 6.19 compares channel frequency responses. Using 256 samples of a modified TCH codeword we first compute its spectrum X_k and then obtain the channel estimate $\tilde{H}_k$. By round tripping to the time domain an enhanced estimate is obtained, $\hat{H}_k$, which follows closely the real channel response.

Then, in Figure 6.20 we compare the time domain channel impulse responses. After estimation the noisy $\tilde{h}$ needs to be resampled: the 32 samples of $\hat{h}_n$ correspond to the cyclic prefix length N_G.

The improvement on the mean square error on the estimates of H_k are presented in Figure 6.21. The dashed line corresponds to the error between the real channel and its estimate, $|\widetilde{H}_k - H_k|^2$. The solid line corresponds to the error between the real channel and its enhanced estimate, $|\hat{H}_k - H_k|^2$.

6.4. Joint coding and spreading in UWB

Impulse radio (IR) techniques [**90**][**91**] are the most popular in implementing Ultra-Wideband (UWB) transmission systems due to the reduced implementation complexity of impulse radio

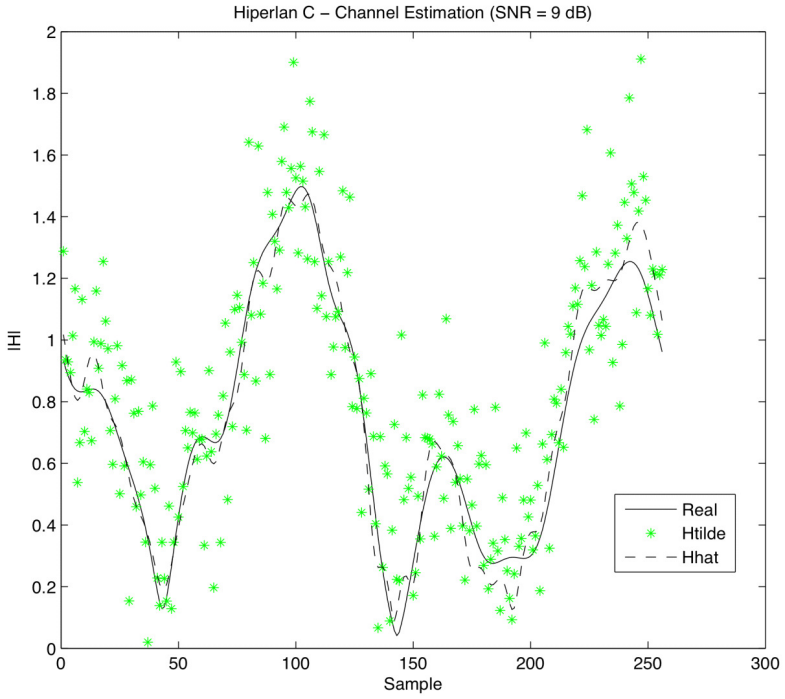


Figure 6.19. A snapshot of the channel frequency response H_k, its estimate $\widetilde{H}_k$ using a modified TCH length-256 code and the corresponding enhanced estimate $\hat{H}_k$.

when compared with continuous-wave UWB options. Impulse radio is a baseband signal approach and refers to the generation of very short duration (less than 1 ns) pulses. For UWB systems, in particular, each pulse has very low energy to adhere to spectral mask requirements. Typically, a series of pulses are combined to transmit a single bit. Continuous pulse transmission introduces strong spectral lines in the spectrum of the transmitted signal. Randomization of the pulse train is therefore necessary to minimize these spectral lines. Time-hopping (TH) combined with Pulse Position Modulation (PPM) [**92**] and Spread Spectrum (SS) are among the methods used for randomization of the pulse train.

Since the spectral efficiency achievable with IR techniques is generally low, there is an increased interest on UWB systems employing continuous-wave techniques such as OFDM (Orthogonal Frequency Division Multiplexing), DS-CDMA (Direct Sequence Code Division Multiple Access) and MC-CDMA (Multi-Carrier CDMA). Here we consider transmission techniques for UWB-based ad-hoc networks. In particular, we will address DS (Direct Sequence) spreading and focus on CS (Code Spread) schemes.

Two different coding stages exist in DS-CDMA. The first coding stage is typically an error-correcting turbo code with a relatively high coding rate (R=1/3). The second coding stage, known as spreading, is obtained by using a low rate repetition code followed by randomization.

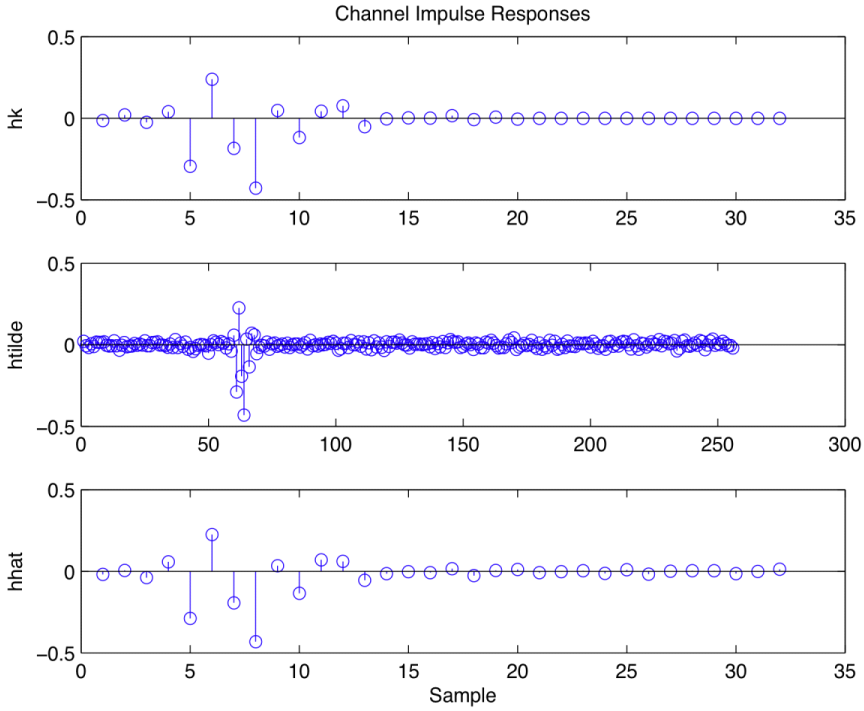


Figure 6.20. A snapshot of the channel impulse response h_n, its estimate $\tilde{h}_n$ using a spectra modified TCH length-256 code and the corresponding enhanced estimate $\hat{h}_n$.

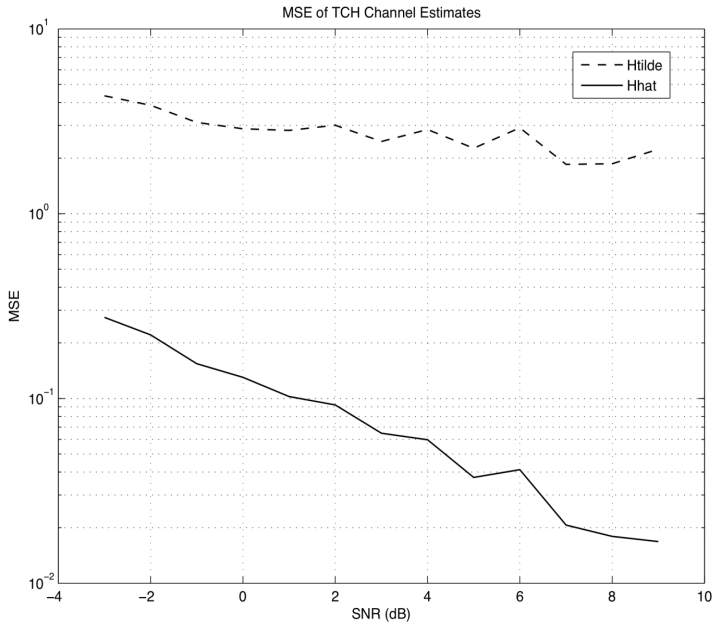


Figure 6.21. Mean square error of modified-TCH code estimates.

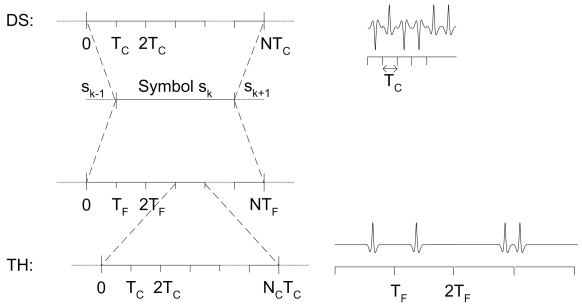


Figure 6.22. Timing definition for DS and TH transmission schemes.

By contrast, in CS-CDMA schemes only one stage is used and the bandwidth expansion is obtained through the use of a single low rate error detecting code. We propose to use TCH (Tomlinson, Cercas, Hughes) codes [**6**], in particular for CS-CDMA schemes.

6.4.1. System Characterization. Conceptually, data modulation occurs in three stages. First, a pulse train is generated. Second, a randomizing technique is applied to reduce the corresponding spectral spikes. Third, a data modulation is applied to carry the information. The two main approaches for randomizing the pulse train are time hopping (TH) and direct sequence (DS) techniques. Figure 6.22 illustrates the timing definition for both randomization schemes. In the DS scheme a symbol s_k is transmitted in a frame divided into N slots of duration T_C (*chip* time), therefore there are N pulses transmitted per symbol (the reason why this scheme is addressed as a high duty-cycle one). In the TH scheme, the same symbol s_k is used repeatedly in N sub-frames (of duration T_F each), i.e. the symbol rate is $1/NT_F$. Each sub-frame is further divided into N_C slots (usually the length of the TH code). According to the code sequence only one slot in each sub-frame is occupied by a pulse and since the average pulse repetition period T_F is much larger than the pulse width this is the reason why this scheme is also tagged as a low duty-cycle one.

TH-PPM Signal Model. In a typical TH-PPM system the transmitted signal from user k is modeled as

$$s_{TH}^{(k)}(t) = \sum_j \sqrt{\frac{E_S}{N_S}} p_{Tx}\left(t - jT_F - c_j^{(k)} T_C - \delta \alpha_{\lfloor j/N_S \rfloor}^{(k)}\right) \tag{6.35}$$

where E_S is the energy per symbol and N_S is the number of consecutive pulses for transmitting a single symbol, leading to a $(N_S, 1)$ repetition code. The transmitter signal pulse is normalized so that its energy is 1, i.e. $\int p_{Tx}^2(t)\,\mathrm{d}t = 1$ and its idealized form is that of a Gaussian monocycle (the second derivative of a Gaussian function) given by

$$p_{Tx}\left(\frac{t}{\tau_p}\right) = \left[1-4\pi\left(\frac{t}{\tau_p}\right)^2\right]\exp\left[-2\pi\left(\frac{t}{\tau_p}\right)^2\right] \tag{6.36}$$

with τ_p denoting a time normalization factor. The pulse width is $T_p \simeq 2\tau_p$. The duration of a frame is T_F and therefore the symbol duration is $N_S T_F$. The pseudo-random time-hopping code for user k is represented by $\left\{c_j^{(k)}\right\}$, $0 \le c_j^{(k)} \le N_h$, $\forall k$ and we assume $N_C = N_h$, i.e. the number of slots (or hops) is equal to the code length. The binary information stream from user k is represented by $\left\{\alpha^{(k)}_{\lfloor j/N_S \rfloor}\right\}$. The notation $\lfloor x \rfloor$ denotes the integer part of x, therefore the modulating data symbol changes only every N_S hops. The time shift δ is associated with the PPM modulation is such that no additional time shift is modulated on the monocycle when the data symbol is 0, but a time shift of δ is added to the monocycle time reference when the symbol is 1. The optimal value of δ is given by $\delta_{\text{opt}} = \arg\max\left[\mathrm{R}(0) - \mathrm{R}(\delta)\right]$ where $\mathrm{R}(.)$ is the cross-correlation between a received monocycle and the receiver assumed template. To avoid successive pulse overlap (inter-chip interference) it is necessary that $T_p + \delta \le T_C$. Commonly, for binary PPM, a value around $\frac{1}{4}T_p$ is used for the index modulation δ [**93**].

