# Pearson Codes Design with Error-Control Capabilities

Haoyu Ai

# Pearson Codes Design with Error-Control Capabilities

by

**Haoyu Ai**

in partial fulfillment of the requirements for the degree of

**Master of Science**

in Electrical Engineering

at the Delft University of Technology,

to be defended publicly on Thursday November 17th, 2016 at 2:00 PM.

| | |
|---|---|
| Supervisor: | Dr. ir. J. H. Weber |
| Thesis committee: | Dr. ir. K. A. S. Immink |
| | Dr. ir. R. Heusdens |

TUDelft Delft University of Technology

# Abstract

In mass storage systems or certain transmission channels, such as optical data storage and non-volatile memory (flash), noise or interference is not the only disturbance during the data transmission, the error performance sometimes can be seriously degraded by the phenomena of unknown channel offset (drift) or gain mismatch, this is because the conventional minimum Euclidean distance decoding, where the receiver picks a codeword from the code book to minimise the Euclidean distance with the received codeword, doesn't offer resistance for the offset and gain mismatch. As a alternative to the minimum Euclidean distance detection, a new distance called Pearson distance and the Pearson-distance-based detection are introduced by Immink and Weber[1], where the error performance is immune to unknown offset and/or gain mismatch but more sensitive to the noise than the traditional Euclidean distance detection. In addition, the Pearson-distance-based decoding can only productively used for sets of q-ary codewords with some specific properties, which is a new class of code called Pearson code. Therefore, to make the Pearson-distance-based detection more applicable against the channel noise, it is crucial to improve the error control capabilities of Pearson code.

In this thesis, we will investigate the performance of different Pearson codes in the Pearson-distance-based detection for offset-only mismatch case and offset-and-gain mismatch case. We will analyze the factors that can reflect or affect the performance of the Pearson code by studying the relations between Pearson distance and Hamming distance. The simulation results will be compared and discussed, the advices for improving the error control capability of the Pearson code will be given according the factors we found.

**Keywords:** noise, offset and gain mismatch, performance, error control capacity, Pearson distance, Pearson code

# Contents

# 1

# Introduction

In recent years, with the rapid development of large-scale, high-speed data networks for commerce, government, and military use, reliable and efficient data transmission and storage technology becomes more and more important for the growing demand of digital transmission and storage systems, for example, cellular telephony and Blu-ray Disc. In telecommunications, code is a system of rules to convert the information–here we are talking about digital data–into another form or representation, and coding theory is the study of the properties of different kinds of codes and their fitness for a specific application.

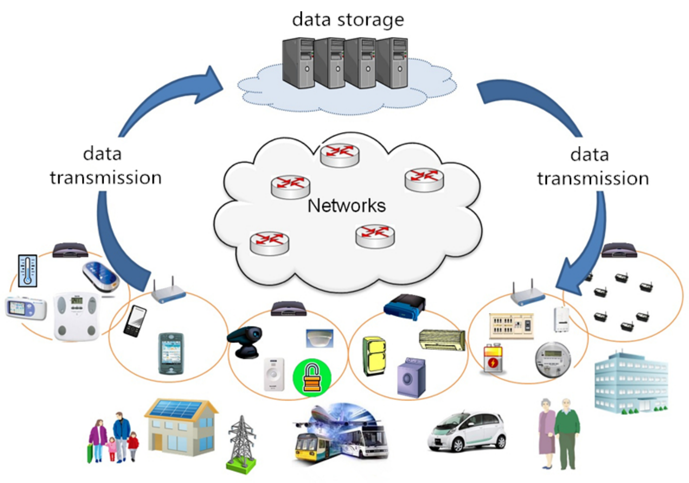

Figure 1.1: The data transmission and data storage in large-scale networks

There are four types of coding[2]: source coding, channel coding, cryptographic coding, and line coding. The source coding[3], also called data compression, attempts to compress naturally redundant source data for efficient transmission or storage, for example, Zip data compression. Cryptographic coding is a technique for secure communication which can protect the private data from third parties, it applications are ATM cards, computer passwords, etc. The line coding is a code chosen scheme for use within a telecommunications system for baseband transmission purposes, it is widely used for digital data transport. In this thesis report, we focus on the channel coding.

As we know, Shannon published an article[4] in 1948 about how best to encode the information a sender wants to transmit. He demonstrated that, by proper encoding of the information,we can reduce the errors caused by the channel to any desired level without sacrifice the information transformation rate, as long as the information rate is less than the channel capacity. Therefore, the channel coding for error control has become an essential part of modern communication technology.

## 1.1. CHANNEL CODING

In digital communication systems or storage systems, a channel, by definition, is the physical or logical link that connects a data source to a data sink, or a storage which can communicate a message over time as well as space. As we can see from the Figure 1.2 below, the source encoder transforms the source information, which could come from a person or machine, into a binary digits sequence called information sequence, then the channel encoder transforms the information sequence into encoded sequence called codeword sequence. The modulator transforms the codeword sequence into wave which is suitable for transmission in the channel, after the transmission, the de-modulator transforms the received wave into the same form of codeword sequence called received sequence. Then the channel decoder transforms the received sequence into estimated information sequence, and the source decoder transforms the estimated information sequence into a estimate of original source information. In the ideal condition, the estimate is a reproduction of the original source information. However, the channels are always not noise free, and noise can bring randomness or unpredictability errors into the transmission information sequences, and sometimes lead to a wrong estimate of original source information for the destination.

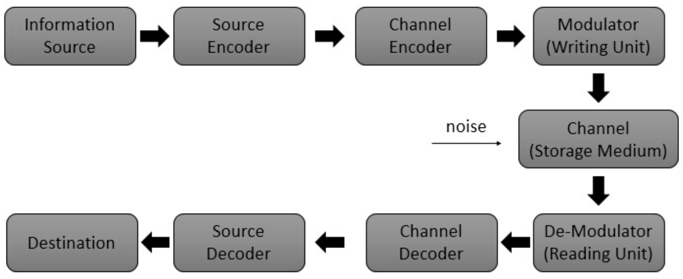

Figure 1.2: Block diagram of a typical data transmission or storage system

Therefore, a major concern for these systems is the control of data errors during transmission or storage in the channel so that the data can be reliably reproduced. There are various coding methods to deal with the transmission errors, the very simple way is asking for data retransmission if the receiver found there are errors occur during the transmission, which is called automatic repeat request(ARQ) and could be costly or impossible sometimes. While channel coding, is a forward error correction (FEC) technique, which combines communications and computer technology, it encodes the information data in the beginning, and enable the receiver to detect and correct a limited number of errors to recover the original data after a transmission from the source to the sink or written and read in a storage device without retransmission. Different from the source coding, channel coding systematically add redundancy to the data to ensure maximum possible rate transmissions with as few errors as possible.

## 1.2. ERROR CORRECTING CODE

The error-correcting code is such a special algorithm for the encoding process of channel coding. The most famous error-correcting code was was invented by a mathematician named Richard Hamming in 1950, known as the Hamming code, and it belongs to the family of block code. When the sender wants to transmit a very long data stream, he will break the long data sequence into pieces with fixed lengths, which is called message, and the block codes encode each message into a individual block called codeword, after the transmission, the receiver can decode the blocks back to original messages. Hamming code one of the block codes that entails the inclusion of additional redundant digits which is called parity bits in the original message, and calculating the parity of the received block sequence on the receiver end can reveal whether there are any errors, where the errors are located, and how to recover the original message. Here a concept named distance (usually is Hamming distance) is introduced, which is a important factor of the quality of block code, it measures the minimum number of substitutions required to change one codeword into the other codewords, or the minimum number of errors that could have transformed one codeword into the other codewords. Therefore, it determines error-correct ability of a block code, and we will introduce different kinds of distance in the next chapter.

The idea behind of channel coding is so crucial because of the inevitable occurrence of different kinds of errors during the transmission on any given type of communication channel, and the errors mostly come from noise or interference in the channel, for example, the additive white Gaussian noise(AWGN). But in some practical cases, noise is not the only issue in the channel, two common physical factors called offset and amplitude(gain) mismatch may also hamper the reliability of communication or storage systems. Here the offset refers to the shift of all information bits value during data transmission or storage, and the gain refers to an amplification factor of the amplitude of all information bits value during data transmission or storage. Note that unlike the random noise, both gain and offset do not vary from bit to bit, but are the same for the whole block of the transmission bit string.

## 1.3. RELATED RESEARCHES

There are many examples about the channels with offset and gain mismatch. In wireless communication channels, fading, which may either be due to multipath propagation(multipath induced fading), or due to shadowing from obstacles affecting the wave propagation(shadow fading), is deviation of the attenuation(unknown gain) of the signal that varying rapidly. In baseband transmission channels, the incoming signal power is evaluated against the baseline (a running average of the received signal power), baseline wandering(a offset in the baseline) can be caused by the attenuation of the low frequencies of the channel. In Non-volatile memories, like the 'EPROM'(Electrically Erasable Programmable Read-Only Memory), data is stored as charge by use of floating gate transistors, which can leak away as the time elapsed between writing and reading the data, therefore, in aging devices, this may leads to offset of threshold voltage of the memory cells, and the offset of the different groups of cells could be very different. In optical disc media, such as Compact Disc, DVD, and blue-ray disc, the retrieved signal depends on the dimensions of the written features and upon the quality of the light path,fingerprints and scratches may result in rapid offset and gain variation[5].

The paper[1] shows that unknown channel gain and offset mismatch may lead to a massive performance degradation when using traditional detector based on thresholds or the Euclidean distance. In the prior art, various methods have been proposed to solve this issue. The first method uses data reference[6], which is two reference symbol values, in each codeword, where the first symbol is set equal to the lowest signal level and the second symbol equal to the highest signal level. The positions and amplitudes of two reference symbols are known to the receiver, the idea is using these reference data to teach the data detection circuitry the momentary values of the channel's characteristics such as impulse response, gain, and offset. So the receiver can directly measure the amplitude of the retrieved reference symbols and normalize the amplitudes of the remaining symbols of the retrieved codeword before applying detection.The second method is called rank modulation scheme, which is used for the multi-level flash memory cells, in this scheme, information is stored in an ordered set of muti-level cells in the permutation induced by the charge levels of the cells. So it allow the decrease of all the charge levels in a block of cells by a constant(like the data reference in the first method) amount smaller than the lowest charge level, which would maintain their relative values, and leave the information unchanged[7], Thus it can overcome the difficulties in flash memories having aging offset levels. Another prior art method in [8] shows that an code book where all codewords have equal balance and energy (BE constraints) has an intrinsic resistance against unknown gain and/or offset, which is called constant composition code and specialize to constant weight codes in the binary case, however, the redundancy price of these codes is unattractively high for small codeword length.

## 1.4. MOTIVATION

In a recent contribution[1], Immink & Weber advocated Pearson-distance-based detection, which uses Pearson distance instead of Hamming distance(or Euclidean distance) and is immune to unknown offset and gain mismatch, the Pearson distance comes from the Pearson product-moment correlation coefficient[9], which can reveal the linear dependence between two codewords. However, not every block code can be used for the Pearson-distance-based detection. So in[5], they came up with new codes called Pearson codes which can be perfectly applied in the Pearson-distance-based detection, whose redundancy is much less than that of balanced codes, and we will introduce it in the next chapter.

The Pearson codes have huge advantages over the other codes when there are offset and gain mismatches in the channel, however, they are still very sensitive to errors. The single Parity Check Pearson codes introduced by Immink and Weber[10] have better error performance and it is very potential to be improved. Thus it is very promising to design and optimize the Pearson code with more error correcting capacity and make them more applicable. In this report, we will analyze and simulate the performance of different Pearson codes and solve these questions: which properties of a Pearson code can have effect on its Performance in the channel with offset and/or gain mismatch? code length? code rate? or Hamming distance? is there any relations between Hamming distance and modified Pearson distance/Pearson distance? if so, can we improve a Pearson code according one of these factors?

## 1.5. THESIS OUTLINE

The outline of this thesis report is as follows:

Chapter **1** introduces the background of error control coding techniques, and the practical issues caused by channel offset and gain mismatch in different fields of the industry, then shows different methods proposed in related literature and the advantages of Pearson distance detection compared with other methods.

Chapter **2** introduces associated basic concepts of error control code such as Hamming distance, Euclidean distance, Pearson distance, Modified Pearson distance, Word error rate etc. and shows the minimum distance decoding computation for the traditional Euclidean distance detection.

Chapter **3** focuses on the offset-only minimum Modified Pearson distance detection and compares the performance of different Pearson codes in several aspects like: code book size, codeword length, code rate, Hamming distance, different $q(q = 2$ and $q = 3)$ to find out which factors could have impact on the code performance.

Chapter **4** investigates the relations between Hamming distance and Modified Pearson distance and come up with advices to design and improve a Pearson code for the offset-only case.

Chapter **5** focuses on the offset and gain mismatch Pearson distance detection, illustrates the advantages of Pearson detector over modified Pearson detector and Euclidean detector. Then compares the performance of different Pearson codes in several aspects like: codeword length, code rate, Hamming distance, and different $q(q = 2$ and $q = 3)$ to find out which factors could have impact on the code performance.

Chapter **6** investigates the relations between Hamming distance and Pearson distance and come up with advices to design and improve a Pearson code for the offset and gain mismatch case.

Chapter **7** summarizes our conclusions and possible future work.