In a system with more than one active user different pseudo-random TH codes separate each user. In a frame there are N_C possible transmission intervals, so under ideal conditions a maximum of N_C users can be allocated into the system without creating interference.

Direct Sequence Signal Model. In this transmission scheme, DS-PAM or DS-BPSK, the signal from user k is modeled as

$$s_{DS}^{(k)}(t) = \sum_j \sqrt{\frac{E_S}{N_S}}\alpha_j^{(k)} \sum_{i=0}^{N_h-1} c_i^{(k)} p_{Tx}(t - jT_F - iT_C) \tag{6.37}$$

The pseudo-random spread-spectrum code assigned to user k is represented by $\left\{c_i^{(k)}\right\} \in \{-1, 1\}$ and the data information stream is represented by $\left\{\alpha_j^{(k)}\right\} \in \{-1, 1\}$.

Received Signal. Assuming there are N_u active users the received signal is

$$r(t) = \sum_{k=1}^{N_u} \sqrt{A_k} s^{(k)}(t) \star h^{(k)}(t) + n(t) \tag{6.38}$$

where A_k represents the attenuation due to the path loss and $n(t)$ is a AWGN process with two-sided power spectral density $N_o/2$. The impulse response $h(t)$ of the channel depends on the channel model used. The most common model is that proposed by the IEEE 802.15.3a working group [**94**] which is a modification of the Saleh-Valenzuela model [**95**].

6.4.2. TH/CS System with TCH Codes.

TH with TCH Sequences. A TCH code was used as the pseudo-random TH code in 6.35. We considered TCH codes of length 4 and 16 dividing a frame into 4 and 16 slots (hops),

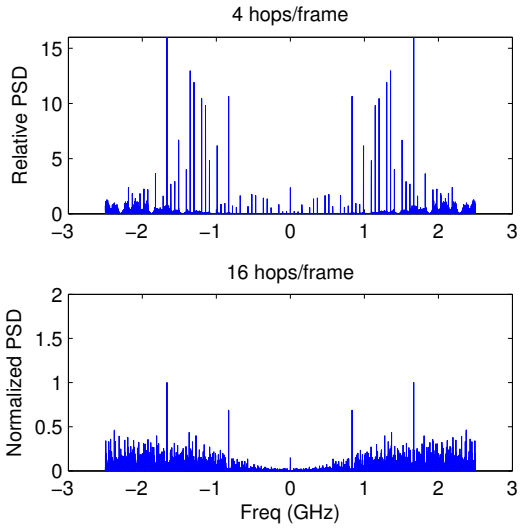


Figure 6.23. PSD comparison of a TH-PPM system.

respectively. A simulation of the transmitter system for a single user was made to generate the Power Spectrum Density (PSD) of the transmitted signal. Figure 6.23compares both cases. In the top graph the TH code hops 4 times in a frame while in the bottom graph the frame is divided into 16 slots (TH code with 16 hops). In the 4 hops/frame case (top graph) the PSD has strong peaks due to the weaker dithering effect of the short code while in the 16 hops/frame case (bottom graph) there is a spectral line reduction and more compact spectrum. The PSD amplitude in both graphs was normalized to the maximum PSD peak value of the 16 hops case.

A Welch spectrum estimate was further produced applying a Hamming window and computing a subsequent periodogram and the result is presented in Figure 6.24. The top curve corresponds to the 4 hops case while the smoother bottom curve corresponds to the 16 hops alternative illustrating the more effective dithering of the longer length code. In fact we are making the transmitted signal more random by using a 16-length code instead of the shorter one. The 6 dB difference matches well to the relative values of maximum amplitude depicted in Figure 6.23.

Assuming ideal free-space propagation, AWGN channel and perfect power control (the same amplitude for the users signals at the receiver), the received signal in the presence of N_U active users is

$$r(t) = \sum_{k=1}^{N_U} s_{TH}^{(k)}(t-\tau_k) + n(t) = \tag{6.39}$$

$$= s_{TH}^{(1)}(t-\tau_1) + \sum_{k=2}^{N_U} s_{TH}^{(k)}(t-\tau_k) + n(t) \tag{6.40}$$

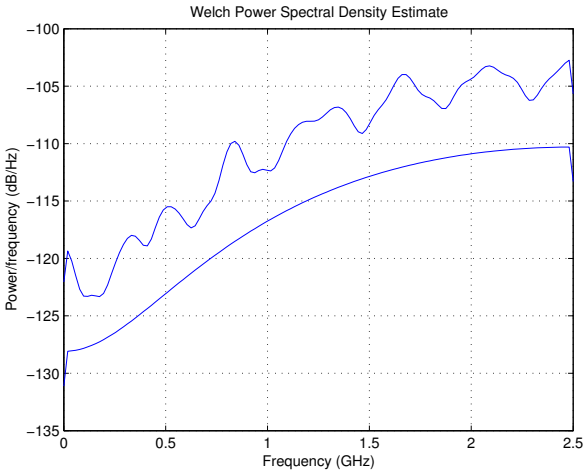


Figure 6.24. Walsh PSD of the TH-PPM system with TCH codes as hopping codes.

where τ_k, for $k = 1,\ldots,N_U$, is the delay associated to each user in an asynchronous multiple access system and $n(t)$ is a white Gaussian noise process as usual (assuming the noise from multi-user interference, MUI, is Gaussian). Considering user 1 as the receiver we are interested in we further assume $\tau_1 = 0$ and that we have perfect knowledge of its time hopping pattern $\left\{c_j^{(1)}\right\}$. The receiver is a single-user optimum correlator that employs $v(t) = p_{Tx}(t) - p_{Tx}(t-\delta)$ as the correlation mask. To be really accurate we should employ the received pulse $p_{Rx}(t)$ since the propagation and antenna effects can alter the transmitted pulse shape. However, as long as the pulse shape is reflected in the correlation mask we can consider the simplification $p_{Rx}(t) = p_{Tx}(t)$. In the presence of multiple pulses per symbol, two possible strategies can be adopted at the receiver: soft decision detection and hard decision detection.

In soft decision detection, the signal formed by N_S pulses is considered by the receiver as a single multi-pulse signal. The received signal is cross-correlated with a correlation mask, which is matched to the train of pulses representing the entire symbol. The decision statistic computes

$$\sum_{m=0}^{N_S-1} \int_{mT_F}^{(m+1)T_F} r(t)\, v\left(t - mT_F - c_m^{(1)} T_C\right) \mathrm{d}t \tag{6.41}$$

and decides upon the symbol transmitted comparing the result with the zero threshold.

In hard decision detection, the receiver implements N_S independent decisions over N_S pulses that represent one bit. The final decision is obtained by applying a simple majority criterion. Given the number of pulses falling over a threshold and comparing this number with the number of pulses falling below the same threshold, the estimated bit corresponds to the highest of these two numbers. An error occurs if more than half of the pulses are misinterpreted.

CS with TCH Sequences. IR-UWB systems can be characterized as an extension of traditional spread spectrum systems. One major difference resides in the radio channel which in the UWB case is extremely multipath rich. At the receiver the multipath components that are combined increase the total signal energy while those that are not contribute to inter-symbol interference (ISI). Among diversity techniques used to increase the received signal energy we find the rake receiver (exploiting time diversity) ideally with a number of fingers equal to the number of multipath components. Due to the high number of multipath contributions involved this ideal receiver is not feasible [**96**]. In practical implementations a reduced number of fingers is considered taking the L_r strongest propagation paths in the so called selective rake (Srake) or considering a further simplified approximation by taking the L_r first propagation paths, an alternative called partial rake (Prake). To obtain this *a-priori* information about the channel impulse response (CIR) channel estimation algorithms must be used. Here we do not consider estimation neither synchronization issues which are assumed perfectly acquired. The received signal for a single data bit may be defined as

$$r(t) = \sum_{n=1}^{L_r} a_n s(t-\tau_n) + n(t) \tag{6.42}$$

where L_r is the number of recovered paths (rake fingers), $s(t)$ the transmitted signal, $n(t)$ the Gaussian noise and $a_n = |a_n| e^{j\theta_n}$, τ_n, the gain and delay of the n^{th} multipath, respectively. Coherently combining the L_r components results in the decision variable

$$U_i = \sum_{n=1}^{L_r} a_n^* \int_0^{N_h T_C} r(t-\tau_n) w_i(t)\, dt \tag{6.43}$$

known as maximal ratio combining (MRC) scheme, where $N_h T_C$ is the data bit length and $w_i(t)$ the waveform, i.e. the train of pulses representing the data bit i at the receiver. If we restrict θ_n to have values in the set $\{0, \pi\}$ then equal gain combining (EGC) results.

6.4.3. Performance Results.

AWGN Channel. We performed a Monte Carlo simulation with the following parameters: the number of frames per symbol is $N_S = 2$ with each frame divided into $N_C = 16$ slots; the chip interval $T_C = 1.2$ ns and the pulse width $T_p = 1$ ns using an index of modulation $\delta = \frac{1}{5} T_p = 0.2$ ns; the bit rate is therefore $R_d \simeq 26$ Mbit/s. The results for the multi-user scenario, under chip synchronous multiple access[**97**],i.e. the random delays τ_k between users are integer multiples of the chip interval T_C, are presented in Figure 6.25. We started with a single user and evaluated both hard and soft decision detection. As expected, in the presence of Gaussian noise, soft decision outperforms hard decision. The theoretical BER for binary orthogonal PPM is also included. Note that there is a performance gain relative to the orthogonal case due to δ being close to the optimum value (the value that minimizes the cross-correlation $R(.)$ as discussed

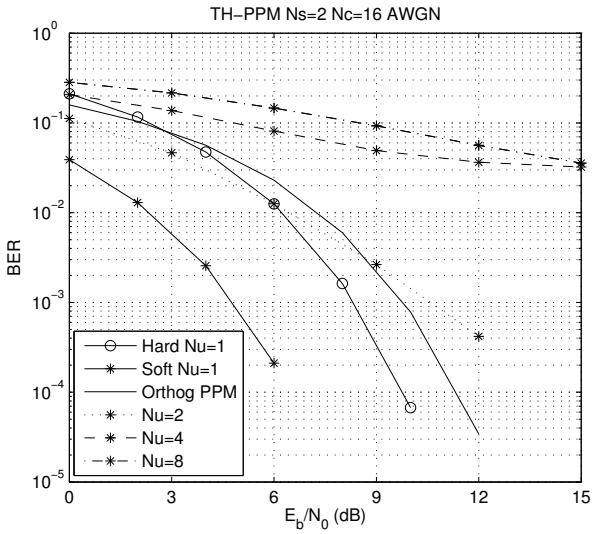


Figure 6.25. BER results for TH-PPM with TCH codes under a AWGN channel.

previously. We also plotted the BER curves for a varying number of active users, i.e., $N_U = 2, 4, 8$ and only for the soft decision detection. For $N_U \geq 4$ the MUI-error floor is reached for bit to noise ratio around 15 dB. It is clear that even for a small number of interferers this repetition coded TH-PPM system is limited.