# 2

# Basic Concepts and Methods

In this chapter, we will review some basic concepts in error-correct coding techniques and some block codes, then introduce new concepts and computations related to the Pearson distance and Pearson codes, after that, we will compute the word error rate upperbound of the traditional minimum Euclidean distance detection for several block codes and simulate the detection process in Matlab, the results can be compared with that of Pearson-distance-based detection in the next chapters.

## 2.1. Basic concepts

In this section, first we will introduce the concept of block code, the channel model we use, and 4 distances: Hamming distance, Euclidean distance, Pearson distance, Modified Pearson distance. Then we will introduce the Q-function, which can be used for the calculation of the word(block) error rate(probability) in the minimum distance decoding.

### 2.1.1. Block code

In coding theory, a block code is a family of error-correcting codes that encode data in blocks over an alphabet $Q = \{0, 1, \ldots, q-1\}$, $q \geq 2$ of size $q$, and every block has the same fixed length. All the encoded blocks, also called codewords, are the elements of a block code, and all the different codewords are gathered into a code book $S$ of the block code. The length of every codeword is called code length, denoted by $n$, and the number of codewords is called the cardinality or size of the code, denoted by $|S|$. If $q = 2$, then the block code is called binary block code; if $q = 3$, then it is called ternary block code. The code rate of a block code is the proportion of the data-stream that is useful (non-redundant)[11]. It is defined as:

$$R = \frac{log_q |S|}{n} \tag{2.1}$$

Because $|S|$ is at most $q^n$, the code rate is a real number between 0 and 1, and the closer it is to 1, the more efficiently the data is encoded by the block code.

### 2.1.2. OUR CHANNEL MODELS

We consider a communication code book, $S$, of chosen q-ary codewords $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ over the $q$-ary alphabet $Q = \{0, 1, \ldots, q-1\}$, $q \geq 2$, where $n$, the length of $\boldsymbol{x}$, is a positive integer. Every codeword have the same length $n$. We suppose the sent codeword $\boldsymbol{x}$ is received as the vector $\boldsymbol{r} = a(\boldsymbol{x} + \boldsymbol{v}) + b$, $r_i \in \mathbb{R}$, where we use the shorthand notation $a\boldsymbol{x} + b = (ax_1 + b, ax_2 + b, \ldots, ax_n + b)$. The basic assumption is that the sent codeword $\boldsymbol{x}$ is scaled by both unknown gain $a$ and offset $b$, where $a$ and $b \in \mathbb{R}$, $a > 0$, and corrupted by additive noise $\boldsymbol{v} = (v_1, v_2, \ldots, v_n)$, $v_i \in \mathbb{R}$ are noise samples with distribution $\mathcal{N}(0, \sigma^2)$. Both quantities of gain and offset: $a$ and $b$, vary slowly so that we can assume they are the same for all $n$ symbols of the codeword $\boldsymbol{x}$. If $a = 1$ and $b = 0$, the channel is matched and only contains the noise; if $a = 1$ and $b \neq 0$, there are only offset mismatch and noise in the channel; if $a \neq 1$ and $b \neq 0$, there are both gain and offset mismatch and noise in the channel.

### 2.1.3. HAMMING DISTANCE

The Hamming distance between two signal vectors with equal length is the number of positions at which the corresponding symbols are different. It represent the minimum number of symbol errors could occur to transform one code word into another. In binary case, the Hamming weight of a binary vector $\boldsymbol{x}$, $w(\boldsymbol{x})$, is defined as the number of nonzero components of $\boldsymbol{x}$, and the hamming distance between two binary signal vector $\boldsymbol{x}$ and $\boldsymbol{y}$ is equal to the Hamming weight of the modulo-2 addition of $\boldsymbol{x}$ and $\boldsymbol{y}$:

$$d_H(\boldsymbol{x}, \boldsymbol{y}) = w(\boldsymbol{x} + \boldsymbol{y})$$

and it is also equal to:

$$d_H(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n} |x_i - y_i| = \sum_{i=1}^{n} (x_i - y_i)^2 \tag{2.2}$$

### 2.1.4. EUCLIDEAN DISTANCE

In the traditional minimum Euclidean distance detection, where the Euclidean distance between two signal vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, $x_i$ and $y_i \in \mathbb{R}$, is defined as(squared):

$$\delta_E(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n} (x_i - y_i)^2 \tag{2.3}$$

When there is channel's momentary gain $a$ and offset $b$ mismatch for the signal vector $\boldsymbol{x}$, we can get:

$$\begin{aligned} \delta_E(a\boldsymbol{x} + b, \boldsymbol{y}) &= \sum_{i=1}^{n} (ax_i + b - y_i)^2 \\ &= -2a \sum_{i=1}^{n} x_i y_i + \sum_{i=1}^{n} (ax_i + b)^2 - 2b \sum_{i=1}^{n} y_i + \sum_{i=1}^{n} y_i^2 \end{aligned} \tag{2.4}$$

Compare the results of two equations above, we can see that the offset $b$ and gain $a$ mismatch have massive influence in Euclidean distance measure, which means they can cause performance degradation in the minimum Euclidean distance detection.

### 2.1.5. PEARSON DISTANCE AND MODIFIED PEARSON DISTANCE

In this section, we will introduce the two new distances: the Pearson distance and the modified Pearson distance, which are proposed in [10]. The Pearson distance is used for the general case, which is intrinsically resistant to both offset and gain mismatch, the regular Pearson distance between the signal vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, $x_i$ and $y_i \in \mathbb{R}$, is defined by

$$\delta_P(\boldsymbol{x},\boldsymbol{y}) = 1 - \rho_{\boldsymbol{x},\boldsymbol{y}} \tag{2.5}$$

where the (*Pearson*) *correlation coefficient* $\rho_{\boldsymbol{x},\boldsymbol{y}}$ is:

$$\rho_{\boldsymbol{x},\boldsymbol{y}} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sigma_x \sigma_y} \tag{2.6}$$

and

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\sigma_x^2 = \sum_{i=1}^{n}(x_i - \overline{x})^2 \tag{2.7}$$

because the Pearson coefficient has the property:

$$\rho_{\boldsymbol{x},\boldsymbol{y}} = \rho_{C_1 + C_2\boldsymbol{x},\boldsymbol{y}}$$

where $C_1$ and $C_2$ are constants and $C_2 > 0$, so we say the Pearson distance, $1 - \rho_{\boldsymbol{x},\boldsymbol{y}}$, is invariant to both offset and gain mismatch. Because $-1 \le \rho_{\boldsymbol{x},\boldsymbol{y}} \le 1$, the Pearson distance $\delta_P(\boldsymbol{x},\boldsymbol{y})$ lies in the interval $[0,2]$.

While the modified Pearson distance, which can reduce the redundancy, and improve the resistance against additive noise, usually used in the offset only mismatch case, is defined by

$$\delta_{MP}(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{n}(x_i - y_i + \bar{y})^2 \tag{2.8}$$

We find

$$\begin{aligned}\delta_{MP}(\boldsymbol{x}+b,\boldsymbol{y}) &= \sum_{i=1}^{n}(x_i + b - y_i + \bar{y})^2 \\ &= \sum_{i=1}^{n}(x_i - y_i + \bar{y})^2 + 2b\sum_{i=1}^{n}(x_i - y_i + \bar{y}) + b^2,\end{aligned}$$

where b is the offset mismatch, we know

$$\sum_{i=1}^{n}(x_i - y_i + \bar{y}) = \sum_{i=1}^{n} x_i$$

So that

$$\delta_{MP}(\boldsymbol{x}+b,\boldsymbol{y}) = \sum_{i=1}^{n}(x_i - y_i + \bar{y})^2 + 2b\sum_{i=1}^{n} x_i + b^2 \tag{2.9}$$

We can see if $\boldsymbol{x}$ is the sent codeword, $\boldsymbol{x}+b$ denote received signal vector, and $\boldsymbol{y}$ denote the codewords in the code book. We can see the last two parts in (2.9) is independent from the codewords $\boldsymbol{y}$, Thus

$$\delta_{MP}(\boldsymbol{x}+b,\boldsymbol{y}) \equiv \sum_{i=1}^{n}(x_i - y_i + \bar{y})^2 = \delta_{MP}(\boldsymbol{x},\boldsymbol{y})$$

So it is shown that the modified Pearson distance is intrinsically resistant to the channel offset b.

### 2.1.6. THE Q-FUNCTION

In statistics, the Q-function is a convenient way to express right-tail probabilities for normal (Gaussian) random variables. For $x \in \mathbb{R}$, $Q(x)$ is defined as the probability that a standard normal random variable (zero mean: $\mu = 0$, unit variance: $\sigma^2 = 1$) exceeds $x$[12]:

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} e^{-\frac{t^2}{2}} dt \tag{2.10}$$

while sometimes the Q-function is also useful for expressing right-tail probabilities of nonstandard normal variates. If:

$$X \backsim \mathcal{N}(\mu, \sigma^2)$$

then

$$\frac{X-\mu}{\sigma} \backsim \mathcal{N}(0,1)$$

To express $Pr[X > \gamma]$ in terms of $Q$, where $\gamma \in \mathbb{R}$, define $\eta = \frac{\gamma-\mu}{\sigma}$. Then

$$\begin{aligned} Pr[X>\gamma] &= Pr[X > \eta\sigma + \mu] \\ &= Pr[\frac{X-\mu}{\sigma} > \eta] \\ &= Q(\eta) \\ &= Q(\frac{\gamma-\mu}{\sigma}) \end{aligned} \tag{2.11}$$

and because $Q(x) = 1 - Q(-x)$, so

$$\begin{aligned} Pr[X \leq \gamma] &= 1 - Pr[X > \gamma] \\ &= 1 - Q(\eta) = Q(-\eta) \\ &= Q(\frac{\mu-\gamma}{\sigma}) \end{aligned} \tag{2.12}$$

Therefore, in the case of $\gamma = 0$, we get:

$$Pr[X \leq 0] = Q(\frac{\mu}{\sigma}) \tag{2.13}$$

## 2.2. SOME TRADITIONAL BLOCK CODES

In this section, we will introduce four traditional block codes: Hamming code, Extended Hamming code(SECDED), BCH code, Reed-Muller code, which will be involved in the next chapters. In this thesis, we only consider the binary cases of these traditional block codes.

### 2.2.1. HAMMING CODE

Hamming codes are a family of linear error-correcting codes, it can detect up to two-bit errors or correct one-bit error without detection of uncorrected errors. Generally, for each integer $t \geq 2$ there is a binary Hamming code with block length $n = 2^t - 1$ and message length $k = 2^t - t - 1$. All the binary Hamming codes are perfect codes, which means they can achieve the highest possible code rate with minimum distance of three compared with other codes with the same codeword length[13]. The rate of binary Hamming codes is

$$R = \frac{k}{n} = 1 - \frac{t}{2^t - 1}$$

### 2.2.2. EXTENDED HAMMING CODE(SECDED)

The extended Hamming code is is popular in computer memory systems, where it is also named as SECDED (single-error correcting, double-error detecting) [14], we know the Hamming codes sometimes cannot detect two-bit error, because it cannot distinguish a two-bit error of some codeword from one-bit error of a different codeword, while adding an extra parity bit to the codewords can overcome this problem, so that its minimum distance increase to 4, which means, the Extended Hamming code detect and correct one-bit error and meanwhile detect (but not correct) two-bit error. Therefore, for each integer $t \geq 2$ there is a binary Extended Hamming code with block length $n = 2^t$ and message length $k = 2^t - t - 1$. The code rate is

$$R = \frac{k}{n} = 1 - \frac{t+1}{2^t}$$

### 2.2.3. BCH CODE

The Bose, Chaudhuri, and Hocquenghem (BCH) codes form a large class of powerful random error-correcting cyclic codes. It is a remarkable generalization of the Hamming code for multiple-error correction. For each integer $t \geq 3$ and $h < 2^{t-1}$, there exists a binary BCH code with block length $n = 2^t - 1$, and its minimum Hamming distance is:

$$d_{min,H} \geq 2h + 1$$

### 2.2.4. REED-MULLER CODE

The Reed-Muller code is named after Irving S. Reed and David E. Muller. It is listed as RM$(u, v)$, where $u$ is the order of the code, $0 \leq u \leq v$, and $v$ determines the code length: $n = 2^v$. The minimum Hamming distance of binary RM$(u, v)$ is

$$\boldsymbol{min}(d_H) = 2^{v-u}$$

We notice that the extended Hamming code $(8, 4, 4)$ is also a Reed-Muller code RM$(1, 3)$.

## 2.3. THE PEARSON CODE

In this section we will introduce a new class of block codes: Pearson code, which will be mainly investigated in this thesis. Then two related examples will be introduced: the T-constrained code, and Single Parity check T-constrained code.

The Pearson code introduced by Immink and Weber is a new class of code and is defined as a set of codewords that can be uniquely decoded by a minimum Pearson distance detector. Let $S$ be a code book of chosen $q$-ary codewords $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ over the $q$-ary alphabet $Q = \{0, 1, \ldots, q-1\}$, $q \geq 2$. Thus the codewords in a Pearson code must have these two properties[5]:

- ***Property A:***
  If $\boldsymbol{x} \in S$, then $c_1 + c_2\boldsymbol{x} \notin S$ for all $c_1, c_2 \in \mathbb{R}$ with $(c_1, c_2) \neq (0, 1)$ and $c_2 > 0$.
- ***Property B:***
  $\boldsymbol{x} = (c, c, \ldots, c) \notin S$ for all $c \in \mathbb{R}$

we found that in the binary case, all block codes with all one and all zero codewords excluded are Pearson codes and can be applied in Pearson-distance-based detection. Here we define the modified block code as the normal block code with all-one codeword and all-zero codeword removed from the code book. To make the results more comparable for different detectors of different detections, the object codes for all computations and simulations in this thesis report are modified block codes, the reason why we also have to exclude the all '0' codeword in minimum modified Pearson distance detection will be discussed in section 3.1 of chapter 3. Moreover, in this thesis report, we mostly focus the design of the binary($q = 2$) block code, except the section 3.3 in chapter 3 and section 5.4 in chapter 5 , which are the study of ternary($q = 3$) code in Pearson-distance-based detection.

### 2.3.1. T-CONSTRAINED CODE

$T$-constrained codes, presented in [1], for enabling simple dynamic threshold detection of $q$-ary codewords, consist of $q$-ary $n$-length codewords, where $T$, $0 < T \leq q$, prescribed symbols must each appear at least once in a codeword, a set of $T$-constrained codewords is denoted by $S_T$. There are two classes of $T$-constrained binary codes, which are denoted by $S_1$ and $S_2$. The first code class, denoted by $S_1$, where $T = 1$, comprises all codewords wherein the symbol '0' appears at least once. The cardinality of $S_1$ equals

$$|S_1| = 2^n - 1$$

The second code class, denoted by $S_2$, where $T = 2$, is of specific interest in this report, it is usually used in conjunction with both the modified Pearson detector(modified Pearson distance) for the offset-only mismatch and Pearson detector(Pearson distance) for offset&gain mismatch. By definition, it contains all codewords wherein both the two symbols '0' and '1' appear at least once. The cardinality of $S_2$ equals

$$|S_2| = 2^n - 2$$

We can see that $S_2$ is a Pearson code, $S_1$ is not a Pearson code but it can be applied in the modified minimum Pearson distance detection.

### 2.3.2. SINGLE PARITY CHECK CODES

The single parity check T-constrained code[10], denoted by $S_{p,1}$, consists of codewords, $\boldsymbol{x}$, taken from $S_1$ that additionally satisfy the parity check:

$$\left(\sum_{i=1}^{n} x_i\right) \quad mod \quad q = a$$

where $a \in Q$ is an integer chosen by the code designer. The code size, $\left|S_{p,1}\right|$, is at least

$$\left|S_{p,1}\right| \geq \lceil \frac{|S_1|}{q} \rceil$$

In the binary case, where $q = 2$, $a \in \{0, 1\}$, when we choose $a = 1$, $S_{p,1}$ is a Pearson code, otherwise it is not a Pearson code.

## 2.4. MINIMUM EUCLIDEAN DISTANCE DETECTION

In this section we will derive the theoretical word error rate(WER) upperbound of block codes in the Minimum Euclidean distance detection. Recall our channel model in subsection 2.1.2, here we assume the channel is ideal and matched,which means $a = 1$ and $b = 0$, so a randomly chosen codeword $\boldsymbol{x}$ is sent and the received signal vector $\boldsymbol{r} = \boldsymbol{x} + \boldsymbol{v}$, where $\boldsymbol{v}$ are white Gaussian noise with distribution $\mathcal{N}(0, \sigma^2)$. We know the error occurs if there is at least one codeword $\hat{\boldsymbol{x}} \neq \boldsymbol{x}, \hat{\boldsymbol{x}} \in S$, such that the distance $\delta$ satisfy

$$\delta(\boldsymbol{r}, \hat{\boldsymbol{x}}) < \delta(\boldsymbol{r}, \boldsymbol{x}) \tag{2.14}$$

In the minimum Euclidean distance detection (2.14) becomes:

$$\delta_E(\boldsymbol{r}, \hat{\boldsymbol{x}}) - \delta_E(\boldsymbol{r}, \boldsymbol{x}) < 0$$

Using (2.3) we obtain:

$$\sum_{i=1}^{n} (r_i - \hat{x}_i)^2 - \sum_{i=1}^{n} (r_i - x_i)^2 < 0 \tag{2.15}$$

Because $\boldsymbol{r} = \boldsymbol{x} + \boldsymbol{v}$, where the additive noise $\boldsymbol{v}$ are noise samples with distribution $\mathcal{N}(0, \sigma^2)$.So the left side of (2.15) becomes

$$\begin{aligned}
&\sum_{i=1}^{n} (\hat{x}_i{}^2 - x_i^2 - 2r_i\hat{x}_i + 2r_i x_i) \\
&= \sum_{i=1}^{n} (2v_i x_i - 2v_i\hat{x}_i + \hat{x}_i{}^2 - 2x_i\hat{x}_i + x_i^2) \\
&= 2\sum_{i=1}^{n} (x_i - \hat{x}_i)v_i + \sum_{i=1}^{n} (x_i - \hat{x}_i)^2 < 0
\end{aligned}$$

We can see it is a stochastic variable with distribution $\mathcal{N}(\alpha_E, \beta_E\sigma^2)$, according to (2.13), so the probability that $\delta_E(\boldsymbol{r}, \hat{\boldsymbol{x}}) < \delta_E(\boldsymbol{r}, \boldsymbol{x})$ equals

$$Q\left(\frac{d_E(\boldsymbol{x}, \hat{\boldsymbol{x}})}{2\sigma}\right)$$

where the distance, $d_E(\boldsymbol{x}, \hat{\boldsymbol{x}})$, is defined by

$$d_E(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \frac{2\alpha_E}{\sqrt{\beta_E}}$$

where

$$\alpha_E = \sum_{i=1}^{n} (x_i - \hat{x}_i)^2$$

and

$$\beta_E = 4\sum_{i=1}^{n} (x_i - \hat{x}_i)^2$$

We define the minimum squared distance between any pair of codewords in this case, $d^2_{min,e}$, by

$$\begin{aligned}
d^2_{min,E} &= min_{\boldsymbol{x},\hat{\boldsymbol{x}} \in Q; \boldsymbol{x} \neq \hat{\boldsymbol{x}}} d^2_E(\boldsymbol{x}, \hat{\boldsymbol{x}}) \\
&= min_{\boldsymbol{x},\hat{\boldsymbol{x}} \in Q; \boldsymbol{x} \neq \hat{\boldsymbol{x}}} \sum_{i=1}^{n} (x_i - \hat{x}_i)^2
\end{aligned}$$

Which equals the squared Euclidean distance $\delta_e(\boldsymbol{x}, \hat{\boldsymbol{x}})$ between $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$. The word error rate (WER) over all coded sequences $\boldsymbol{x}$ is upper bounded by

$$WER < \frac{1}{|S|} \sum_{\boldsymbol{x} \in S} \sum_{\boldsymbol{x} \neq \hat{\boldsymbol{x}}} Q\left(\frac{d_E(\boldsymbol{x}, \hat{\boldsymbol{x}})}{2\sigma}\right)$$

For large signal-to-noise ratio, it is overbounded as[15]

$$WER < N_E Q\left(\frac{d_{min,E}}{2\sigma}\right) \tag{2.16}$$

The $N_E$ is the average number of nearest neighbors at minimum distance $d_{min,E}$ for every codeword. Assume the number of all codewords in the code book is $|S|$, and after using the exhaustive method in Matlab we know there are $N_t$ pairs of codewords at minimum distance $d^2_{min,E}$ and, so after we check each codeword's nearest neighbors at minimum distance one by one, we get the total number $N_E|S|$, where every codeword at minimum distance is counted twice, it is also two times of the number of codeword pairs at minimum distance, which is $2N_t$. Thus

$$N_E = \frac{2N_t}{|S|} \tag{2.17}$$

Then we can plug (2.17) into (2.16) to compute the upper bound of word error rate as a function of $SNR$, where the quantity $SNR$ is defined by

$$SNR(dB) = -20log_{10}\sigma$$

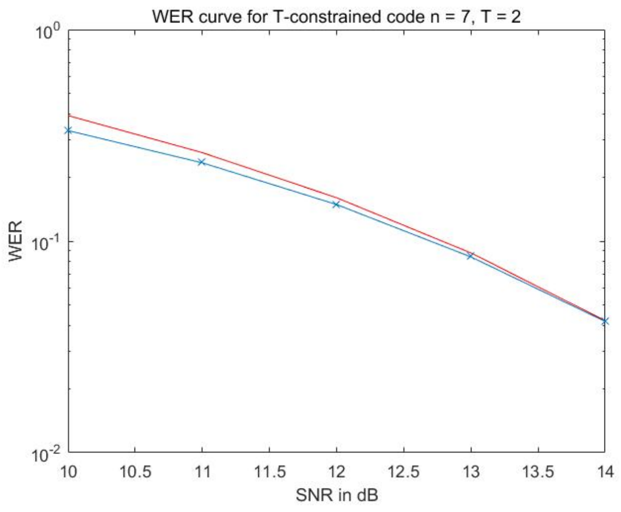

Figure 2.1: Word error rate(WER) of Euclidean detector as a function of signal-to-noise ratio(SNR) for T-constrained code($n = 7$, $T = 2$) in the ideal, matched case only with additive white Gaussian noise. The error performance was computed by upper bound (2.16) and computer simulations(lines with markers).

Figure 2.1 and 2.2 shows the word error rate theoretical upper bound and simulation curve for the T-constrained code and modified Hamming code(7,4,3) whose code length are the same, the modified Hamming code is the Hamming code with all zero and all one codewords excluded. For T-constrained code, the average number of nearest neighbors $N_E$ is 6.89, and the minimum squared distance $d^2_{min,E}$ is 1; For modified Hamming code(7,4,3), the average number of nearest neighbors $N_E$ is 6, and the minimum squared distance $d^2_{min,E}$ is 3. We can see for both codes the computer simulation (blue line with markers) matches theoretical computation very well, and the modified Hamming code has smaller code size but it overperforms T-constrained code because of its Larger minimum Euclidean distance.

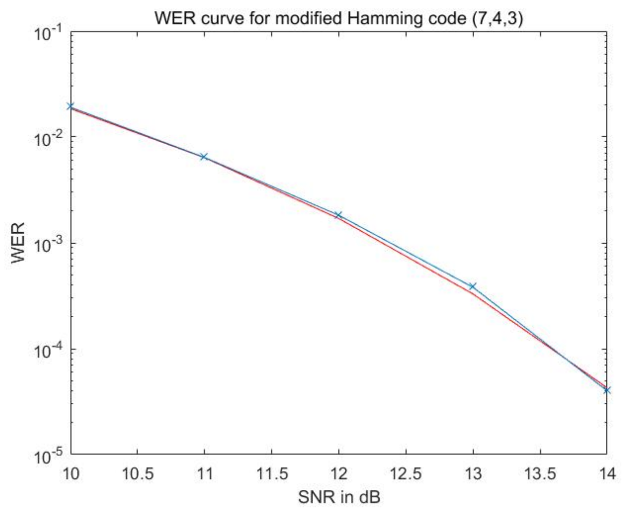

Figure 2.2: Word error rate(WER) of Euclidean detector as a function of signal-to-noise ratio(SNR) for modified Hamming code $(7,4,3)$ in the ideal, matched case only with additive white Gaussian noise. The error performance was computed by upper bound (2.16) and computer simulations(lines with markers).

# 3 Minimum Modified Pearson distance detection

In this chapter, we will investigate the minimum modified Pearson distance detection for different codes and different $q(q = 2$ and $q = 3)$ in offset mismatch only channel. In the offset only case, modified Pearson distance is applied instead of Pearson distance. The object is to make comparison between different Pearson codes and investigate which factor of a Pearson code may affect its error performance in the minimum modified Pearson distance detection.

## 3.1. Theoretical upperbound of word error rate

Recall the concept of binary T-Constrained Codes we introduced in chapter 2, Here we first assume the signal vector $\boldsymbol{x} \in S_1$ is sent , where all '1' codeword is excluded, and the channel is ideal and matched, only contains additive Gaussian noise. On the receiving end the modified Pearson distance (2.8) is applied to compute the distance between the received vector $\boldsymbol{r} = \boldsymbol{x} + \boldsymbol{v}$ and all the other codewords in $S_1$, the receiver decides that the codeword $\boldsymbol{x}_0$ was sent if (2.8) attains its least value for $\hat{\boldsymbol{x}} = \boldsymbol{x}_0$, that is

$$\boldsymbol{x}_0 = \arg\min_{\hat{\boldsymbol{x}}\in S_1} \delta_{MP}(\boldsymbol{r}, \hat{\boldsymbol{x}}) = \arg\min_{\hat{\boldsymbol{x}}\in S_1} \sum_{i=1}^{n} (r_i - \hat{x}_i + \bar{\hat{x}})^2$$

Because $r_i = x_i + v_i$, so that

$$\begin{aligned}\delta_{MP}(\boldsymbol{r}, \hat{\boldsymbol{x}}) &= \sum_{i=1}^{n} (x_i + v_i - \hat{x}_i + \bar{\hat{x}})^2 \\ &\equiv \sum_{i=1}^{n} (x_i + v_i - \hat{x}_i + \bar{\hat{x}} - \bar{x})^2\end{aligned}$$

Define $\boldsymbol{e} = \boldsymbol{x} - \hat{\boldsymbol{x}}$ and $\bar{e} = \bar{x} - \bar{\hat{x}}$, then

$$\delta_{MP}(\boldsymbol{r}, \hat{\boldsymbol{x}}) \equiv \sum_{i=1}^{n} (e_i - \bar{e} + v_i)^2$$