In Figure 6.26 we increased the number of frames ($N_S = 4$) reducing the bit rate in half and compared the performance of the system in the multi-user case. For comparison the BER for ($N_S = 2$; $N_U = 4$) is also shown. The redundancy introduced by N_S permits more user simultaneous users. In fact, by decreasing the bit rate in half the system now allows two-times as much active users than the previous case.

The same type of simulations were executed for the DS-PAM (or equivalently DS-BPSK) system. In Figure 6.27 we evaluated the effect of increasing the number of code repetitions for each symbol. The case for $N_S = 1$ is not plotted because it matches with the theoretical BER curve for 2-PAM (solid line without markers). For the other cases, both hard and soft decision results are presented with a clear indication to favor the soft decision. The data bit rate R_d is decreased as N_S increases.

The multi-user case for the DS-PAM system with codes of 16 chips is presented in Figure 6.28. MUI error floors are reached for $N_U \geq 4$ and $\frac{E_b}{N_o} > 20$ dB. Theoretical symbol error rate for the multi-user scenario are generally based on the standard Gaussian approximation (GA) hypothesis, where the cumulative effect of all interfering contributions at the receiver is treated as an additive Gaussian noise with uniform PSD over the range of frequencies of interest. It seems that the GA hypothesis cease having validity when the number of pulses in the air is not sufficiently high to fill up the time dimension (associated with a low density of users), when the

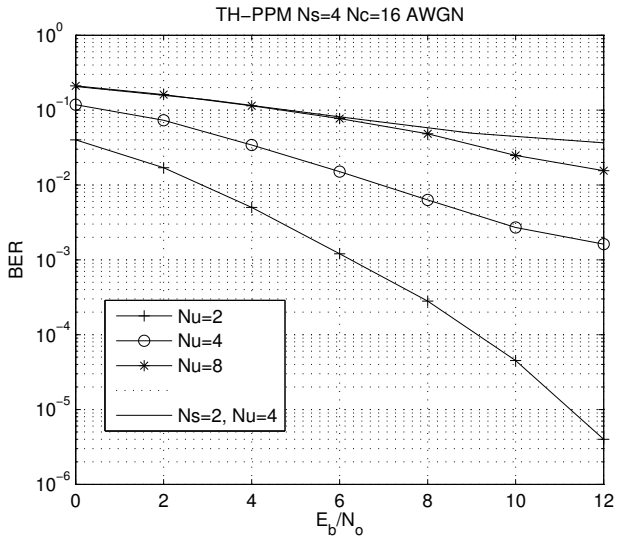


Figure 6.26. TH-PPM with increased number of frames per symbol.

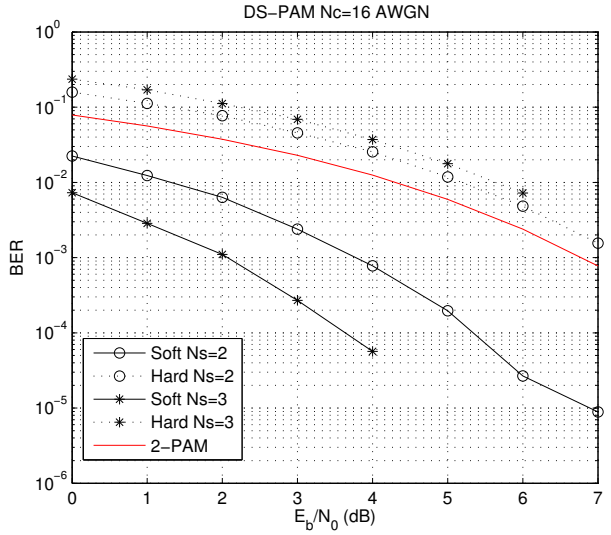


Figure 6.27. BER results for single-user DS-PAM with TCH codes under a AWGN channel.

transmitters are characterized by a low data rate, when the number of pulses per bit is low, there are dominant interferers, or combinations of the above conditions. Alternatives to the GA for computing the probability of bit error have been published (see [**98**][**99**] and references therein).

IEEE 802.15.3a channel. The UWB radio channel is similar to a wideband channel as may be experienced in spread spectrum or CDMA systems. The main distinct feature of the 'ultra' wideband channel model is the extremely multipath-rich channel profile. The reference model that we used in the following performance results is that adopted by the IEEE 802.15.3a study group [**100**]. In particular we will consider a Line-of-Sight (LOS) scenario, referenced

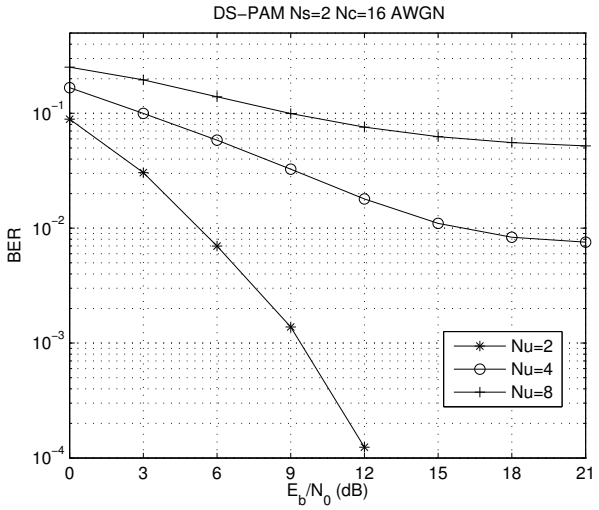


Figure 6.28. Multi-user DS-PAM with TCH 16-chip codes under a AWGN channel.

as CM1, with an average mean excess delay of 5 ns and a Non-Line-of-Sight (NLOS) scenario, referenced as CM2, with a mean excess delay of 10 ns.

For the TH-PPM system we performed a Monte Carlo simulation generating 100 channel realizations for each scenario and for each E_b/N_0 value with the following parameters: the number of frames per symbol is $N_S = 1$ with each frame divided into $N_C = 16$ slots; the chip interval $T_C = 3.2$ ns (the previous 1.2 ns value plus a guard space of 2 ns to help with ISI due to multipath delay spread) and the pulse width $T_p = 1$ ns using an index of modulation $\delta = \frac{1}{5}T_p = 0.2$ ns; the bit rate is therefore $R_d \simeq 20$ Mbit/s. In 6.29 (top set of curves) we present the simulation results. Srake means selective Rake where the 8 strongest power multipaths are used in the receiver mask whereas Prake means partial Rake, i.e., the first 8 multipaths are considered regardless of their power values. Note that either the selective Rake and partial Rake have similar performances (floor of around 1% error rate). This is due to the insufficient guard space as confirmed in the bottom set of curves. Here we increased the guard space, i.e. we increased the interval between pulses and therefore decreased the bit rate R_d to a value of approximately one third of that used in the top set of curves. Only the partial Rake was simulated to alleviate the receiver complexity. As expected the results for the NLOS case are worse than the LOS case due to the increased number of multipaths necessary to collect the same amount of energy.

For the DS-PAM system we performed a Monte Carlo simulation generating 100 channel realizations for each scenario and for each E_b/N_0 value with the following parameters: the chip interval $T_C = 3$ ns (the pulse width $T_p = 1$ ns plus a guard space of 2 ns to help with ISI due to multipath delay spread) and a 16 chip TCH code as spread-spectrum code. An 8 finger partial rake with MRC was used in the receiver. Figure 6.30 presents the results for one LOS and two

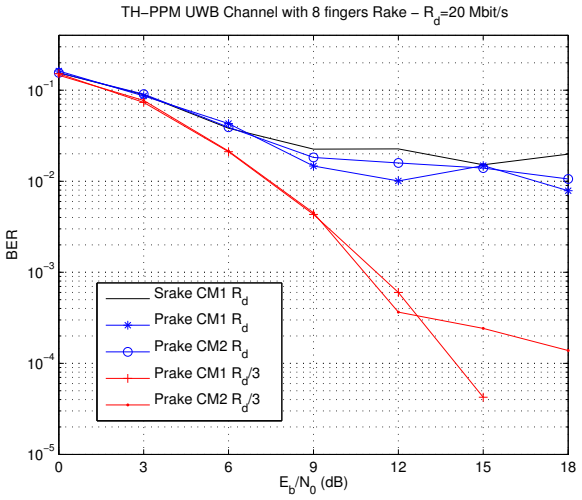


Figure 6.29. TH-PPM with 16 chip time-hopping TCH codes using a rake receiver with 8 fingers and considering a LOS (CM1) and a NLOS (CM2) channel model.

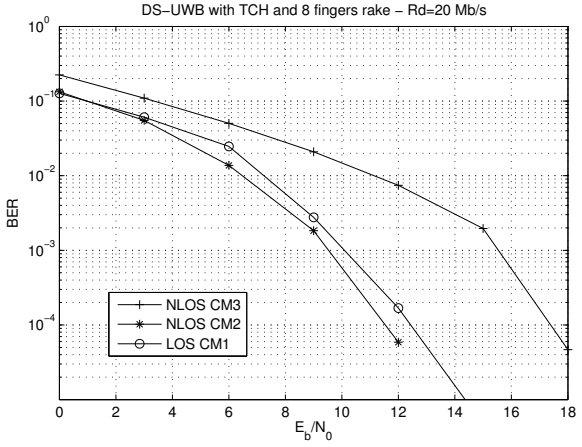


Figure 6.30. DS-PAM with 16 chip TCH spreading code using a partial rake receiver with 8 fingers and considering both LOS and NLOS UWB channel models.

NLOS channel scenarios. For the same data rate R_d there is approximately a 5 dB performance advantage of DS over TH (BER=10^{-2}). Comparing these results with those for the TH system we see that the DS system is much better. For approximately the same BER values the TH system needs to use a third of the data rate employed in the DS system.

TCH codes were also used in a combined spreading and coding operation. The results presented in Figure 6.31 used a TCH(16,4) code boosting the data rate to 80 Mb/s. The 16 chip TCH code used was a TCH(16,4) code. The error floors are reached for a E_b/N_0 value that correspond to SNR (per sample) close to 0. To improve the error correcting capability a Reed-Solomon code was also used in one of the simulations (for this case the combined code

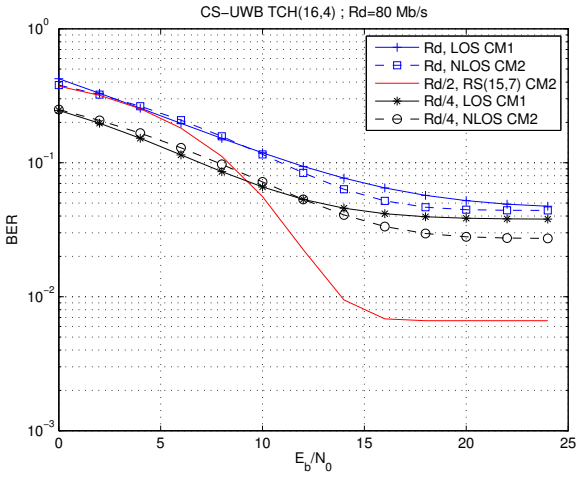


Figure 6.31. CS-TCH for UWB channels using a joint spreading and coding operation.

rate is $\frac{4}{16}\frac{7}{15} \approx 0.12$). For channel compensation the same channel estimate used for the previous DS system was employed.

.

CHAPTER 7

CONCLUSION

We focused the work of this thesis into the design of a class of cyclic codes inspired by TCH codewords. In order to accomplish such task we needed a deeper understanding of its algebraic structure. Thus we had to study abstract algebra and number theory. Only with this set of tools it was possible to pursue further developments not purely from a mathematical perspective but always with engineering applications in mind.

During our previous work with TCH codewords we began to note some relationships between codewords. For example, if we take a binary TCH codeword we can obtain another codeword if we first flip it around a middle word axis and then rotate it once to the left. Using other axes and similar flipping operations other codewords could be produced. We tested this method on several different codeword lengths and it still held. It was clear then that there was some kind of mathematical structure present but not completely understood from a formal and rigorous point of view.

Some initial questions were formulated to guide the investigation work that followed. Since the codewords from the TCH(16,5) code could be obtained from a single codeword through bit operations (although in an ad-hoc manner) could there be a structured and unified way to obtain such codewords from a seed codeword? Once the structure was known could we obtain a more efficient way to generate all the codewords in the time domain? Using the properties of the DFT could we extrapolate the results to the frequency domain? Although we started with binary sequences could we devise a method to include M-ary sequences as well? What about trying the generalize TCH to obtain the other power of two lengths not covered by the original TCH generating equation (section 3.6)? The quest for answering these questions drove the core work presented in this thesis, in particular in chapters 4 and 5. These sequences can be applied in several communication applications like DS-CDMA and UWB as discussed in chapter 6.

7.1. Synopsis

The fundamental concepts and facts about abstract algebra and number theory constitute the first part of the thesis. In chapter 2, after an introduction to basic material like set theory, mappings and partitions we focused on group theory. In chapter 3, we provided the context and introduced some definitions of coding theory. Then we addressed some core results from number theory since they are essential for understanding the theoretical basis of certain existent codes. We also discussed codes as sequences since sometimes we are not interested in the error-correcting capability of the codewords but are more focused on their random-like nature when they are used in spreading applications. We concluded chapter 3 by referring a class of cyclic codes, named TCH, that provided the motivation for the subsequent work presented in this thesis.

In chapter 4 we used group geometric properties and related some operations on codes as equivalent to the symmetries of the dihedral group. Exploring these properties allowed us to simplify the generation of codewords by saving on the necessary number of computations required. Moreover, we also presented an algebraic method to obtain binary generalized TCH codewords of length $N = 2^k$, $k = 1, 2, \ldots, 16$. By exploring Zech logarithm's properties as well as a group theoretic isomorphism this method is both faster and less complex than what was proposed before. In addition, it is valid for all relevant cases relating the codeword length N and not only those resulting from $N = p_i - 1$ for Fermat primes p_i. The method also derives the maximum set of all the codewords of a certain code bringing clear advantages in terms of code size and minimum distance.

In chapter 5, we focused mostly on group permutations as an efficient way to generate all codewords of a particular cyclic code. For binary sequences associated to sub-Pythagorean primes the method only requires the repeated application of 3 permutations (two for the time domain and one extra for the frequency domain) and a DFT operation, thus saving memory space and processing time. For general M-ary sequences the procedure may require, at most, M additional permutations.

After our sequence design efforts we can summarize some of the advantages obtained:

- the sequence period or codeword length is not limited to a power of two (and not equal to a Fermat prime minus 1),
- by construction, all codewords have a weight equal to half the length of the codeword,
- using the same algebraic generating procedure we can produce a larger number of codewords (better data rate, better error correction),
- the mathematical knowledge of the code structure permits not having a loose collection of codewords, but a codeword list with a cohesive structure, i.e. where one codeword is

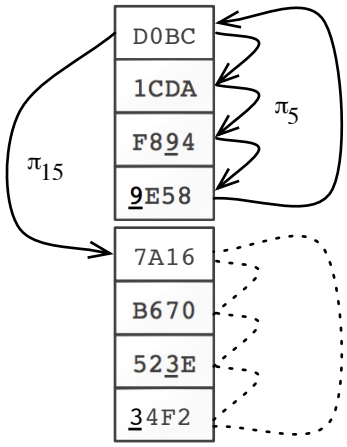


Figure 7.1. All the Sidel'nikov codewords from GF(17).

linked by appropriate permutations to another codeword, and this is of huge importance in terms of optimizing the coding/decoding operations,

- the algebraic generation procedure is not limited to binary but allows M-ary sequences as well,
- the link through permutations of time-domain codewords is nicely extended to their representation in the frequency-domain.

7.2. Future work

In Figure 7.1 we represent all the 8 Sidel'nikov codewords from GF(17). As previously described all these codewords can be obtained from the seed codeword "D0BC" (associated with the primitive element 3) by the application of two permutations, π_5 and π_{15}. The permutation π_5 with order 4 generate in sequence a group of 4 codewords, while the permutation π_{15} with order 2 transfers from one group to another.

Due to the rich structure involving these codewords we can identify a couple of patterns. First, note that in each column either all even or all odd decimal values (from the corresponding hexadecimal characters) are present. For example, in the most significant hexadecimal column we see all odd values, i.e., D, 1, F, 9, 7, B, 5, 3 from top to bottom. The next column, going from left to right, presents all the even values. The last two columns repeat this pattern, first the odd values then the even values. Secondly, a more relevant pattern is the appearance of the codewords "9" and "3" resulting from GF(5), implying that these two 4-bit codewords are present in the eight 16-bit codewords[1]. This motivated us to investigate the embedding of smaller codewords into larger length codewords. This type of embedding has great application

[1]Note that the boolean negation of these codewords, i.e. "C" and "6", are also present

Table 7.1. Cycle period for each bit on a 16-bit word upon application of permutation π_5.

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
#Cyc:	4	2	4	0	4	2	4	0	4	2	4	0	4	2	4	0
Bit:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0

Table 7.2. Cycle period patterns (at level 0) for each bit on a 256-bit word upon application of permutation π_5.

Bit:	255	254	253	252	251	250	249	248	247	246	245	244	243	242	241	240	$\cdots$
#Cyc:	64	32	64	16	64	32	64	8	64	32	64	16	64	32	64	4	
Bit:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	

Table 7.3. Cycle period patterns for 256-bit words (at level 1).

Bit:	252	248	244	240	236	232	228	224	220	216	212	208	204	200	196	192	$\cdots$
#Cyc:	16	8	16	4	16	8	16	2	16	8	16	4	16	8	16	0	
Bit:	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60	64	

Table 7.4. Cycle period patterns for 256-bit words (at level 2).

Bit:	240	224	208	192	176	160	144	128	112	96	80	64	48	32	16	0
#Cyc:	4	2	4	0	4	2	4	0	4	2	4	0	4	2	4	0
Bit:	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240	0

significance since one can transmit data with a larger codeword while at the same time searching for smaller codewords to track synchronism (assumed previously acquired).

The permutation π_5 applied to a 16-bit word involves 2 cycles of length 4 and 2 cycles of length 2:

$$(1,13,9,5)(2,10)(3,7,11,15)(6,14) \tag{7.1}$$

The length or period of the cycles associated to each of the 16 bits are presented in Table 7.1.

Note that the reflected codeword (third line of table 7.1) has the same cycle pattern as the original codeword. Note also that we have 4 fixed points at bit positions 0, 4, 8 and 12.

The same permutation applied to a 256-bit word is presented similarly in Table 7.2. Since there is symmetry the most significant bits are on the top row and the least significant bits are in the bottom row.

We now make groups of 4 bits since the pattern “64, 32, 64, x” is recurring and keep only one, i.e. the “x”, of those 4 bits. This is illustrated in Table 7.3.

Since there is another recurring pattern we now make groups of 16 bits and keep only one bit for each group as indicated in Table 7.4.

Table 7.5. Cycle periods (multiplied by 16) of 16-bit words

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
16 #:	64	32	64	$0 \equiv 16$	64	32	64	0	64	32	64	$0 \equiv 16$	64	32	64	0

Table 7.6. Alignment of cycle periods between the 16-bit word and the 256-bit word.

256:	255	254	253	252	...	131	130	129	128	127	126	125	124	...	3	2	1	0
#Cyc:	64	32	64	16	...	64	32	64	0	64	32	64	16	...	64	32	64	0
16:	15	14	13	12	...	11	10	9	8	7	6	5	4	...	3	2	1	0

How does a 16-bit word embed in a 256-bit word? What is a good mask? We want to find 8 16-bit words in a set of 128 256-bit words. Since dividing the entire code of 128 codewords into blocks of, say 16 codewords, we have 128/16=8 meaning that we should apply π_5 consecutively 16 times. If a cycle has length 64 then we need 4 turns of 16 consecutive applications of π_5. If we multiply by 16 the cycles presented in Table 7.1 we have for the 16-bit words the cycle structure of Table 7.5.

As indicated in Table 7.6 we can align the 16-bit codeword with the 256-bit codeword using the fixed points as anchors.

This reveals the good mask composed of 4 hexadecimal characters in the format x — xx — x, as illustrated in Table 7.7. Only the first codewords are presented but the complete listing can be easily obtained by applying the appropriate permutations.

Having identified this embedding nature we now need to study how to formally describe it. We envision an application where simultaneous data transmission and synchronization tracking is possible. While data is encoded and transmitted with a larger length codeword a smaller length embedded codeword is searched during reception to keep track of synchronism.

Another envisioned evolution of this work is the construction of new sequences via the kronecker product. As such, using sequences of length 16 and 256 it is possible to generate easily sequences of length 4096 = 16 x 256. Other combinations are also possible.

Table 7.7. The first 256-bit codewords from GF(257) illustrating the embedding of the 16-bit codewords from GF(17).

dc0bec19f9735494195aea73c648a2d0bc208e6d36041f9af8ff37959ac50adc
a350e20f1ab9d2d4e0bed5d5ccdb1268863b1ccdfdfc50a4fc3d901b4a24bf20
44fa48b5de9282b83cafdc5993634d523fcf27399edd40d29028385df08e14ee
6c37d127c898687ab1551c013cdc1ca4e0dadcd3aadbffb01686988d63bd72c6
b 4d99c5df80365525288eb291e48f41 67 af48e5b8304883e3fe72a95ded99cf 0
5650927d073dd8c400d2e4a1da997bacc117981da0e43caaec9dd36bd638be7e
55c01bd95aeef0b868e98edb1437ca72360d72fb1c49848690b4441f9d1aadfe
58d3f423cc5cf9eac8757015ad32089a188a33c1fab7f68c0594decd22f53c9e
f0cb0081bd63ba5ad7105e9d5c4ce610ba64cedfd85ebb7c1e1127d1a8ab0cb4
6b3c962df43b98081cfaf7b5ae8ff0ccccb54841f17414da8a9912f5c278d306
96860ed158ba24d8fecfd4b357af9b288319417f30fd4c549ce2109fbc4a6878
289d3db3441cdc54f9a97c01e0325c6a06de32a5aad78194fedcdaef31d3d882
98db80b3eee1af5c92440d7b7a94da567e1cf81717caee38df41a44530a91c98
ea38fc796103986c4274baed189fd5f895fd589bc410f62ec6992ba796d49204
9302dd9114ac02ca5cdfa2f356d1c69e586dbc7f34215cde0c2ac0fbb9dc2b38
38a578cf1cd89100e873f08bc1567e7cd6567fad08b53684abb89cdb4c97e092
9c8b26f79f953e98f97c8e3daa4402d67c2aee01d248d7949853f959746308d8
ad3a844d5c57da34c636911968bd94a0a0f9d0879bb8726cf21d7edfcee813c0
a47e0c1996fe40da328b3af5db8585de5de9e91df41308321cae54799aca56e0
60d1ddc920d844b03d179007fa9e7a6cc61658156ab97bd2b0ee9ca38dddbca6
5 ad53ef19a8da996b6ec8b0fde40147 23 6faae5d4b08c4707981c819b453fc1 e
d89eb03d835dbe284416c0497439bde665d192f78eac7aee8251df29d2b0589c
⋮

Appendices

APPENDIX A

GROUPS CHARACTERIZING GEOMETRIC SYMMETRIES

A.1. Permutation Groups

Permutation groups are central to the study of geometric symmetries and to Galois theory, the study of finding solutions of polynomial equations. They also provide abundant examples of nonabelian groups. Let us recall for a moment the symmetries of the equilateral triangle $\triangle ABC$ from section 2.5. The symmetries actually consist of permutations of the three vertices, where a *permutation* of the set $S = \{A, B, C\}$ is a one-to-one and onto map $\pi : S \to S$. The three vertices have the following six permutations.

$$\begin{pmatrix} A & B & C \\ A & B & C \end{pmatrix} \begin{pmatrix} A & B & C \\ C & A & B \end{pmatrix} \begin{pmatrix} A & B & C \\ B & C & A \end{pmatrix}$$

$$\begin{pmatrix} A & B & C \\ A & C & B \end{pmatrix} \begin{pmatrix} A & B & C \\ C & B & A \end{pmatrix} \begin{pmatrix} A & B & C \\ B & A & C \end{pmatrix}$$

We have used the array

$$\begin{pmatrix} A & B & C \\ B & C & A \end{pmatrix}$$

to denote the permutation that sends A to B, B to C, and C to A. That is,

$$\begin{aligned} A &\mapsto B \\ B &\mapsto C \\ C &\mapsto A. \end{aligned}$$

The symmetries of a triangle form a group. In this section we will study groups of this type.