Same as (2.14), the detector errs if there is at least one codeword $\hat{\boldsymbol{x}} \in S_1$, $\hat{\boldsymbol{x}} \neq \boldsymbol{x}$, such that

$$\delta_{MP}(\boldsymbol{r}, \hat{\boldsymbol{x}}) < \delta_{MP}(\boldsymbol{r}, \boldsymbol{x})$$

$$2\sum_{i=1}^{n} v_i(e_i-\bar{e})+\sum_{i=1}^{n}(e_i-\bar{e})^2<0 \tag{3.1}$$

The left-hand side of (3.1) is a stochastic variable with distribution $\mathcal{N}(\alpha_{MP}, \beta_{MP}\sigma^2)$, where

$$\alpha_{MP}=\sum_{i=1}^{n}(e_i-\bar{e})^2$$

and

$$\beta_{MP}=4\sum_{i=1}^{n}(e_i-\bar{e})^2$$

We define the distance between the vectors $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ by

$$\begin{aligned} d_{MP}(\boldsymbol{x},\hat{\boldsymbol{x}}) &= \frac{2\alpha_{MP}}{\sqrt{\beta_{MP}}}=\sqrt{\sum_{i=1}^{n}(e_i-\bar{e})^2} \\ &= \sqrt{\sum_{i=1}^{n}e_i^2-2\bar{e}\sum_{i=1}^{n}e_i+\sum_{i=1}^{n}\bar{e}^2} \\ &= \sqrt{\sum_{i=1}^{n}e_i^2-n\bar{e}^2} \end{aligned} \tag{3.2}$$

Therefore, the word error rate over all coded sequences $\boldsymbol{x}$ is now upper bounded by

$$WER<\frac{1}{|S_1|}\sum_{\boldsymbol{x}\in S_1}\sum_{\boldsymbol{x}\neq\hat{\boldsymbol{x}}}Q\left(\frac{d_{MP}(\boldsymbol{x},\hat{\boldsymbol{x}})}{2\sigma}\right)$$

For asymptotically large signal-to-noise-ratio's, i.e. $\sigma \ll 1$,the WER is over bounded by

$$WER<N_{MP}Q\left(\frac{d_{min,MP}}{2\sigma}\right) \tag{3.3}$$

It is clear that a pair of codewords $\hat{\boldsymbol{x}}$ of $\boldsymbol{x}$ at Euclidean distance two is also a pair of codewords at minimum distance $d_{min,MP}$. From [1] we can approximate the average number of neighbors, $N_{MP}$ at the minimum distance $d_{min,MP}$ in $S_1$ by

$$N_{MP}\approx\frac{2n(q-1)}{q}$$

In the worst scenario, $\sum_{i=1}^{n} e_i^2 = 1$ and $\bar{e}=-1/n$, and from (3.2) we obtain

$$d_{min,MP}=\sqrt{1-\frac{1}{n}}$$

Then (3.3) becomes

$$\text{WER}<\frac{2n(q-1)}{q}Q\left(\frac{1}{2\sigma}\sqrt{1-\frac{1}{n}}\right).$$

Let $n=7$, then the average number of nearest neighbors $N_{MP}$ is 7 and the minimum modified Pearson squared distance(or minimum squared distance for shorthand representation) $d^2_{min,MP}$ is 0.86.

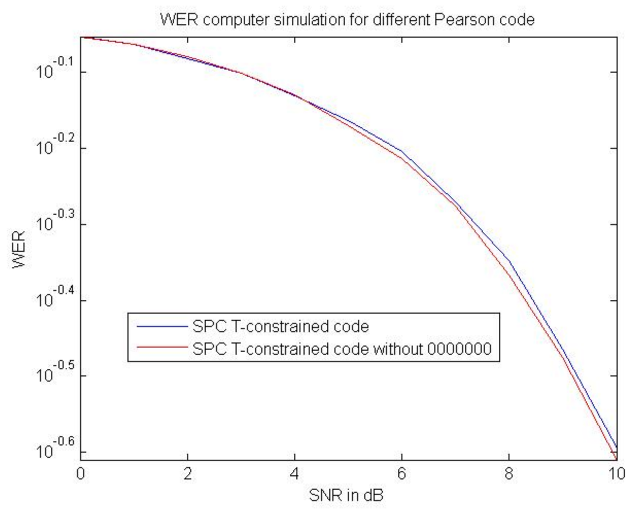

Figure 3.1: Word error rate(WER) of Pearson offset-resistant detection computer simulation, as a function of signal-to-noise ratio(SNR) for $q = 2, n = 7$ (single) parity check constrained code with and without all-zero codeword.

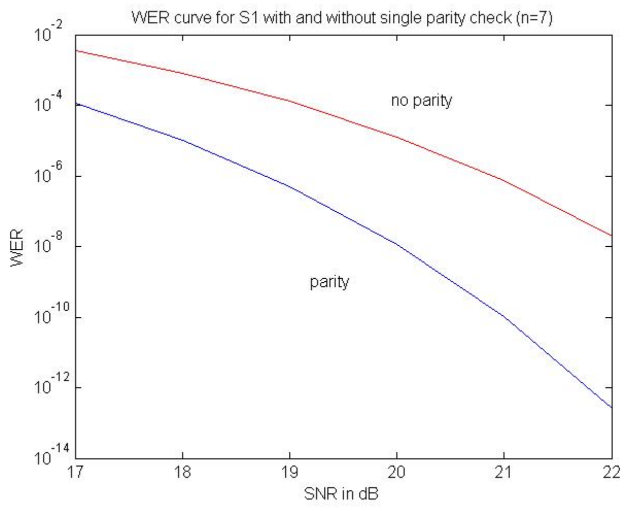

Figure 3.2: Word error rate(WER) of Pearson offset-resistant detection, as a function of signal-to-noise ratio(SNR) for $q = 2, n = 7$ with and without (single) parity check computed using upper bound (3.3) and (3.5).

Recall the single parity check $T$-constrained code, denoted by $S_{p,1}$, consists of codewords,$\boldsymbol{x}$, taken from $S_1$ that additionally satisfy the parity check

$$(\sum_{i=1}^{n} x_i) \mod q = a,$$

For the binary case, $q = 2$, we choose $a = (n+1) \bmod 2$, then the code size equals $|S_{p,1}| = 2^{n-1}$, so the worst scenario in this case is $\sum_{i=1}^{n} e_i^2 = 2$ and $\bar{e} = -2/n$, apply it in (3.2), we get the minimum distance in $S_{p,1}$

$$d_{min,MP} = \sqrt{2}\sqrt{1 - \frac{2}{n}} \tag{3.4}$$

Let $n = 7$, from (3.4) we know the minimum squared distance $d^2_{min,MP}$ is 1.42, according to the definition of the single parity check $T$-constrained code, all codewords with odd Hamming weight are removed, then we divide the code book by the Hamming weight of codewords$(0,2,4,6)$, so there are 4 sets of codewords: $\{0000000\}_0$, $\{0000011\}_2$, $\{0001111\}_4$, $\{0111111\}_6$. We know a pair of codewords $\hat{\boldsymbol{x}}$ of $\boldsymbol{x}$ at minimum distance $d_{min,MP}$ is also a pair of codewords at Euclidean distance two[1], in this case, a neighbor at Euclidean distance two from $\boldsymbol{x}$ is obtained by adding or subtracting '1' from each two symbols in $\boldsymbol{x}$, for example, the nearest neighbors of codeword '0000011' should be in the set of codewords with Hamming weight which is 4: $\{0001111\}_4$. We also notice that there should be as many '1's as possible overlapped between the neighbor and '0000011', so the last two symbols in its neighbor should be one, then the number of nearest neighbors is $C_5^2$. In this way we can get the total number $N_t$ of codewords pairs at the minimum distance $d_{min,MP}$ is

$$N_t = C_7^2 + C_7^2 C_5^2 + C_7^4 C_3^2 = 336$$

We notice that the squared distance between '0000000' and '0111111' is 0.86, which is smaller than the minimum squared distance computed by (3.4), to make it same as the computation result and easy to compare, we use $S_2$ instead of $S_1$ in the minimum modified Pearson distance detection for offset-only mismatch case, that is, both all '1' and all '0' codewords are excluded from the code book. Also we know that removing the all-zero codeword can increase the minimum distance $d_{min,MP}$ of the codewords, we did decoding simulation about the $WER$ performance for SPC $T$-constrained code with and without all-zero codeword, the results in Figure 3.1 shows that remove all-zero actually can slightly improve performance in high SNR. After the all-zero codeword is removed, the total number $N_t^{'}$ of codewords pairs at the minimum distance $d_{min,MP}$ is:

$$N_t^{'} = C_7^2 C_5^2 + C_7^4 C_3^2 = 315$$

so the average number of neighbors, $N_{MP}^{'}$, at minimum distance $d_{min,MP}$ in $S_{p,1}(n=7)$:

$$N_{MP}^{'} = \frac{2N}{|S_{p,1}|} = \frac{2\times 315}{63} = 10$$

Then we can compute the theoretical bound of word error rate for the single parity check $T$-constrained code$(n=7)$

$$WER < N_{MP}^{'} Q\left(\frac{d_{min,MP}}{2\sigma}\right) \tag{3.5}$$

Figure 3.2 shows the comparison of theoretical WER bound for $T$-constrained code with (all-zero codeword removed) and without single parity check when $n=7$. We can see that the performance of code with single parity check is better, but the code size is also smaller.

### 3.1.1. COMPARISON OF THEORETICAL WER BOUND

A co-set of a code $\boldsymbol{C}$ is the subset of the $n$-dimensional vector space $V_n$ given by

$$\boldsymbol{C} + \boldsymbol{a} = \{\boldsymbol{x} + \boldsymbol{a} | \boldsymbol{x} \in \boldsymbol{C}\}$$

where $\boldsymbol{a}$ is a vector in $V_n$. In this report we choose the vector $\boldsymbol{a}$ to be '1000000'. Here we calculate the theoretical WER of several Pearson codes: co-set of Hamming code(7,4,3), modified Hamming code(7,4,3), co-set of modified Hamming code(7,4,3). The computation results are shown in the Table 3.1.

Figure 3.3 shows the theoretical word error rate(WER) bound for the three Pearson codes computed using (3.3), we also add $T$-constrained codes(n=7) with and without single parity check in the comparison. We can see in all these five Pearson codes with the same length $n=7$, the modified Hamming code has the lowest bound, with the smallest size 14 and largest minimum squared distance 2.86; while the $T$-constrained code has the highest bound, with the largest size 127 and smallest minimum squared distance 0.86.

| | Average number of nearest neighbors $N_{MP}$ | Minimum squared distance $d^2_{min,MP}$ |
|---|---|---|
| co-set of Hamming code(7,4,3) | 2.5 | 1.71 |
| modified Hamming code(7,4,3) | 6 | 2.86 |
| co-set of modified Hamming code(7,4,3) | 0.86 | 1.71 |

Table 3.1: Comparison of different Pearson codes

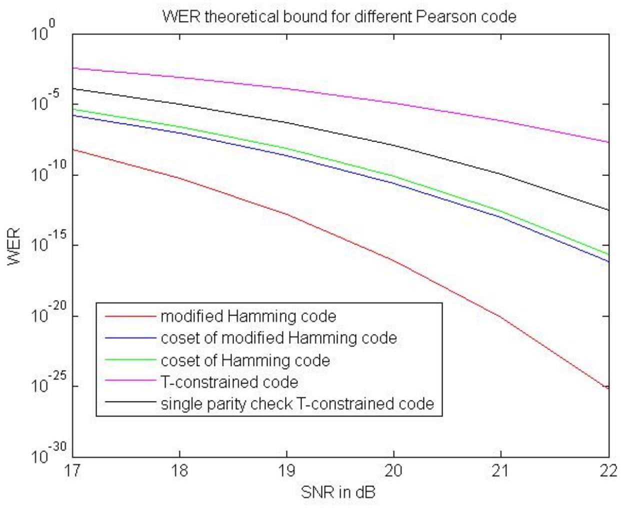

Figure 3.3: Word error rate(WER) bound of Pearson offset-resistant detection, as a function of signal-to-noise ratio(SNR) for $q = 2, n = 7$ of 5 different Pearson codes.

### 3.1.2. COMPARISON OF DECODING SIMULATION

Figure 3.4 shows the modified Pearson distance detection simulation curve for the five Pearson codes we mentioned above in relatively low signal-to-noise ratio. We can clearly see it matches well with the theoretical bound in high SNR, the modified Hamming code also have the best WER performance in the low SNR, and $T$-constrained code have the worst WER.

## 3.2. PERFORMANCE COMPARISON FOR MORE DIFFERENT MODIFIED BINARY CODES

In this section, we continue to investigate the performance of more different kinds of modified codes: Modified Hamming code(7,4,3), Modified Hamming code(15,11,3), and three modified Reed-Muller codes RM(1,3), RM(2,3), RM(1,4). All of these modified block codes above are original codes excluding all '0' and all '1' code sequence. The object is to compare the error performance of different Pearson code in other aspects: code rate, Hamming distance. From the chapter 2 We already know that the Reed-Muller code(1,3) is also called the extended Hamming code(8,4,4) which comes from the Hamming code(7,4,3) by adding an extra parity bit on top of its encoded word.