A.1.1. Definitions and Notation. In general, the permutations of a set X form a group S_X. If X is a finite set, we can assume $X = \{1, 2, \ldots, n\}$. In this case we write S_n instead of S_X. The following theorem says that S_n is a group. We call this group the *symmetric group on n letters*.

THEOREM 92. *The symmetric group on n letters, S_n, is a group with n! elements, where the binary operation is the composition of maps.*

A subgroup of S_n is called a *permutation group*.

EXAMPLE. Consider the subgroup G of S_5 consisting of the identity permutation *id* and the permutations

$$\begin{aligned}
\sigma &= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 5 & 4 \end{pmatrix} \\
\tau &= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 2 & 1 & 4 & 5 \end{pmatrix} \\
\mu &= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 2 & 1 & 5 & 4 \end{pmatrix}.
\end{aligned}$$

The following table tells us how to multiply elements in the permutation group G.

$\circ$	id	σ	τ	μ
id	id	σ	τ	μ
σ	σ	id	μ	τ
τ	τ	μ	id	σ
μ	μ	τ	σ	id

REMARK. Though it is natural to multiply elements in a group from left to right, functions are composed from right to left. Let σ and τ be permutations on a set X. To compose σ and τ as functions, we calculate $(\sigma \circ \tau)(x) = \sigma(\tau(x))$. That is, we do τ first, then σ. There are several ways to approach this inconsistency. *We will adopt the convention of multiplying permutations right to left. To compute $\sigma\tau$, do τ first and then σ.* That is, by $\sigma\tau(x)$ we mean $\sigma(\tau(x))$. (Another way of solving this problem would be to write functions on the right; that is, instead of writing $\sigma(x)$, we could write $(x)\sigma$. We could also multiply permutations left to right to agree with the usual way of multiplying elements in a group. Certainly all of these methods have been used.

EXAMPLE. Permutation multiplication is not usually commutative. Let

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 1 & 2 & 3 \end{pmatrix}$$
$$\tau = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 4 & 3 \end{pmatrix}.$$

Then

$$\sigma\tau = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 4 & 3 & 2 \end{pmatrix},$$

but

$$\tau\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 4 \end{pmatrix}.$$

Cycle Notation. The notation that we have used to represent permutations up to this point is cumbersome, to say the least. To work effectively with permutation groups, we need a more streamlined method of writing down and manipulating permutations. A permutation $\sigma \in S_X$ is a *cycle of length k* if there exist elements $a_1, a_2, \ldots, a_k \in X$ such that

$$\begin{aligned} \sigma(a_1) &= a_2 \\ \sigma(a_2) &= a_3 \\ &\vdots \\ \sigma(a_k) &= a_1 \end{aligned}$$

and $\sigma(x) = x$ for all other elements $x \in X$. We will write $(a_1, a_2, \ldots, a_k)$ to denote the cycle σ. Cycles[1] are the building blocks of all permutations.

EXAMPLE. The permutation

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 6 & 3 & 5 & 1 & 4 & 2 & 7 \end{pmatrix} = (162354)$$

is a cycle of length 6, whereas

$$\tau = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 4 & 2 & 3 & 5 & 6 \end{pmatrix} = (243)$$

[1]Lagrange first thought of permutations as functions from a set to itself, but it was Cauchy who developed the basic theorems and notation for permutations. He was the first to use cycle notation.

is a cycle of length 3. Not every permutation is a cycle. Consider the permutation

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 2 & 4 & 1 & 3 & 6 & 5 \end{pmatrix} = (1243)(56).$$

This permutation actually contains a cycle of length 2 and a cycle of length 4.

EXAMPLE. It is very easy to compute products of cycles. Suppose that

$$\begin{aligned} \sigma &= (1352) \\ \tau &= (256). \end{aligned}$$

We can think of σ as

$$\begin{aligned} 1 &\mapsto 3 \\ 3 &\mapsto 5 \\ 5 &\mapsto 2 \\ 2 &\mapsto 1 \end{aligned}$$

and τ as

$$\begin{aligned} 2 &\mapsto 5 \\ 5 &\mapsto 6 \\ 6 &\mapsto 2 \end{aligned}$$

Hence, $\sigma\tau = (1356)$. If $\mu = (1634)$, then $\sigma\mu = (1652)(34)$.

Two cycles in S_X, $\sigma = (a_1, a_2, \ldots, a_k)$ and $\tau = (b_1, b_2, \ldots, b_l)$, are *disjoint* if $a_i \neq b_j$ for all i and j.

EXAMPLE. The cycles (135) and (27) are disjoint; however, the cycles (135) and (347) are not. Calculating their products, we find that

$$\begin{aligned} (135)(27) &= (135)(27) \\ (135)(347) &= (13475). \end{aligned}$$

The product of two cycles that are not disjoint may reduce to something less complicated; the product of disjoint cycles cannot be simplified.

PROPOSITION 93. *Let σ and τ be two disjoint cycles in S_X. Then $\sigma\tau = \tau\sigma$.*

PROOF. Let $\sigma = (a_1, a_2, \ldots, a_k)$ and $\tau = (b_1, b_2, \ldots, b_l)$. We must show that $\sigma\tau(x) = \tau\sigma(x)$ for all $x \in X$. If x is neither $\{a_1, a_2, \ldots, a_k\}$ nor $\{b_1, b_2, \ldots, b_l\}$, then both σ and τ fix x. That is, $\sigma(x) = x$ and $\tau(x) = x$. Hence,

$$\sigma\tau(x) = \sigma(\tau(x)) = \sigma(x) = x = \tau(x) = \tau(\sigma(x)) = \tau\sigma(x).$$

Do not forget that we are multiplying permutations right to left, which is the opposite of the order in which we usually multiply group elements. Now suppose that $x \in \{a_1, a_2, \ldots, a_k\}$. Then $\sigma(a_i) = a_{(i \bmod k)+1}$; that is,

$$\begin{aligned} a_1 &\mapsto a_2 \\ a_2 &\mapsto a_3 \\ &\vdots \\ a_{k-1} &\mapsto a_k \\ a_k &\mapsto a_1. \end{aligned}$$

However, $\tau(a_i) = a_i$ since σ and τ are disjoint. Therefore,

$$\sigma\tau(a_i) = \sigma(\tau(a_i)) = \sigma(a_i) = a_{(i \bmod k)+1} = \tau(a_{(i \bmod k)+1}) = \tau(\sigma(a_i)) = \tau\sigma(a_i).$$

Similarly, if $x \in \{b_1, b_2, \ldots, b_l\}$, then σ and τ also commute. □

THEOREM 94. *Every permutation in S_n can be written as the product of disjoint cycles.*

EXAMPLE. Let

$$\begin{aligned} \sigma &= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 6 & 4 & 3 & 1 & 5 & 2 \end{pmatrix} \\ \tau &= \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 2 & 1 & 5 & 6 & 4 \end{pmatrix}. \end{aligned}$$

Using cycle notation, we can write

$$\begin{aligned} \sigma &= (1624) \\ \tau &= (13)(456) \\ \sigma\tau &= (136)(245) \\ \tau\sigma &= (143)(256). \end{aligned}$$

REMARK. From this point forward we will find it convenient to use cycle notation to represent permutations. When using cycle notation, we often denote the identity permutation by (1).

A.1.2. Transpositions.

DEFINITION 95. A transposition is the simplest permutation i.e. a cycle of length 2.

Since

$$(a_1, a_2, \ldots, a_n) = (a_1 a_n)(a_1 a_{n-1}) \cdots (a_1 a_3)(a_1 a_2),$$

any cycle can be written as the product of transpositions, leading to the following proposition.

PROPOSITION 96. *Any permutation of a finite set containing at least two elements can be written as the product of transpositions.*

EXAMPLE. Consider the permutation

$$(16)(253) = (16)(23)(25) = (16)(45)(23)(45)(25).$$

As we can see, there is no unique way to represent permutation as the product of transpositions. For instance, we can write the identity permutation as $(12)(12)$, as $(13)(24)(13)(24)$, and in many other ways. However, as it turns out, no permutation can be written as the product of both an even number of transpositions and an odd number of transpositions. For instance, we could represent the permutation (16) by

$$(23)(16)(23)$$

or by

$$(35)(16)(13)(16)(13)(35)(56),$$

but (16) will always be the product of an odd number of transpositions.

LEMMA 97. *If the identity is written as the product of r transpositions,*

$$id = \tau_1 \tau_2 \cdots \tau_r,$$

then r is an even number.

THEOREM 98. *If a permutation σ can be expressed as the product of an even number of transpositions, then any other product of transpositions equaling σ must also contain an even number of transpositions. Similarly, if σ can be expressed as the product of an odd number of transpositions, then any other product of transpositions equaling σ must also contain an odd number of transpositions.*

In light of Theorem 98, we define a permutation to be *even* if it can be expressed as an even number of transpositions and *odd* if it can be expressed as an odd number of transpositions.

A.2. Dihedral Groups

Another special type of permutation group is the dihedral group.

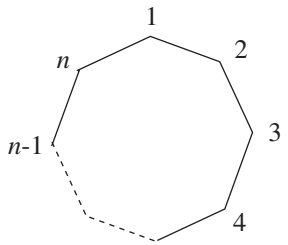


Figure A.1. A regular n-gon

Recall the symmetry group of an equilateral triangle in section 2.5. Such groups consist of the rigid motions of a regular n-sided polygon or n-gon. For $n = 3, 4, \ldots$, we define the *nth dihedral group* to be the group of rigid motions of a regular n-gon. We will denote this group by D_n. We can number the vertices of a regular n-gon by $1, 2, \ldots, n$ (Figure A.1). Notice that there are exactly n choices to replace the first vertex. If we replace the first vertex by k, then the second vertex must be replaced either by vertex $k+1$ or by vertex $k-1$; hence, there are $2n$ possible rigid motions of the n-gon. We summarize these results in the following theorem.