First we compare the modified Hamming codes (7,4) and (15,11), the WER bound and simulation results are shown in the Figure 3.5 and 3.6. As we can see, the bounds fits the simulation results well, and the performance of modified Hamming code(7,4) is better than the modified Hamming

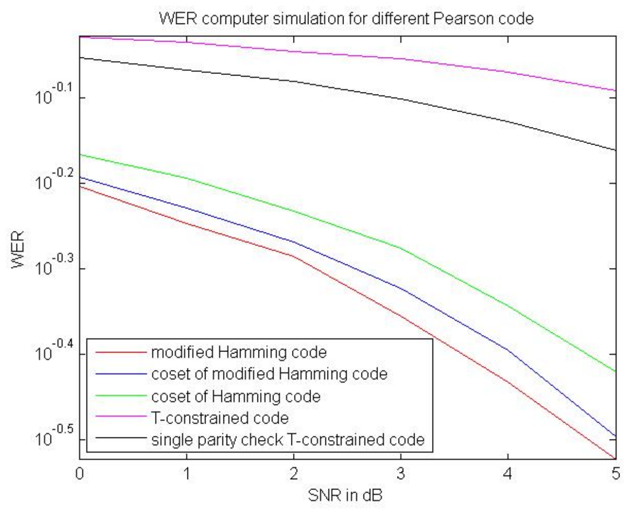

Figure 3.4: Word error rate(WER) computer simulation of Pearson offset-resistant detection, as a function of signal-to-noise ratio(SNR) for $q = 2, n = 7$ of 5 different Pearson codes, the message is generated with 10000 codewords.

code$(15,11)$ with the same minimum Hamming distance 3, but the later code has longer length which means more symbol error probability in the same channel condition compared with the former code, so the efficiency of error correcting is worse.

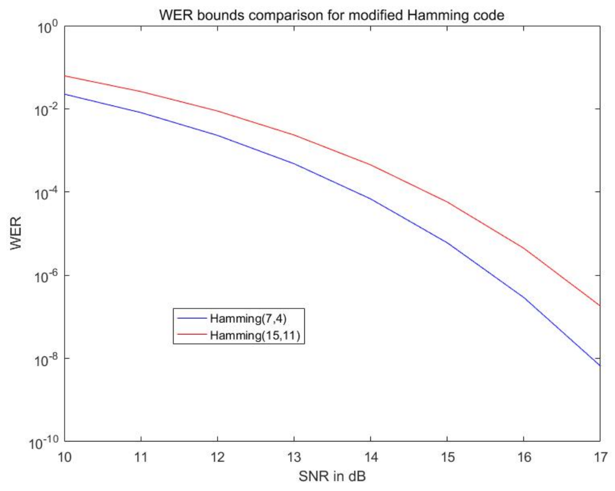

Figure 3.5: Word error rate(WER) bounds for Pearson offset-resistant detection, as a function of signal-to-noise ratio(SNR) for Hamming code $(7,4)$ and $(15,11)$ without both all-zero and all-one codewords.

Figure 3.7 and 3.8 shows the WER bound and simulation results comparison between modified Hamming code$(7,4)$ and modified Reed-Muller code$(1,3)$, we can see the bounds computation also fits well with the simulations, and the modified RM$(1,3)$ code, which is also extended Hamming code, performs better than modified Hamming code$(7,4)$, so adding extra parity check bits to increase Hamming distance also can improve the performance in Pearson detection for Hamming code(7,4).

To make it more comparable for these codes, we compute the code rate, minimum Hamming distance $d_{min,H}$, and minimum squared distance $d^2_{min,MP}$ for all these codes, which are listed in the Table 3.2 below.

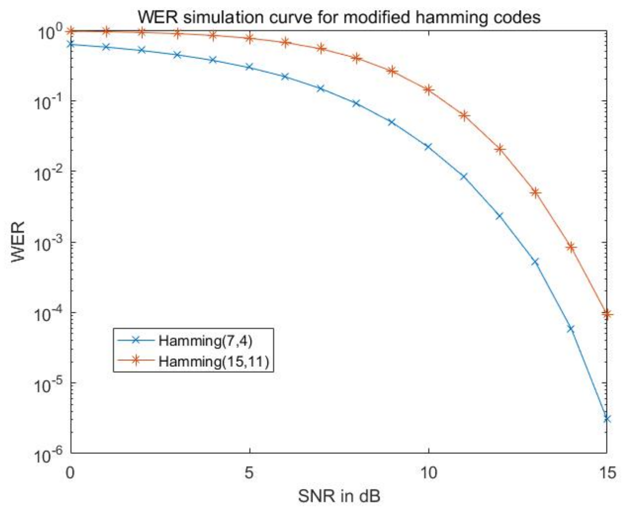

Figure 3.6: Word error rate(WER) simulations of Pearson offset-resistant detection, as a function of signal-to-noise ratio(SNR) for Hamming code $(7,4)$ and $(15,11)$ without both all-zero and all-one codewords.

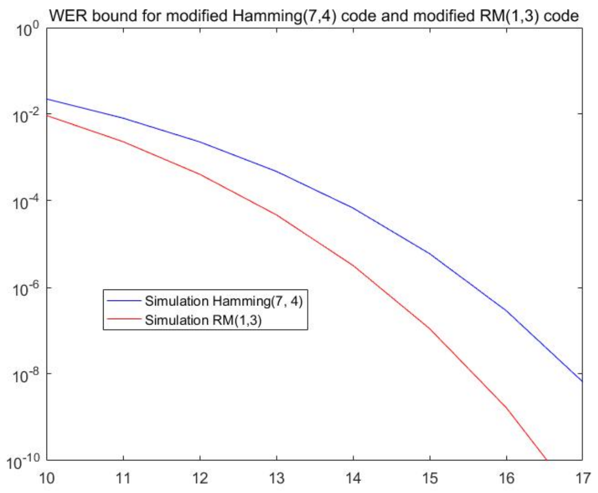

Figure 3.7: Word error rate(WER) bounds for Pearson offset-resistant detection, as a function of signal-to-noise ratio(SNR) for Hamming code $(7,4)$ and Reed-Muller code$(1,3)$ without both all-zero and all-one codewords.

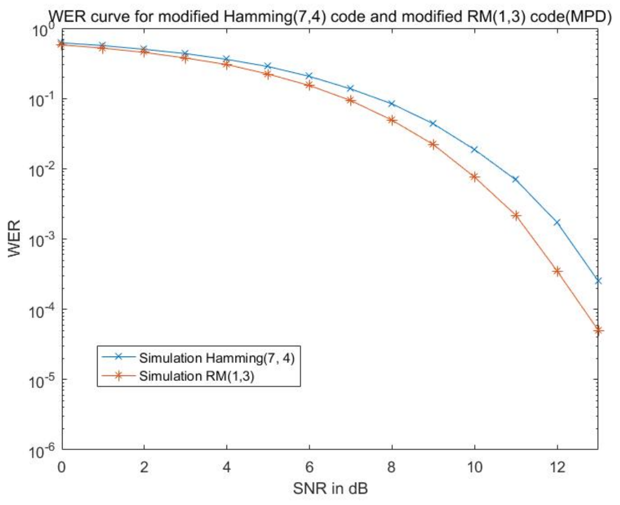

Figure 3.8: Word error rate(WER) bounds for Pearson offset-resistant detection, as a function of signal-to-noise ratio(SNR) for Hamming code (7,4) and Reed-Muller code(1,3) without both all-zero and all-one codewords.

| | Code rate | $d_{min,H}$ | $d^2_{min,MP}$ | Performance ranking |
|---|---|---|---|---|
| MH(7,4) | 0.544 | 3 | 2.86 | 3 |
| MH(15,11) | 0.733 | 3 | 2.4 | 4 |
| MRM(1,3) | 0.476 | 4 | 4 | 2 |
| MRM(2,3) | 0.872 | 2 | 1.5 | 5 |
| MRM(1,4) | 0.307 | 8 | 8 | 1 |

Table 3.2: Comparison of different codes

From the table we can see, the Pearson code with smaller code rate always has better performance in offset-only Pearson detection, and the Hamming distance generally follow the same trend with the modified Pearson distance. In the next chapter, we will investigate the relations between modified Pearson squared distance and Hamming distance in offset-only Gaussian channel.

## 3.3. MODIFIED PEARSON DETECTION FOR TERNARY CODES

In this section, we will investigate the offset-only minimum modified Pearson distance detection for T-constrained codes and single parity check codes in the case of $q$ is 3, with the code length $n = 4$, so the codewords are chosen over the alphabet $Q = \{0,1,2\}$ and the size of a full code book is $|S| = q^n = 3^4 = 81$. Note that the ternary T-constrained codes($T = 1$) is not Pearson code, because it doesn't satisfy the first property of Pearson code in section 2.3 of chapter 2, but it is still suitable for the minimum modified Pearson distance detection. Recall that a single parity check $T$-constrained code consists of codewords, $x$, taken from $S$, that additionally satisfy the parity check equation:

$$(\sum_{i=1}^{n} x_i) \quad mod \quad q = a$$

In this case, $a \in \{0,1,2\}$, so the full code book is divided into 3 single parity check codes which have the same size of 27, and the codewords in each single parity check code have 4 symbol compositions over the alphabet $Q$ which is shown in the Table 3.3:

First, we consider the $T$-constrained code when $T = 1$, which means all codewords wherein the

| mod 3 = 0 | mod 3 = 1 | mod 3 = 2 |
|---|---|---|
| 0 0 0 0 | 1 1 1 1 | 1 1 1 2 |
| 0 1 1 1 | 0 0 0 1 | 0 0 1 1 |
| 0 2 2 2 | 1 2 2 2 | 0 0 0 2 |
| 0 0 2 1 | 0 1 2 1 | 0 2 2 1 |
| 1 2 1 2 | 0 0 2 2 | 2 2 2 2 |

Table 3.3: All the symbol compositions of full ternary code book with code length $n = 4$

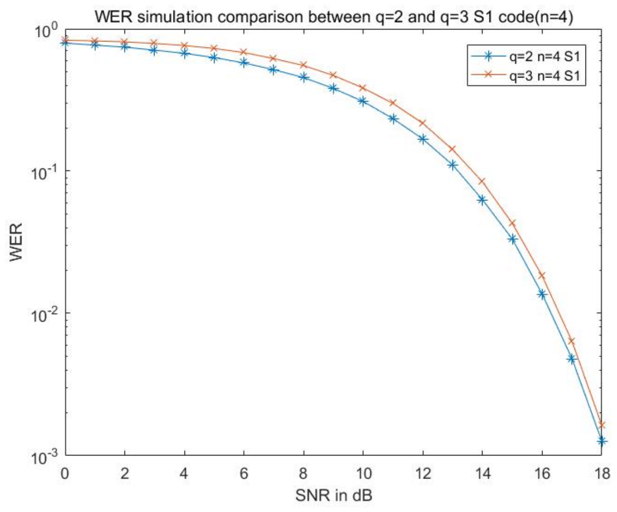

Figure 3.9: Word error rate(WER) simulations for Pearson offset-resistant detection, as a function of signal-to-noise ratio(SNR) for single parity check $T = 1$-constrained codes($n = 4$) without both all '$0'$' and all '$q-1'$' codewords for both $q = 2$ and $q = 3$.

symbol '$0'$' appears at least once. The computation results show that, for $q = 2$, the code book size $|S_1| = 15$, the minimum squared distance $d^2_{min,MP}$ among all codewords is 0.75, and the average number of nearest neighbors is 4.27; for $q = 3$, the code book size $|S_1| = 65$, the minimum squared distance $d^2_{min,MP}$ among all codewords is also 0.75, and the average number of nearest neighbors is 5.66. The Figure 3.9 shows the WER performance simulation of SPC $T = 1$ constrained code for both $q = 2$ and $q = 3$, we can see as the $q$ increase to 3, the performance of the Pearson code is worse than the binary case because the average number of nearest neighbors increases with the same minimum squared distance 0.75. Then the symbol compositions of three SPC codes becomes:

| $SPC_{1a}$ | $SPC_{1b}$ | $SPC_{1c}$ |
|---|---|---|
| mod 3 = 0 | mod 3 = 1 | mod 3 = 2 |
| 0 0 0 0 | 0 0 0 1 | 0 0 1 1 |
| 0 1 1 1 | 0 1 2 1 | 0 0 0 2 |
| 0 2 2 2 | 0 0 2 2 | 0 2 2 1 |
| 0 0 2 1 | | |

Table 3.4: The symbol compositions of different SPC T-constrained codes with $T = 1$ and $n = 4$

The Figure 3.10 shows the WER simulation performance comparison for these three Pearson codes, it shows that $SPC_{1a}$ is better than other two codes. Now consider the $T = 2$, by definition, both the symbols '$0'$' and '$q-1'$' appear at least once. So the size of code book decrease to $S_2 = 50$, the minimum squared distance $d^2_{min,MP}$ is still 0.75, but the average number of nearest neighbors is

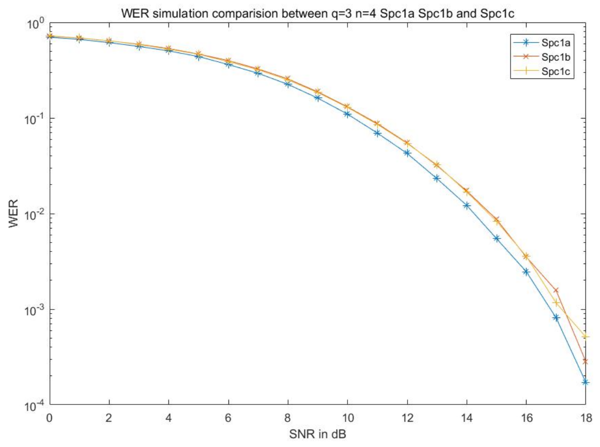

Figure 3.10: Word error rate(WER) simulations of Pearson offset-resistant detection, as a function of signal-to-noise ratio(SNR) for the three single parity check $T = 1$ constrained codes with $n = 4$.

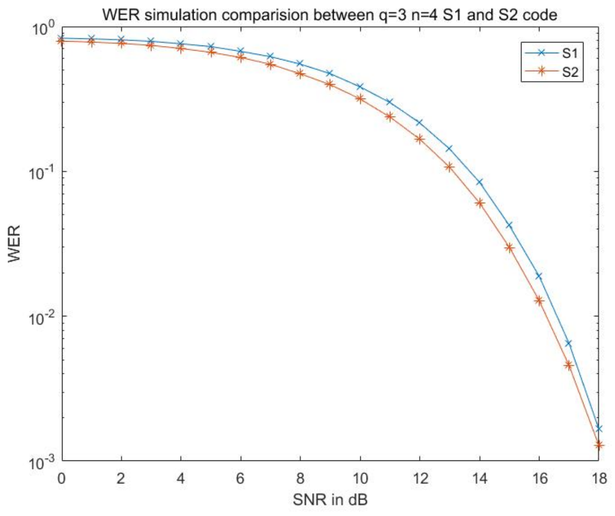

Figure 3.11: Word error rate(WER) simulations of Pearson offset-resistant detection, as a function of signal-to-noise ratio(SNR) for both $T = 1$ and $T = 2$ constrained codes with $n = 4$, $q = 3$.

3.84. Figure 3.11 shows the simulation performance of both $T = 1$ and $T = 2$ in case of $q = 3$, we can see the $T = 2$ constrained codes is better in price of smaller code book size.

Then in this case($T = 2$) the symbol compositions of three SPC codes becomes:

| $SPC_{2a}$ mod 3 = 0 | $SPC_{2b}$ mod 3 = 1 | $SPC_{2c}$ mod 3 = 2 |
|---|---|---|
| 0 2 2 2 | 0 1 2 1 | 0 0 0 2 |
| 0 0 2 1 | 0 0 2 2 | 0 2 2 1 |

Table 3.5: The symbol compositions of different SPC T-constrained codes with $T = 2$ and $n = 4$

The Figure 3.12 shows the WER simulation performance comparison for these three Pearson codes,it shows that $SPC_{2a}$ and $SPC_{2c}$ code are better than $SPC_{2b}$ code.

The Table 3.6 shows the information of all these codes above. As we can see, the codes $SPC_{2a}$

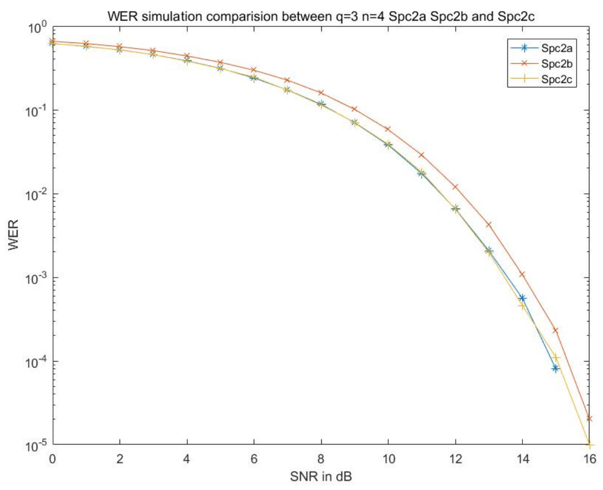

Figure 3.12: Word error rate(WER) simulations of Pearson offset-resistant detection, as a function of signal-to-noise ratio(SNR) for the three single parity check $T = 2$ constrained codes with $n = 4$.

and $SPC_{2c}$ is the best two among these codes. The performance not only depends on the minimum squared distance $d^2_{min,MP}$: when two codes have the same minimum squared distance, the one with less average nearest neighbors is always better. Another interesting fact is when the $S_1$ is divided into three SPC codes, the average number of nearest neighbors decrease dramatically while $d^2_{min,MP}$ stays as 0.75, this is because $d^2_{min,MP}$ is achieved when the hamming distance of two codewords is 1(or 3) and the average symbol value difference between them are $\frac{1}{4}$(or $\frac{3}{4}$), then we can see the majority of nearest neighbors are divided into other different SPC codes.

| | Size | Minimum squared distance $d^2_{min,MP}$ | Average number of nearest neighbors $N_{MP}$ |
|---|---|---|---|
| $S_1$ | 65 | 0.75 | 5.66 |
| $SPC_{1a}$ | 21 | 0.75 | 0.7619 |
| $SPC_{1b}$ | 22 | 0.75 | 1.0909 |
| $SPC_{1c}$ | 22 | 0.75 | 1.0909 |
| $S_2$ | 50 | 0.75 | 3.84 |
| $SPC_{2a}$ | 16 | 2 | 2.25 |
| $SPC_{2b}$ | 18 | 2 | 5.33 |
| $SPC_{2c}$ | 16 | 2 | 2.25 |

Table 3.6: Comparison of different codes($q = 3$)

# 4

# Modified Pearson squared distance vs Hamming distance

In this chapter we continue to investigate effect of the Hamming distance on the error performance of a Pearson code in offset-only Gaussian channel by studying the relations between Hamming distance $d_H$ and the modified Pearson squared distance $d^2_{MP}$(or minimum squared distance for shorthand representation).

## 4.1. Modified Pearson squared distance vs Hamming distance

Consider a binary communication code book with full size but excluded the all one sequence and all zero sequence, $S$, of chosen binary codewords over the binary alphabet $Q = \{0,1\}$. Suppose any two codewords with length $n$: $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)$ and $\boldsymbol{b} = (b_1, b_2, \ldots, b_n)$ of the code book, to make it more simple to compare, we can arrange them in this way:

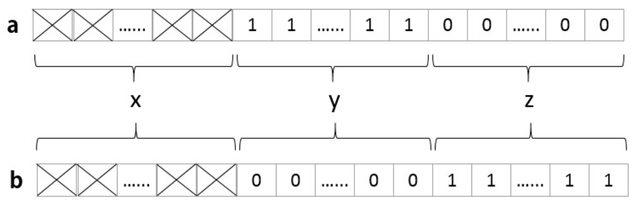

Figure 4.1: The comparison of two codewords with the same length by rearrange their symbols to three parts: same symbol value positions; ‘1’ to ‘0’; ‘0’ to ‘1’.

we can see there are three different kinds of positions in the comparison of two code sequences, the quantities of these three kinds of positions are $x$, $y$, $z$ respectively. The first one represents identical symbols in both code sequences, where the codewords $\boldsymbol{a}$ and $\boldsymbol{b}$ both have the same(both ‘0’ or ‘1’) bits, the second one are the positions where the codeword $\boldsymbol{a}$ have ‘1’ while $\boldsymbol{b}$ have ‘0’, and the last one are the positions where the codeword $\boldsymbol{a}$ have ‘0’ while $\boldsymbol{b}$ have ‘1’. Clearly we can rearrange the order of these positions but keep the same Hamming weight.in the same. So there are many different codeword pairs $(\boldsymbol{a}, \boldsymbol{b})$ in the same constant composition subset pair[16]. Recall that in the chapter 3, the modified Pearson squared distance between any possible pair of codewords in minimum distance decoding with modified Pearson distance detector for the offset only mismatch case is defined as:

$$d^2_{MP}(\boldsymbol{a},\boldsymbol{b}) = \sum_{i=1}^{n}(e_i - \bar{e})^2 \tag{4.1}$$

where $\boldsymbol{e} = \boldsymbol{a} - \boldsymbol{b}$; $\bar{e} = \bar{a} - \bar{b}$, and $\bar{a} = \frac{1}{n}\sum_{i=1}^{n} a_i$; $\bar{b} = \frac{1}{n}\sum_{i=1}^{n} b_i$. We know that it is the minimum squared distance among all possible codeword pairs of the code book that determines the performance of the Pearson code, so if we can find the relations between the squared distance $d^2_{MP}$ and Hamming distance $d_H$, it could be more simple and direct to reshape and optimize the code book to improve the code performance in offset mismatch case. According the properties of Hamming distance, it can be written in the binary case as:

$$d_H(\boldsymbol{a},\boldsymbol{b}) = \sum_{i=1}^{n}(a_i - b_i)^2$$

If $\bar{a} = \bar{b}$, the squared distance $d^2_{MP}(\boldsymbol{a},\boldsymbol{b})$ reveals the same property as the Hamming distance; If $\bar{a} \neq \bar{b}$, we suppose the Hamming weight difference between $\boldsymbol{a}$ and $\boldsymbol{b}$ is $m$, then we get:

$$\begin{cases} y - z = m \\ x + y + z = n \end{cases}$$

therefore:

$$\begin{cases} y = \frac{m+n-x}{2} \\ z = \frac{n-x-m}{2} \end{cases} \tag{4.2}$$

we also notice that in the case of the first kind of positions(Identical), $e_i - \bar{e} = \bar{a} - \bar{b}$; in the second kind of positions('1' to '0') it is: $e_i - \bar{e} = 1 - (\bar{a} - \bar{b})$; and in the third kind of positions('0' to '1') it is: $e_i - \bar{e} = -1 - (\bar{a} - \bar{b})$. So the equation (4.1) of squared distance can be written as:

$$d^2_{MP}(\boldsymbol{a},\boldsymbol{b}) = (\bar{a} - \bar{b})^2 \cdot x + [1 - (\bar{a} - \bar{b})]^2 \cdot y + [1 + (\bar{a} - \bar{b})]^2 \cdot z$$

Because $\bar{a} - \bar{b} = \frac{m}{n}$, plug it and (4.2) into the equation above, we can get:

$$\begin{aligned} d^2_{MP}(\boldsymbol{a},\boldsymbol{b}) &= \frac{m^2}{n^2} \cdot x + \frac{(n-m)^2}{n^2} \cdot \frac{n+m-x}{2} + \frac{(n+m)^2}{n^2} \cdot \\ &\quad \frac{n-m-x}{2} \\ &= \frac{m^2 x}{n^2} + \frac{2n(n^2-m^2) - 2(n^2+m^2)x}{2n^2} \\ &= \frac{2n(n^2-m^2) - 2n^2 x}{2n^2} \\ &= \frac{n^2 - m^2}{n} - x \end{aligned}$$

since we have: $d_H(\boldsymbol{a},\boldsymbol{b}) = n - x$, finally it can be derived as:

$$\boxed{d^2_{MP}(\boldsymbol{a},\boldsymbol{b}) = d_H(\boldsymbol{a},\boldsymbol{b}) - \frac{m^2}{n}} \tag{4.3}$$

The equation (4.3) clearly reveals the relations between the squared distance and Hamming distance: for a Pearson code with length $n$, the modified Pearson squared distance $d^2_{MP}$ between any possible pair of codewords not only depends on their Hamming distance $d_H$ but also their Hamming weight difference $m$. Figure 4.2 shows the relations between the modified Pearson squared distance and Hamming distance, the five lines with different color represent the equation (4.3) with

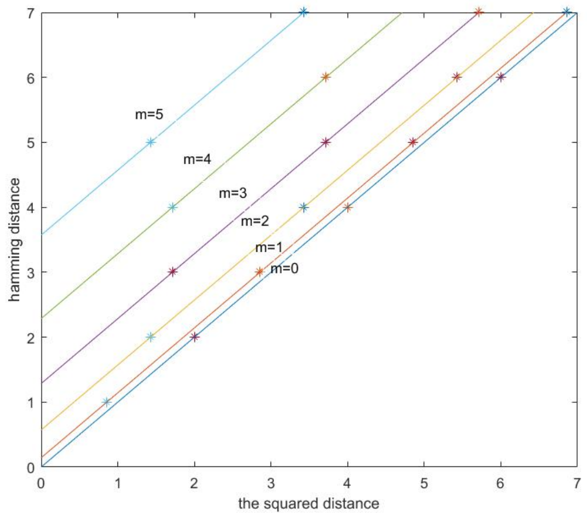

Figure 4.2: Each dots represents the set of codeword pairs with the corresponding Hamming distance and squared distance in the full code book with length $n = 7$, only the all 0 and all 1 codewords are excluded.

$m \in \{0, 1, 2, 3, 4, 5\}$. We can see the Figure 4.2 fits very well with equation (4.3), therefore, there are **two strategies** to improve a Pearson code in offset-only Gaussian channel: Firstly, we note that in each set of codewords in the full code book with fixed $m$, there is a positive correlation between the $d^2_{MP}$ and $d_H$, which means we can improve the code performance by excluding the 'bad' codewords with lower Hamming distance from the code book. On the other hand, we can also reduce the maximum Hamming weight difference by deleting some specific codeowrds to increase the minimum modified Pearson squared distance, this two strategies used for optimizing a Pearson code book are discussed in the next section.

## 4.2. ALGORITHM FOR DESIGNING OFFSET-ONLY PEARSON CODE

In the previous section we conclude that both minimum Hamming distance and maximum Hamming weight difference are the key factors that can determine the performance of a Pearson code in offset-only Gaussian channel, in this case we can optimise a code according to its weight distribution. The weight distribution of a full code book with length $n$ where all '$0'$' and all '$1'$' code sequences are excluded is as shown in Figure 4.3 and Figure 4.4:

So we can divide all words into many sections by their different Hamming weight. According to the equation (4.3), the weight difference between any possible codewords should as small as possible, and from the distribution it shows that the set of weight $\frac{n}{2}$ in the middle contains the most codewords, to always get a larger code size, we should delete the codewords from both edges of the weight distribution alternately. In an extreme case, where let $m = 0$, only the codewords from in the center are left, then increase the minimum Hamming distance by deleting codewords in the center section. Here we propose an algorithm to improve the performance of a Pearson code in offset-only Gaussian channel, as shown in the Table 4.1.