THEOREM 99. *The dihedral group, D_n, is a subgroup of S_n of order $2n$.*

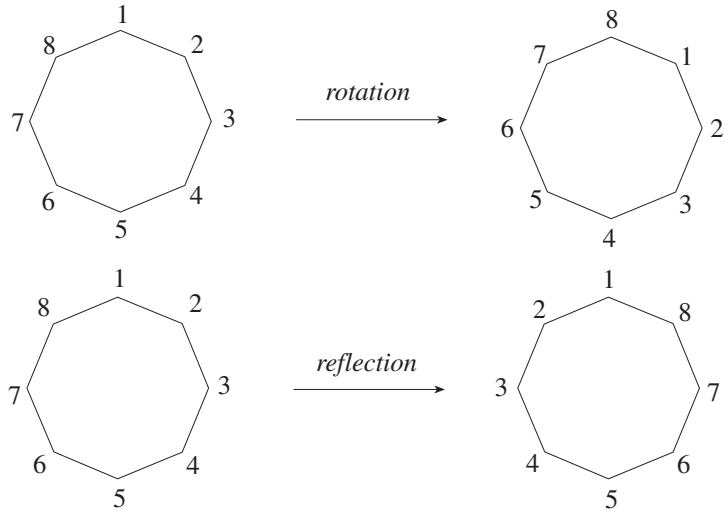


Figure A.2. Rotations and reflections of a regular n-gon

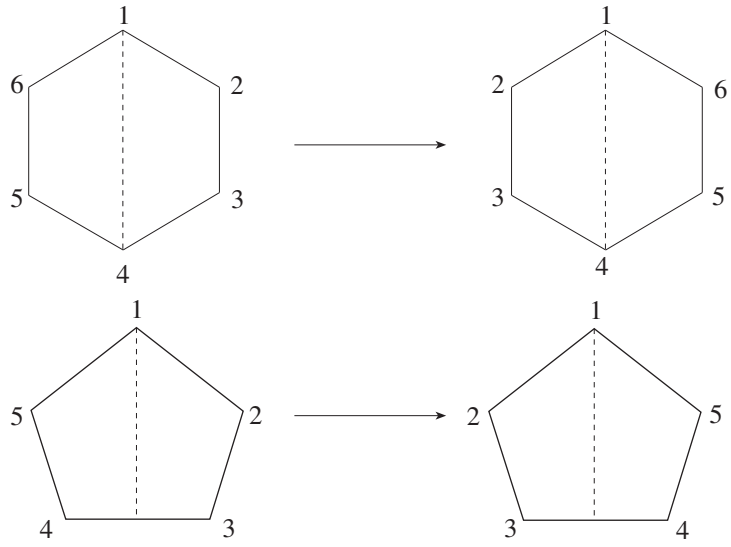


Figure A.3. Types of reflections of a regular n-gon

THEOREM 100. *The group D_n, $n \geq 3$, consists of all products of the two elements r and s, satisfying the relations*

$$\begin{aligned} r^n &= id \\ s^2 &= id \\ srs &= r^{-1}. \end{aligned}$$

APPENDIX B

LINEAR CODES

B.1. Codes as Groups

To gain more knowledge of a particular code and develop more efficient techniques of encoding, decoding, and error detection, we need to add additional structure to our codes. One way to accomplish this is to require that the code also be a group. A *group code* is a code that is also a subgroup of $\mathbb{Z}_2^n$. To check that a code is a group code, we need only verify one thing. If we add any two elements in the code, the result must be an n-tuple that is again in the code. It is not necessary to check that the inverse of the n-tuple is in the code, since every codeword is its own inverse, nor is it necessary to check that $\mathbf{0}$ is a codeword. For instance,

$$(11000101) + (11000101) = (00000000).$$

EXAMPLE. Suppose that we have a code that consists of the following 7-tuples:

$$\begin{array}{cccc} (0000000) & (0001111) & (0010101) & (0011010) \\ (0100110) & (0101001) & (0110011) & (0111100) \\ (1000011) & (1001100) & (1010110) & (1011001) \\ (1100101) & (1101010) & (1110000) & (1111111). \end{array}$$

It is a straightforward though tedious task to verify that this code is also a subgroup of $\mathbb{Z}_2^7$ and, therefore, a group code. This code is a single error-detecting and single error-correcting code, but it is a long and tedious process to compute all of the distances between pairs of codewords to determine that $d_{\min} = 3$. It is much easier to see that the minimum weight of all the nonzero codewords is 3. As we will soon see, this is no coincidence. However, the relationship between weights and distances in a particular code is heavily dependent on the fact that the code is a group.

LEMMA 101. *Let* $\mathbf{x}$ *and* $\mathbf{y}$ *be binary n-tuples. Then* $w(\mathbf{x}+\mathbf{y}) = d(\mathbf{x},\mathbf{y})$.

THEOREM 102. *Let* $d_{\min}$ *be the minimum distance for a group code C. Then* $d_{\min}$ *is the minimum of all the nonzero weights of the nonzero codewords in C. That is,*

$$d_{\min} = \min\{w(\mathbf{x}) : \mathbf{x} \neq \mathbf{0}\}.$$

PROOF. Observe that

$$\begin{aligned} d_{\min} &= \min\{d(\mathbf{x},\mathbf{y}) : \mathbf{x} \neq \mathbf{y}\} \\ &= \min\{d(\mathbf{x},\mathbf{y}) : \mathbf{x}+\mathbf{y} \neq \mathbf{0}\} \\ &= \min\{w(\mathbf{x}+\mathbf{y}) : \mathbf{x}+\mathbf{y} \neq \mathbf{0}\} \\ &= \min\{w(\mathbf{z}) : \mathbf{z} \neq \mathbf{0}\}. \end{aligned}$$

□

From the last example, it is now easy to check that the minimum nonzero weight is 3; hence, the code does indeed detect and correct all single errors. We have now reduced the problem of finding "good" codes to that of generating group codes. One easy way to generate group codes is to employ a bit of matrix theory. Define the *inner product* of two binary n-tuples to be

$$\mathbf{x} \cdot \mathbf{y} = x_1 y_1 + \cdots + x_n y_n,$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)^{\mathrm{T}}$ are column vectors.[1] For example, if $\mathbf{x} = (011001)^{\mathrm{T}}$ and $\mathbf{y} = (110101)^{\mathrm{T}}$, then $\mathbf{x} \cdot \mathbf{y} = 0$. We can also look at an inner product as the product of a row matrix with a column matrix; that is,

$$\begin{aligned} \mathbf{x} \cdot \mathbf{y} &= \mathbf{x}^{\mathrm{T}}\mathbf{y} \\ &= \begin{pmatrix} x_1 & x_2 & \cdots & x_n \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \\ &= x_1 y_1 + x_2 y_2 + \cdots + x_n y_n. \end{aligned}$$

EXAMPLE. Suppose that the words to be encoded consist of all binary 3-tuples and that our encoding scheme is even-parity. To encode an arbitrary 3-tuple, we add a fourth bit to obtain an even number of 1's. Notice that an arbitrary n-tuple $\mathbf{x} = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$ has an even number of 1's exactly when $x_1 + x_2 + \cdots + x_n = 0$; hence, a 4-tuple $\mathbf{x} = (x_1, x_2, x_3, x_4)^{\mathrm{T}}$ has an even number

[1] Since we will be working with matrices, we will write binary n-tuples as column vectors for the remainder of this section.

of 1's if $x_1+x_2+x_3+x_4=0$, or

$$\mathbf{x}\cdot\mathbf{1}=\mathbf{x}^{\mathrm{T}}\mathbf{1}=\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \end{pmatrix}\begin{pmatrix} 1\\1\\1\\1 \end{pmatrix}=0.$$

This example leads us to hope that there is a connection between matrices and coding theory.

Let $\mathbb{M}_{m\times n}(\mathbb{Z}_2)$ denote the set of all $m\times n$ matrices with entries in $\mathbb{Z}_2$. We do matrix operations as usual except that all our addition and multiplication operations occur in $\mathbb{Z}_2$. Define the *null space*of a matrix $H\in\mathbb{M}_{m\times n}(\mathbb{Z}_2)$ to be the set of all binary n-tuples $\mathbf{x}$ such that $H\mathbf{x}=\mathbf{0}$. We denote the null space of a matrix H by $\mathrm{Null}(H)$.

EXAMPLE. Suppose that

$$H=\begin{pmatrix} 0 & 1 & 0 & 1 & 0\\ 1 & 1 & 1 & 1 & 0\\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

For a 5-tuple $\mathbf{x}=(x_1,x_2,x_3,x_4,x_5)^{\mathrm{T}}$ to be in the null space of H, $H\mathbf{x}=\mathbf{0}$. Equivalently, the following system of equations must be satisfied:

$$\begin{aligned} x_2+x_4 &= 0\\ x_1+x_2+x_3+x_4 &= 0\\ x_3+x_4+x_5 &= 0. \end{aligned}$$

The set of binary 5-tuples satisfying these equations is

$$(00000)\quad(11110)\quad(10101)\quad(01011).$$

This code is easily determined to be a group code.

B.2. Efficient Decoding

We are now at the stage where we are able to generate linear codes that detect and correct errors fairly easily, but it is still a time-consuming process to decode a received n-tuple and determine which is the closest codeword, because the received n-tuple must be compared to each possible codeword to determine the proper decoding. This can be a serious impediment if the code is very large.

EXAMPLE. Given the binary matrix

$$H = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

and the 5-tuples $\mathbf{x} = (11011)^{\mathrm{T}}$ and $\mathbf{y} = (01011)^{\mathrm{T}}$, we can compute

$$H\mathbf{x} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

and

$$H\mathbf{y} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}.$$

Hence, $\mathbf{x}$ is a codeword and $\mathbf{y}$ is not, since $\mathbf{x}$ is in the null space and $\mathbf{y}$ is not. Notice that $H\mathbf{x}$ is identical to the first column of H. In fact, this is where the error occurred. If we flip the first bit in $\mathbf{y}$ from 0 to 1, then we obtain $\mathbf{x}$.

If H is an $m \times n$ matrix and $\mathbf{x} \in \mathbb{Z}_2^n$, then we say that the *syndrome* of $\mathbf{x}$ is $H\mathbf{x}$. The following proposition allows the quick detection and correction of errors.

PROPOSITION 103. *Let the $m \times n$ binary matrix H determine a linear code and let $\mathbf{x}$ be the received n-tuple. Write $\mathbf{x}$ as $\mathbf{x} = \mathbf{c} + \mathbf{e}$, where $\mathbf{c}$ is the transmitted codeword and $\mathbf{e}$ is the transmission error. Then the syndrome $H\mathbf{x}$ of the received codeword $\mathbf{x}$ is also the syndrome of the error $\mathbf{e}$.*

PROOF. $H\mathbf{x} = H(\mathbf{c}+\mathbf{e}) = H\mathbf{c} + H\mathbf{e} = \mathbf{0} + H\mathbf{e} = H\mathbf{e}$. □

This proposition tells us that the syndrome of a received word depends solely on the error and not on the transmitted codeword. The proof of the following theorem follows immediately from Proposition 103 and from the fact that $H\mathbf{e}$ is the ith column of the matrix H.

THEOREM 104. *Let $H \in \mathbb{M}_{m\times n}(\mathbb{Z}_2)$ and suppose that the linear code corresponding to H is single error-correcting. Let $\mathbf{r}$ be a received n-tuple that was transmitted with at most one error. If the syndrome of $\mathbf{r}$ is $\mathbf{0}$, then no error has occurred; otherwise, if the syndrome of $\mathbf{r}$ is equal to some column of H, say the ith column, then the error has occurred in the ith bit.*

EXAMPLE. Consider the matrix

$$H = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

and suppose that the 6-tuples $\mathbf{x} = (111110)^{\mathrm{t}}$, $\mathbf{y} = (111111)^{\mathrm{t}}$, and $\mathbf{z} = (010111)^{\mathrm{t}}$ have been received. Then

$$H\mathbf{x} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, H\mathbf{y} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}, H\mathbf{z} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}.$$

Hence, $\mathbf{x}$ has an error in the third bit and $\mathbf{z}$ has an error in the fourth bit. The transmitted codewords for $\mathbf{x}$ and $\mathbf{z}$ must have been (110110) and (010011), respectively. The syndrome of $\mathbf{y}$ does not occur in any of the columns of the matrix H, so multiple errors must have occurred to produce $\mathbf{y}$.