To remove the codewords in a most efficient way and get largest number of remained codewords, it's necessary to compute the performance of the two strategies, Consider a full code book with codeword length $n = 6$ and excluding the all 0 and all 1 codewords, we divide them into 5

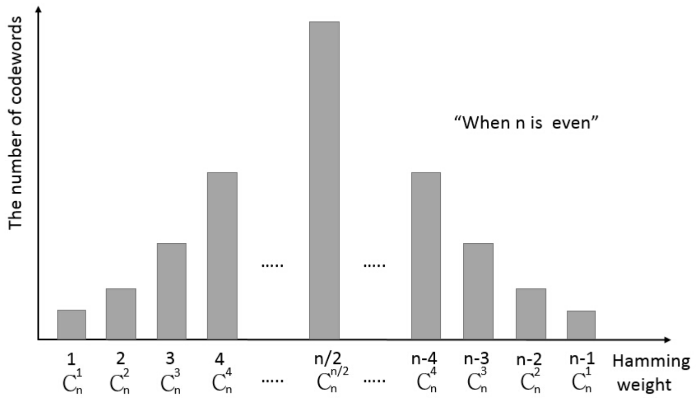

Figure 4.3: The weight distribution of full set of code book with even code length $n$, all '$0'$' and all '$1'$' code sequences are excluded.

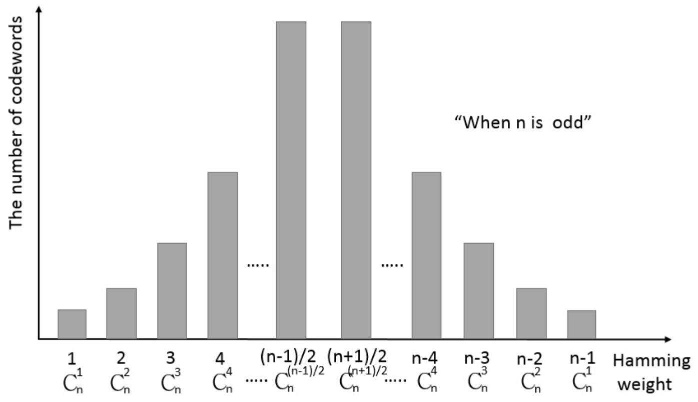

Figure 4.4: The weight distribution of full set of code book with odd code length $n$, all '$0'$' and all '$1'$' code sequences are excluded.

```
Algorithm: Best scheme to optimize a given Pearson
code conditionally in offset-only Gaussian channel
Input : A bad Pearson code with length $n$;
        Object minimum squared distance $d_{MP}^2$
        Hamming distance $d_H$
Output : A Pearson code with length $n$ and object $d_{MP}^2$
and largest size $N_{max}$
1: Compute the weight distribution of given code;
2: For all $d_H \in [0, n]$ do:
      Compute the m for each $d_H$ by solving the equation:
            $d_{MP}^2 = d_H - \frac{m^2}{n}$
3: For each set of $(d_H, m)_i$, $i \in [0, n]$ do:
      Delete the codewords according the weight distribution using the two strategies,
      Get the final code size $N_i$ and defined the scheme $S_i$;
4: Compute the largest code size $N_{max}$, get the optimize scheme $S_{max}$;
5: Output the optimized Pearson code with object $d_{MP}^2$ and largest size $N_{max}$;
```

Table 4.1: The Best scheme to optimize a given Pearson code conditionally in offset-only Gaussian channel

sections(combinations) according to their Hamming weight: {000001}, {000011}, {000111}, {001111}, and {011111}, we know the minimum Hamming distance is 1, and the number of nearest neighbors for each section are: 5, 6, 6, 6, 5 respectively, then we come up with five scenarios for the comparison:

- *Scenario 1:* Always remove the codewords with most nearest neighbors(Hamming distance) alternately from both edges of the weight distribution;
- *Scenario 2:* Always remove the codewords with most nearest neighbors(Hamming distance) in the central of the weight distribution;
- *Scenario 3:* Always remove the codewords with most nearest neighbors(Hamming distance) and lowest Hamming weight in weight distribution;
- *Scenario 4:* Remove the codewords that have nearest neighbors(Hamming distance) alternately from both edges of the weight distribution;
- *Scenario 5:* Remove the codewords alternately from both edges of the weight distribution

The scenarios *1,2,3* and *4* are to remove codewords based on minimum Hamming distance(strategy 1) but in different ways; the scenario *5* is to remove codewords based on the maximum Hamming weight difference(strategy 2). Figure 4.5 shows us the performance comparison of these five scenarios.

As we can see, the performance of scenario 4 and 5 is the same, because in such alternate removement, every codewords always have nearest neighbors at minimum Hamming distance. We note that the scenario 1, 2 and 3 performs better than scenario 5, which means that improving Hamming distance(strategy 1) could always be more efficient than reducing the maximum Hamming

weight difference(strategy 2), but removing codewords with most nearest neighbors from different location could lead different efficiency, we know the codewords with combinations: '000011$'$, '000111$'$, '001111$'$ have most nearest neighbors which is 6, we can consider the whole code book as a graph, every codeword is a node and only the nearest neighbors are connected as shown in the Figure 4.6.
Scenario 1 and 3 perform the same because we always remove the codewords with degree 6 until the minimum Hamming distance improves to next level, which means removing 6 links every time until the entire graph has been completely fragmented and there is no links between any nodes. while in scenario 2, we can see after about 20 codewords are removed, the maximum degree of all codewords drops from 6 to 5, then every removement could only involve codewords with 5 nearest neighbors, that is, 5 links in each removement, thus more codewords should be removed to improve the minimum Hamming distance and this clearly decrease the efficiency. Therefore, although removing codewords with most nearest neighbors in different way could lead to different results, but the most efficient way to improve the minimum Hamming distance is always remove the codewords with the most nearest neighbors and smallest average neighbors number of its neighbors.

However, we found there are only some bounds[17][18][19] on the size of a code with given code length and Hamming distance, and it is almost impossible to optimise a block code to one specific object minimum Hamming distance by removing codewords in the algorithm we proposed before, these two strategies can only improve the Pearson code, but for Pearson code design, it is more important to understand the structure of the code, the next section will give some good choices for the Pearson code design by study the relations between weight pillars in the weight distribution and minimum modified Pearson distance.

## 4.3. HOW TO CHOOSE THE PILLAR

From the view of the weight distribution for full set of a Pearson code whose codeword length is $n$, the codeword pair with the minimum modified squared Pearson distance could comes from the same weight pillar or different weight pillars, the minimum modified squared Pearson distance between codewords in a full set of weight pillar is 2, but $d^2_{MP}$ between different full size weight pillars can vary very much, therefore, how to choose different weight pillars to construct a Pearson code is another important question for the code designer.
Consider two codewords $\boldsymbol{a}$ and $\boldsymbol{b}$ from two different full set weight pillars respectively, and the

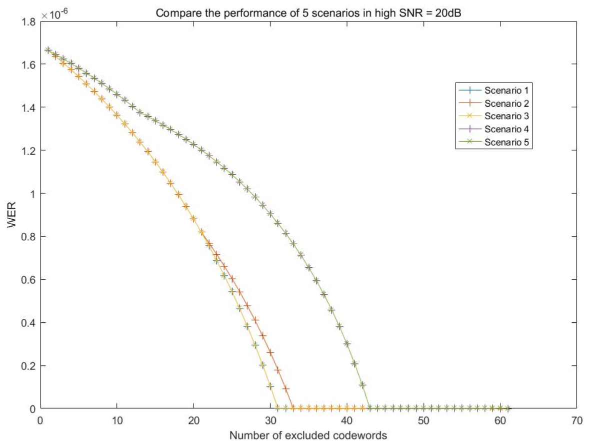

Figure 4.5: Compute and compare the theoretical bound of the codewords in the 5 scenarios with different strategies, to make it close to simulation results, the SNR is 20dB.

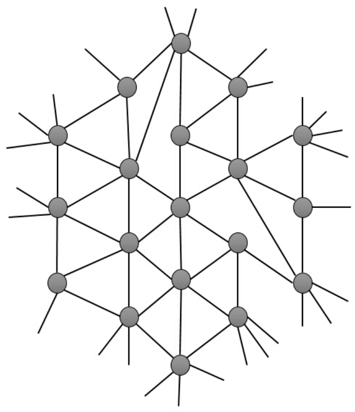

Figure 4.6: Consider the code book as a graph and each codewords as a node, and all node only connect to its nearest neighbors at minimum Hamming distance

weight difference of these two pillars is $x$, we can minimize their modified squared Pearson distance by minimizing the Hamming distance between $\boldsymbol{a}$ and $\boldsymbol{b}$ and choosing them like the Figure 4.7:

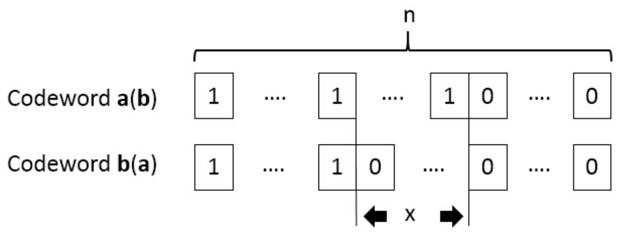

Figure 4.7: Choose two codewords with minimum Hamming distance, which also have the minimum modified squared Pearson distance

In this case, the hamming distance between $\boldsymbol{a}$ and $\boldsymbol{b}$ is equal to the their weight difference, so we have:

$$min(d_{MP}^2(\boldsymbol{a},\boldsymbol{b})) = x - \frac{x^2}{n} \tag{4.4}$$

(4.4) is the equation of a Parabola, and the parabola opens downward, so it has a peak which is the maximum value of the modified Pearson squared distance between $\boldsymbol{a}$ and $\boldsymbol{b}$, and it is $\frac{n}{4}$, the weight difference of $\boldsymbol{a}$ and $\boldsymbol{b}$ is $\frac{n}{2}$. For example, when the code length $n$ is 10, the relations between minimum modified Pearson squared distance and the weight difference between weight pillars are shown in the Figure 4.8, We can see, every two full set pillars with weight difference 5 have the largest minimum modified Pearson squared distance 2.5, and the minimum modified Pearson squared distance of pillar pairs with weight difference 1 is the smallest. Therefore, to improve a Pearson code, shrinking its weight range is not enough, it's weight distribution should also be more sparse than before.

## 4.4. CASE STUDY

The balanced code is a well studied Pearson code, and there are several prior art constructions of it. By definition, the binary balanced code contains codewords that have equal numbers of '1's

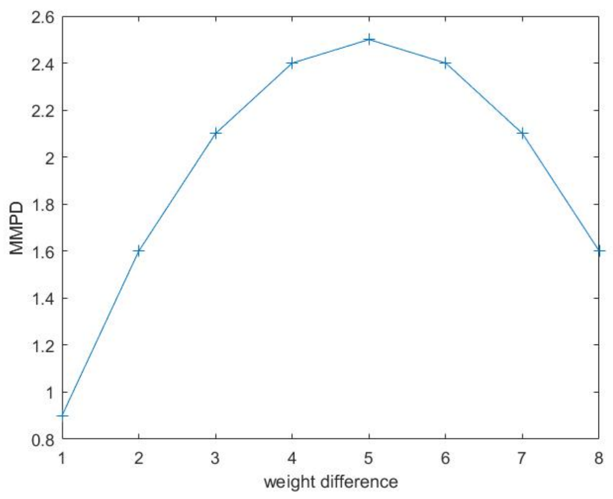

Figure 4.8: The relations between minimum modified Pearson squared distance and the weight difference between weight pillars of Pearson code with code length $n = 10$.

and ’0’s, which means it consists of all codewords in the central pillar of the weight distribution. For a full set of balanced code with code length $n$,$n$ is even, it code rate is:

$$R_{bc} = \left( log_2 \binom{n}{\frac{n}{2}} \right) / n \tag{4.5}$$

and the redundancy is:

$$H_{bc} = n - \left( log_2 \binom{n}{\frac{n}{2}} \right) \tag{4.6}$$

if the code length $n$ satisfy: $n \gg 1$, the redundancy $H_{bc}$ can be approximated by[20]:

$$H_{bc} \approx \frac{1}{2} log_2 n + 0.326 \tag{4.7}$$

Since it can be easily constructed, and it’s minimum modified Pearson squared distance is exactly its minimum Hamming distance, the balanced code is a fast lower bound approach for code designer given the object code with code length $n$ and minimum modified Pearson squared distance larger than or equal to 2, After check the table of constant weight code, we can compare lower bounds of code rate with codeword length $n$ grows for balanced code with different minimum Hamming distance.

From the Figure 4.9 we can see that, higher minimum Hamming distance always means lower code rate, and the trend for different minimum Hamming distance is almost the same.
While the Figure 4.10 below shows lower bounds of code rate with minimum Hamming distance grows for balanced code with different length. We can see that the lower bounds also almost follows the same trends.
However, is the binary balanced code the largest size Pearson code with minimum modified Pearson squared distance 2? can we add more codewords to the balanced code to construct a new Pearson code with larger code rate, but still hold the minimum Hamming distance 2?