Coset Decoding. We can use group theory to obtain another way of decoding messages. A linear code C is a subgroup of $\mathbb{Z}_2^n$. *Coset* or *standard decoding* uses the cosets of C in $\mathbb{Z}_2^n$ to implement maximum-likelihood decoding. Suppose that C is an (n,m)-linear code. A coset of C in $\mathbb{Z}_2^n$ is written in the form $\mathbf{x} + C$, where $\mathbf{x} \in \mathbb{Z}_2^n$. By Lagrange's Theorem, there are 2^{n-m} distinct cosets of C in $\mathbb{Z}_2^n$.

EXAMPLE. Let C be the $(5,3)$-linear code given by the parity-check matrix

$$H = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

The code consists of the codewords

$$(00000) \quad (01101) \quad (10011) \quad (11110).$$

There are $2^{5-2} = 2^3$ cosets of C in $\mathbb{Z}_2^5$, each with order $2^2 = 4$. These cosets are listed in Table B.1.

Our task is to find out how knowing the cosets might help us to decode a message. Suppose that $\mathbf{x}$ was the original codeword sent and that $\mathbf{r}$ is the n-tuple received. If $\mathbf{e}$ is the transmission error, then $\mathbf{r} = \mathbf{e} + \mathbf{x}$ or, equivalently, $\mathbf{x} = \mathbf{e} + \mathbf{r}$. However, this is exactly the statement that $\mathbf{r}$ is an element in the coset $\mathbf{e} + C$. In maximum-likelihood decoding we expect the error $\mathbf{e}$ to be as small as possible; that is, $\mathbf{e}$ will have the least weight. An n-tuple of least weight in a coset

Table B.1. Cosets of C

	Cosets
C	(00000) (01101) (10011) (11110)
(10000) + C	(10000) (11101) (00011) (01110)
(01000) + C	(01000) (00101) (11011) (10110)
(00100) + C	(00100) (01001) (10111) (11010)
(00010) + C	(00010) (01111) (10001) (11100)
(00001) + C	(00001) (01100) (10010) (11111)
(10100) + C	(00111) (01010) (10100) (11001)
(00110) + C	(00110) (01011) (10101) (11000)

is called a *coset leader*. Once we have determined a coset leader for each coset, the decoding process becomes a task of calculating $\mathbf{r}+\mathbf{e}$ to obtain $\mathbf{x}$.

EXAMPLE. In Table B.1, notice that we have chosen a representative of the least possible weight for each coset. These representatives are coset leaders. Now suppose that $\mathbf{r} = (01111)$ is the received word. To decode $\mathbf{r}$, we find that it is in the coset $(00010)+C$; hence, the originally transmitted codeword must have been $(01101) = (01111)+(00010)$.

A potential problem with this method of decoding is that we might have to examine every coset for the received codeword. The following proposition gives a method of implementing coset decoding. It states that we can associate a syndrome with each coset; hence, we can make a table that designates a coset leader corresponding to each syndrome. Such a list is called a *decoding table*.

PROPOSITION 105. *Let C be an (n,k)-linear code given by the matrix H and suppose that $\mathbf{x}$ and $\mathbf{y}$ are in $\mathbb{Z}_2^n$. Then $\mathbf{x}$ and $\mathbf{y}$ are in the same coset of C if and only if $H\mathbf{x} = H\mathbf{y}$. That is, two n-tuples are in the same coset if and only if their syndromes are the same.*

PROOF. Two n-tuples $\mathbf{x}$ and $\mathbf{y}$ are in the same coset of C exactly when $\mathbf{x}-\mathbf{y} \in C$; however, this is equivalent to $H(\mathbf{x}-\mathbf{y}) = 0$ or $H\mathbf{x} = H\mathbf{y}$. □

EXAMPLE. Table B.2 is a decoding table for the code C given in Example 18. If $\mathbf{x} = (01111)$ is received, then its syndrome can be computed to be

$$H\mathbf{x} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}.$$

Examining the decoding table, we determine that the coset leader is (00010). It is now easy to decode the received codeword.

Given an (n,k)-block code, the question arises of whether or not coset decoding is a manageable scheme. A decoding table requires a list of cosets and syndromes, one for each of the 2^{n-k} cosets of C. Suppose that we have a $(32,24)$-block code. We have a huge number of codewords, 2^{24}, yet there are only $2^{32-24} = 2^8 = 256$ cosets.

Table B.2. Syndromes for each coset

Syndrome	Coset Leader
(000)	(00000)
(001)	(00001)
(010)	(00010)
(011)	(10000)
(100)	(00100)
(101)	(01000)
(110)	(00110)
(111)	(10100)

BIBLIOGRAPHY

[1] T. W. Judson, *Abstract Algebra: Theory and Applications*. Virginia Commonwealth University Mathematics, 2011.

[2] M. Sauter, *Communication Systems for the Mobile Information Society*. John Wiley and Sons, 2006.

[3] S. Lin and D. Costello Jr., *Error Control Coding: Fundamentals and Applications*. Prentice-Hall, 1983.

[4] J. Forney, G.D., "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268 – 278, march 1973.

[5] F. B. Cercas, M. Tomlinson, and A. Albuquerque, "Tch: A new family of cyclic codes length 2^m," in *IEEE International Symposium on Information Theory (ISIT '93)*, 1993, p. 198.

[6] F. B. Cercas, "A new family of codes for simple receiver implementation," Ph.D. dissertation, Technical University of Lisbon, Inst. Superior Técnico, March 1996.

[7] F. B. Cercas and A. Correia, *Third Generation Mobile Communication Systems*. Artech House, 2000, ch. 5.7.3, pp. 151–153.

[8] F. Cercas and W. Krewel, *Sattelite personal communications for future generation systems, Section 4.4 Novel Coding Techniques: TCH Codes*. Springer, 2002, final report: cost 252 action 4: "Air interface aspects".

[9] A. P. Almeida, R. Dinis, and F. Cercas, "On the use of tch codes in ultrawide band systems," in *Signal Processing, Pattern Recognition, and Applications*, no. 721-064. Acta Press, 2011.

[10] F. Cercas, J. Silva, N. Souto, and R. Dinis, "Optimum bit-mapping of tch codes," in *Satellite and Space Communications, 2009. IWSSC 2009. International Workshop on*, 2009, pp. 92 –96.

[11] N. S. Souto, J. C. Silva, and F. Cercas, "Special bit-mapping of tch codes for application in turbo schemes," in *Proc. Conf. on Telecommunications - ConfTele*, vol. 1, June 2003, pp. 445–448.

[12] A. P. Almeida, F. B. Cercas, and R. Dinis, "Flexible synchronization and decoding for tch sequences with variable length," in *IEEE/IEE ISCTA'07, Ambleside - UK*, July 2007.

[13] J. Silva, N. Souto, R. Dinis, and P. Montezuma, "On the use of tch sequences for synchronization, channel and noise estimation," in *Signal Processing and Communication Systems, 2009. ICSPCS 2009. 3rd International Conference on*, sept. 2009, pp. 1 –5.

[14] N. S. Souto, J. C. Silva, F. Cercas, A. Almeida, and A. Correia, "Synchronization with tch codes," in *Proc Wireless Personal Multimedia Communications Symp. - WPMC*, Aalborg, Denmark, September 2005, pp. 998–1002.

[15] J. Silva, N. Souto, and F. Cercas, "Usage of turbo tch codes for spread spectrum applications," in *Spread Spectrum Techniques and Applications, 2004 IEEE Eighth International Symposium on*, aug.-2 sept. 2004, pp. 648 – 652.

[16] J. C. Silva, N. S. Souto, F. Cercas, A. Correia, and A. J. Rodrigues, "Turbo tch codes for application in cs-cdma schemes," in *Proc SEACORN Workshop on Simulation of Enhanced UMTS Access and Core Networks*, Cambridge, United Kingdom, March 2004.

[17] J. Silva, N. Souto, and F. Cercas, "Parity concatenated turbo codes: study of their structure and performance bounds," in *Spread Spectrum Techniques and Applications, 2004 IEEE Eighth International Symposium on*, aug.-2 sept. 2004, pp. 300 – 304.

[18] N. S. Souto, J. C. Silva, and F. Cercas, "Block turbo schemes using tch codes," in *Proc. Conf. on Telecommunications - ConfTele*, vol. 1, Aveiro, Portugal, June 2003, pp. 429–432.

[19] P. Sebastiao, F. Cercas, and A. Cartaxo, "Performance of tch codes in a land mobile satellite channel," in *Personal, Indoor and Mobile Radio Communications, 2002. The 13th IEEE International Symposium on*, vol. 4, sept. 2002, pp. 1675 – 1679 vol.4.

[20] A. Almeida, P. Silva, and M. Torres, "Generation of sidel'nikov sequences and their spectra via permutations," *Electronics Letters*, vol. 47, no. 19, pp. 1079 –1081, 9 2011.

[21] A. P. Almeida, "Generalised tch codes via zech logs," in *International Symposium on Signals, Systems and Electronics (ISSSE 2012)*, October 2012, pp. 367–372.

[22] ——, "On the structure of sidel'nikov sequences for pythagorean primes," in *International Symposium on Signals, Systems and Electronics (ISSSE 2012)*, October 2012, pp. 296–301.

[23] A. P. Almeida, F. B. Cercas, and R. Dinis, "Frame synchronization using frequency domain processing for spread-spectrum systems," in *The Sixth IASTED International Conference on Communications, Internet, and Information Technology*, ser. CIIT '07. Anaheim, CA, USA: ACTA Press, 2007, pp. 102–105. [Online]. Available: http://dl.acm.org/citation.cfm?id=1672083.1672104

[24] A. P. Almeida, R. Dinis, and F. B. Cercas, "An fft-based acquisition scheme for ds-cdma systems," *IEEE ISCIT'07 - Sydney Australia*, pp. 905–910, Oct. 2007.

[25] L. Childs, *A Concrete Introduction to Higher Algebra*. Springer-Verlag, NY, 1979.

[26] J. B. Fraleigh, *A First Course in Abstract Algebra*. Addison-Wesley, Reading, MA, 1989.

[27] J. A. Gallian, *Contemporary Abstract Algebra*. Heath, Lexington, MA, 1990.

[28] R. Lidl and G. Pilz, *Applied Abstract Algebra*. Springer-Verlag, NY, 1984.

[29] G. Mackiw, *Applications of Abstract Algebra*. Wiley, NY, 1985.

[30] W. Burnside, *Theory of Groups of Finite Order*. Cambridge University Press, 1953.

[31] N. Koblitz, *A Course in Number Theory and Cryptography*. Springer-Verlag, NY, 1987.

[32] M. Hall, *Theory of Groups*. Chelsea, NY, 1975.

[33] A. E. Kurosh, *The Theory of Groups*. Chelsea, NY, 1979.

[34] I. D. MacDonald, *The Theory of Groups*. Krieger, London, 1988.

[35] J. Rose, *A Course on Group Theory*. Cambridge University Press, 1978.

[36] J. Rotman, *An Introduction to the Theory of Groups*. Allyn and Bacon, Boston, 1984.

[37] D. Shanks, *Solved and Unsolved Problems in Number Theory*, 4th ed. New York: Chelsea, 1993.

[38] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.

[39] F. MacWilliams and N. Sloane, *The theory of Error-Correcting Codes*. North-Holland, 2006.

[40] *Codes and Designs*, vol. 52, 1979.

[41] R. A. Hill, *A First Course in Coding Theory*. Oxford Univ. Press, 1986.

[42] J. H. van Lint, *Introduction to Coding Theory*. Springer-Verlag, NY, 1982.

[43] S. Roman, *Coding and Information Theory*, ser. Graduate Texts in Mathematics. Springer, 1992, vol. 134.

[44] C. V. Eynden, *Elementary Number Theory*. Random House, NY, 1987.

[45] G. H. Hardy and E. M. Wright, *An Introduction to the Theory of Numbers*. Oxford Univ. Press, 1979.

[46] I. Niven and H. S. Zuckerman, *An Introduction to the Theory of Numbers*. Wiley, NY, 1991.

[47] A. Grytczuk, M. Wójtowicz, and F. Luca, "Another note on the greatest prime factors of fermat numbers." *Southeast Asian Bulletin of Mathematics*, vol. 25, no. 1, p. 111, 2001.

[48] C. G. J. Jacobi, "Uber die kreistheilung und ihre anwendung auf die uber die kreistheilung und ihre anwendung auf die zahlentheorie," in *in C. G. J. Jacobi's Gesurnmelte Werke*. Berlin: Verlag von Georg Reimer, Oct. 1837, vol. 6, pp. S.254–274.

[49] K. Imamura, "A method for computing addition tables in $GF(p^n)$," *Information Theory, IEEE Transactions on*, vol. 26, no. 3, pp. 367 – 369, may 1980.

[50] K. Huber, "Some comments on Zech's logarithms," *Information Theory, IEEE Transactions on*, vol. 36, no. 4, pp. 946–950, Jul 1990.

[51] F. Assis and C. Pedreira, "An architecture for computing Zech's logarithms in $GF(2^m)$," *Computers, IEEE Transactions on*, vol. 49, no. 5, pp. 519 –524, may 2000.

[52] K. Huber, "Solving equations in finite fields and some results concerning the structure of $GF(p^m)$," *Information Theory, IEEE Transactions on*, vol. 38, no. 3, pp. 1154–1162, May 1992.

[53] D. M. Burton, *Elementary Number Theory*, 4th ed. Mcgraw-Hill College, 1997.

[54] R. Lidl and H. Niederreiter, *Finite Fields*. Cambridge University Press, 1984.

[55] H. Riesel, *Prime Numbers and Computer Methods for Factorization*, 2nd ed. Birkhäuser, 1994.

[56] T. G. Group, "Gap – groups, algorithms, and programming, version 4.4.12."

[57] V. M. Sidel'nikov, "Some k-valued pseudo-random sequences and nearly equidistant codes," *Prob. Inf. Transm.*, vol. 5, no. 1, pp. 12–16, 1969.

[58] N. Y. Yu and G. Gong, "New construction of -ary sequence families with low correlation from the structure of sidelnikov sequences," *Information Theory, IEEE Transactions on*, vol. 56, no. 8, pp. 4061 –4070, 2010.

[59] A. Lempel, M. Cohn, and W. Eastman, "A class of balanced binary sequences with optimal autocorrelation properties," *Information Theory, IEEE Transactions on*, vol. 23, no. 1, pp. 38 – 42, Jan. 1977.

[60] D. Sarwate, "Comments on 'a class of balanced binary sequences with optimal autocorrelation properties' by lempel, a., et al." *Information Theory, IEEE Transactions on*, vol. 24, no. 1, pp. 128 – 129, Jan. 1978.

[61] W. C. Huffman and V. Pless, *Fundamentals of Error-Correcting Codes*. Cambridge University Press, 2003.

[62] Y.-J. Kim and H.-Y. Song, "Cross correlation of sidel'nikov sequences and their constant multiples," *Information Theory, IEEE Transactions on*, vol. 53, no. 3, pp. 1220 –1224, march 2007.

[63] J. Proakis, *Digital communications*, 4th ed. Mcgraw-Hill, 2001.

[64] W. C. Jakes, *Microwave mobile communications*. IEEE Press., 1974.

[65] J. D. Parsons, *The mobile radio propagation channel*, 1st ed. John Wiley and Sons, 1992.

[66] T. S. Rappaport, *Wireless communications principles and practice*, 2nd ed. Prentice-Hall, 2002.

[67] R. Steele, *Mobile radio communications*. John Wiley and Sons, 1996.

[68] G. L. Stüber, *Principles of mobile communication*, 2nd ed. Kluwer Academic Publishers, 2001.

[69] M. Yacoub, *Foundations of mobile radio engineering*. CRC Press Inc., 1993.

[70] M. Abramovitz and I. A. Stegun, *Handbook of Mathematical Handbook of Mathematical Functions with formulas, graphs, and mathematical tables*. National Bureau of Standards, Washington, 1972.

[71] P. J. A. Sebastião, F. B. Cercas, and A. V. T. Cartaxo, "Efficient simulation of fec systems for radio channels," in *IEE Proc.-Commun*, 2005.

[72] A. Polydoros and C. Weber, "A unified approach to serial search spread-spectrum code acquisition–part i: General theory," *Communications, IEEE Transactions on*, vol. 32, no. 5, pp. 542 – 549, may 1984.

[73] A. Polydoros and C. L. Weber, "Optimal detection considerations for low probability of intercept," in *Military Communications Conference - Progress in Spread Spectrum Communications, 1982. MILCOM 1982. IEEE*, vol. 1, oct. 1982, pp. 2.1–1 –2.1–5.

[74] A. Polydoros and C. Weber, "A unified approach to serial search spread-spectrum code acquisition–part i: General theory," *Communications, IEEE Transactions on*, vol. 32, no. 5, pp. 542 – 549, may 1984.

[75] ——, "A unified approach to serial search spread-spectrum code acquisition–part ii: A matched-filter receiver," *Communications, IEEE Transactions on*, vol. 32, no. 5, pp. 550 – 560, may 1984.

[76] A. Nielsen and S. Korpela, "Wcdma initial cell search," in *Vehicular Technology Conference, 2000. IEEE VTS-Fall VTC 2000. 52nd*, vol. 1, 2000, pp. 377 –383 vol.1.

[77] M. Srinivasan and D. Sarwate, "Simple schemes for parallel acquisition of spreading sequences in ds/ss systems," *Vehicular Technology, IEEE Transactions on*, vol. 45, no. 3, pp. 593 –598, Aug. 1996.

[78] W. Hai, C. Zhiqiang, and M. Sundelin, "The impact of cell search on system performance in wcdma," in *Vehicular Technology Conference Proceedings, 2000. VTC 2000-Spring Tokyo. 2000 IEEE 51st*, vol. 2, 2000, pp. 1425 –1429.

[79] W. Zhuang, "Noncoherent hybrid parallel pn code acquisition for cdma mobile communications," *Vehicular Technology, IEEE Transactions on*, vol. 45, no. 4, pp. 643 –656, nov 1996.

[80] W. Zhang, H. Poor, and Z. Quan, "Frequency-domain correlation: An asymptotically optimum approximation of quadratic likelihood ratio detectors," *Signal Processing, IEEE Transactions on*, vol. 58, no. 3, pp. 969 –979, march 2010.

[81] J. Cimini, L., "Analysis and simulation of a digital mobile channel using orthogonal frequency division multiplexing," *Communications, IEEE Transactions on*, vol. 33, no. 7, pp. 665 – 675, jul 1985.

[82] C. Ciochina and H. Sari, "A review of ofdma and single-carrier fdma," in *Wireless Conference (EW), 2010 European*, april 2010, pp. 706 –710.

[83] P. Hoeher, S. Kaiser, and P. Robertson, "Two-dimensional pilot-symbol-aided channel estimation by wiener filtering," in *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, vol. 3, apr 1997, pp. 1845 –1848 vol.3.

[84] L. Deneire, P. Vandenameele, L. van der Perre, B. Gyselinckx, and M. Engels, "A low complexity ml channel estimator for ofdm," in *Communications, 2001. ICC 2001. IEEE International Conference on*, vol. 5, 2001, pp. 1461 –1465 vol.5.

[85] A. Orozco-Lugo, M. Lara, and D. McLernon, "Channel estimation using implicit training," *Signal Processing, IEEE Transactions on*, vol. 52, no. 1, pp. 240 – 254, jan. 2004.

[86] D. Chu, "Polyphase codes with good periodic correlation properties (corresp.)," *Information Theory, IEEE Transactions on*, vol. 18, no. 4, pp. 531 – 532, jul 1972.

[87] R. Dinis and P. Silva, "Analytical evaluation of nonlinear effects in mc-cdma signals," *Wireless Communications, IEEE Transactions on*, vol. 5, no. 8, pp. 2277 –2284, aug. 2006.

[88] R. Dinis, N. Souto, J. Silva, A. Kumar, and A. Correia, "Joint detection and channel estimation for ofdm signals with implicit pilots," in *Mobile and Wireless Communications Summit, 2007. 16th IST*, july 2007, pp. 1 –5.

[89] R. Dinis, C.-T. Lam, and D. Falconer, "Joint frequency-domain equalization and channel estimation using superimposed pilots," in *Wireless Communications and Networking Conference, 2008. WCNC 2008. IEEE*, 31 2008-april 3 2008, pp. 447 –452.

[90] R. A. Scholtz and M. Z. Win, "Impulse radio, invited paper," in *IEEE PIMRC*, Helsinki, Finland, 1997.

[91] R. A. Scholtz, "Multiple access with time-hopping impulse modulation," in *IEEE Military Communications Conference*, Oct. 1998, pp. 447–450.

[92] G. Biradar, S. Merchant, and U. Desai, "An adaptive frequency and time hopping ppm uwb for multiple access communication," in *Information, Communications Signal Processing, 2007 6th International Conference on*, 2007, pp. 1 –5.

[93] P. Withington and et al., "Preliminary results from ultra-wideband (impulse) scanning receivers," in *Proc. IEEE MILCOM*, NJ, USA, 1999, pp. 1186–1190.

[94] J. Foerster and et al., "Channel modeling sub-committee report final," IEEE P802.15 Wireless Personal Area Networks, P802.15-02/49r1-SG3a, Tech. Rep., February 2003.

[95] A. Saleh and R. A. Valenzuela, "A statistical model for indoor multipath propagation," *IEEE Journal of Selected Areas in Communications*, vol. 5, pp. 128–137, February 1987.

[96] W. Malik, C. Stevens, and D. Edwards, "Multipath effects in ultrawideband rake reception," *Antennas and Propagation, IEEE Transactions on*, vol. 56, no. 2, pp. 507 –514, 2008.

[97] G. Durisi and S. Benedetto, "Performance evaluation and comparison of different modulation schemes for uwb multiaccess systems," in *IEEE International Conference on Communications (ICC '03)*, vol. 3, May 2003, pp. 2187–2191.

[98] N. Bo Hu; Beaulieu, "Exact bit error rate analysis of th-ppm uwb systems in the presence of multiple-access interference," *Communications Letters, IEEE*, vol. 7, no. 12, pp. 572–574, Dec. 2003.

[99] J. Park, "A simple analysis of bit error probability of uwb-th ppm systems with the multiple access interference," *IEICE Electronics Express*, vol. 1, no. 14, pp. 423–428, October 2004.

[100] J. Foerster and Q. Li, "Uwb channel modeling contribution from intel," Intel Corporation, Tech. Rep., 2002.

Index

Z