**Theorem 1.** For the full set binary balanced code $\mathscr{C}$ in a full set of Pearson binary code book with all zero and one sequences excluded, and the codeword length $n > 8$, there always exist a codeword $\boldsymbol{c}$ out of the balanced code $\mathscr{C}$ so that we can build a new code $\mathscr{D}$ contains both $\mathscr{C}$ and $\boldsymbol{c}$, and

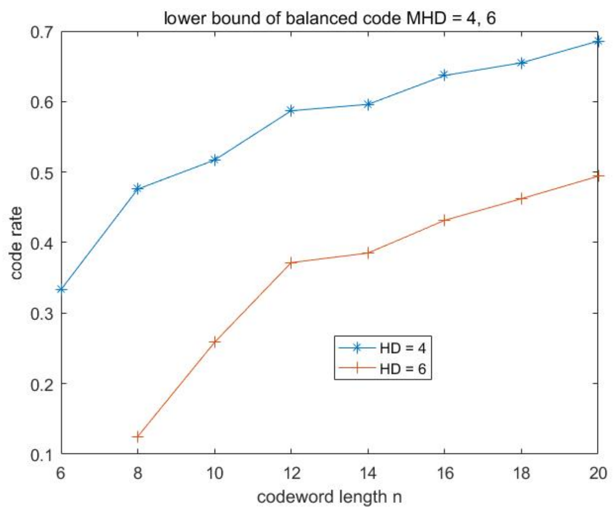

Figure 4.9: Lower bound of code rate for balanced code given its codeword length $n \in [6,20]$ with minimum modified squared Pearson distance 4 and 6

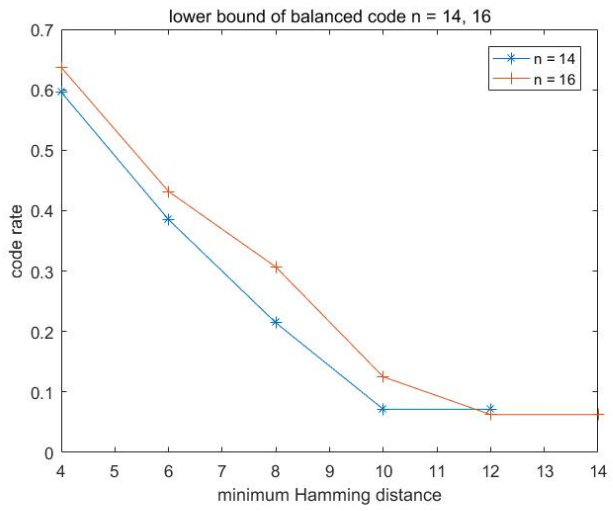

Figure 4.10: Lower bound of code rate for balanced code given its minimum modified Pearson squared distance (Hamming distance) $d_{mpd} \in [4,14]$ with codeword length 14 and 16

the minimum modified Pearson squared distance of code $\mathscr{D}$ is still the same as code $\mathscr{C}$ but has larger code rate.

**Proof.** Suppose the Hamming weight of the codeword we choose is $w = 1$, and codeword length is $n$, the Hamming weight of the balanced code $\mathscr{C}$ is $\frac{n}{2}$. So minimum the modified Pearson squared distance between the codeword we choose and any codewords of $\mathscr{C}$ is:

$$\boldsymbol{min}(d_{MP}) = \boldsymbol{min}(d_H) - \frac{(\frac{n}{2}-1)^2}{n} \tag{4.8}$$

from the last section we know that $\boldsymbol{min}(d_H)$ is the weight difference between the codeword we choose and the balanced code $\mathscr{C}$: therefore, we can get:

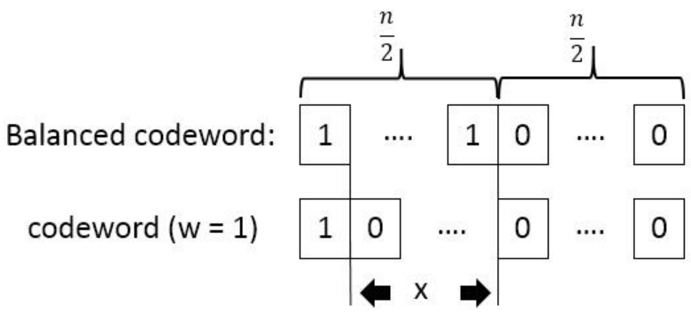

Figure 4.11: The minimum Hamming distance is also equal to the weight difference between the codeword we choose and the balanced code.

$$\begin{aligned} \boldsymbol{min}(d_{MP}) &= \frac{n}{2} - 1 - \frac{(\frac{n}{2}-1)^2}{n} \\ &= \frac{n}{4} - \frac{1}{n} \end{aligned} \tag{4.9}$$

we can see that $\boldsymbol{min}(d_{MP})$ increase with $n$ increase, we already know $n > 8$, thus

$$\boldsymbol{min}(d_{MP}) \geq \frac{9}{4} - \frac{1}{9} > 2 \tag{4.10}$$

therefore, we can always choose the codewords with weight $w = 1$ when $n > 8$. From the Theorem 1 we know, in the offset-only Pearson detection, when the codeword length $n > 8$, we can always combine more codewords with balanced code to form new Pearson code with minimum modified Pearson distance 2.

# 5
# Minimum Pearson distance detection

In this chapter, we will investigate the minimum Pearson distance detection for different Pearson codes, different from offset only case, in minimum Pearson distance detection, the channel contains both offset and gain mismatch. The object is to make comparison between different Pearson codes and investigate which possible factor of a Pearson code may affect its error performance in the minimum Pearson distance detection.

## 5.1. Theoretical upperbound of word error rate

Recall the channel models we introduced in chapter 2, here we assume the channel is ideal and matched, only contains additive Gaussian noise, and the signal vector $\boldsymbol{x}$ of a Pearson code is sent , where all '0' and all '1' codewords are excluded. So the the receiver use the Pearson distance (2.5) to compute the distance between the received vector $\boldsymbol{r} = \boldsymbol{x} + \boldsymbol{v}$ and all the other codewords in the code book, Thus in this case (2.14) is:

$$\delta_P(\boldsymbol{r}, \hat{\boldsymbol{x}}) < \delta_P(\boldsymbol{r}, \boldsymbol{x}) \tag{5.1}$$

Combine (2.5) and (2.6) we can get:

$$\delta_P(\boldsymbol{r}, \hat{\boldsymbol{x}}) = 1 - \frac{\sum_{i=1}^{n}(r_i - \overline{r})(\hat{x}_i - \overline{\hat{x}})}{\sigma_r \sigma_{\hat{x}}}$$

$$= 1 - \frac{\sum_{i=1}^{n} r_i(\hat{x}_i - \overline{\hat{x}})}{\sigma_r \sigma_{\hat{x}}} + \overline{r}\frac{\sum_{i=1}^{n}(\hat{x}_i - \overline{\hat{x}})}{\sigma_r \sigma_{\hat{x}}}$$

remove the factor which is independent of $r_i$, we can write down the equivalents of the Pearson distance measure:

$$\delta_P(\boldsymbol{r}, \hat{\boldsymbol{x}}) \equiv -\frac{\sum_{i=1}^{n} r_i(\hat{x}_i - \overline{\hat{x}})}{\sigma_{\hat{x}}}$$

Now apply it in (5.1), we can get:

$$\sum_{i=1}^{n}(x_i + v_i)(a_i - \hat{a}_i) < 0 \tag{5.2}$$

where

$$a_i = \frac{x_i - \overline{x}}{\sigma_x}$$

and

$$\hat{a}_i = \frac{\hat{x}_i - \overline{\hat{x}}}{\sigma_{\hat{x}}}$$

Because the left side of (5.2) is a stochastic variable with distribution $\mathcal{N}(\alpha_P, \beta_P\sigma^2)$, where

$$\alpha_P = \sum_{i=1}^{n} x_i(a_i - \hat{a}_i)$$

and

$$\beta_P = \sum_{i=1}^{n} (a_i - \hat{a}_i)^2$$

According to the transformation[1], we can define the minimum Pearson squared distance(or minimum squared distance for shorthand representation) between any pair of codewords in $S$, $d^2_{min,P}$, by,

$$d^2_{min,P} = \frac{4\alpha_P^2}{\beta_P} = min_{\boldsymbol{x},\hat{\boldsymbol{x}}\in S;\boldsymbol{x}\neq\hat{\boldsymbol{x}}} 2\sigma_x^2(1-\rho_{\boldsymbol{x},\hat{\boldsymbol{x}}}) \tag{5.3}$$

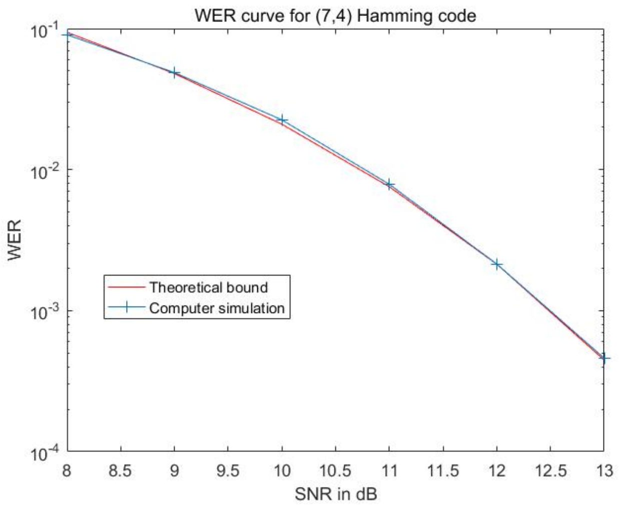

Figure 5.1: Word error rate(WER) bound and simulation of Pearson detector as a function of signal-to-noise ratio(SNR) for modified Hamming code (7,4,3) in the ideal, matched case only with additive white Gaussian noise. The error performance was computed by upper bound (5.5) and computer simulations(lines with markers).

Because the detector decides by the minimum distance between every codeword and received signal vector, Then we can transform the original word error rate (WER) upper bounded inequality:

$$WER < \frac{1}{|S|}\sum_{\boldsymbol{x}\in S}\sum_{\boldsymbol{x}\neq\hat{\boldsymbol{x}}} Q\left(\frac{d_P(\boldsymbol{x},\hat{\boldsymbol{x}})}{2\sigma}\right) \tag{5.4}$$

to

$$WER < N_P Q\left(\frac{d_{min,P}}{2\sigma}\right) \tag{5.5}$$

Here $N_P$ is the average number of nearest neighbors at minimum distance $d_{min,P}$ for every codeword. The Figure 5.1 shows the word error rate computed using upper bound (5.5) and detection simulation for modified Hamming code$(7,4,3)$, which is the Hamming code (7,4,3) with all zero and all one codewords excluded. In this case, the average number of nearest neighbors $N_p$ is 6, and the minimum squared distance $d_{min,P}$ is 2.86, we can see that the simulation results matches very well with the upper bound in high signal to noise ratio.

## 5.2. THE ADVANTAGE OF PEARSON DISTANCE DETECTION

Here we compare the minimum Pearson distance detection with the minimum Euclidean distance detection and minimum Modified Pearson distance detection in the different condition with the same code, and because Pearson detection is dedicated for Pearson code, this code should be Pearson code, here we use the modified Hamming code(7,4,3), which comes from the Hamming code(7,4,3) with all zero and all one codewords excluded.

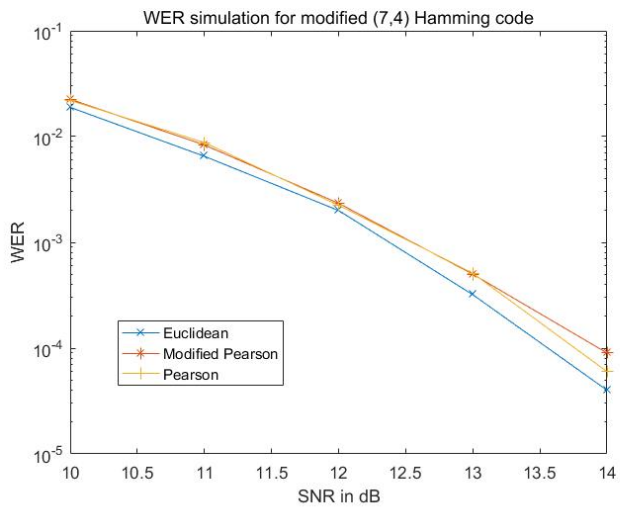

Figure 5.2: Word error rate(WER) computer simulation of Euclidean detector, modified Pearson detector, and Pearson detector for modified Hamming code $(7,4,3)$ in the ideal, matched case only with additive white Gaussian noise.

Figure 5.2 shows the computer simulation comparison of Pearson detector and Euclidean detector in the ideal matched case, results shows that the Euclidean detector outperforms the Pearson detector, because its the minimum Euclidean squared distance(3) is larger than minimum Pearson squared distance(2.86) with same average number of nearest neighbors. While the performance of minimum Modified Pearson distance detection and minimum Pearson distance detection is almost the same because their minimum squared distance are same(2.86).

Figure 5.3 shows the computer simulation comparison of Pearson detector and Euclidean detector with only offset mismatch $b = 0.2$, results shows that the performance of traditional Euclidean detector drops dramatically, while the performance of modified Pearson detector and Pearson detector almost stay the same as that in Figure 5.2. While Figure 5.4 shows the same computer simulation comparison but with both offset and gain mismatch: $b = 0.2$; $a = 2.5$. it shows that the Euclidean detector performs even worse the results in Figure 5.3, however, the other two detector still stay the same, however, we notice that the modified Pearson detector performs exactly the same as Pearson detector when there is gain mismatch, this is because the modified Hamming code $(7,4,3)$ is a special code where all the codewords have the same variance. Assume codeword $\boldsymbol{a}$ and

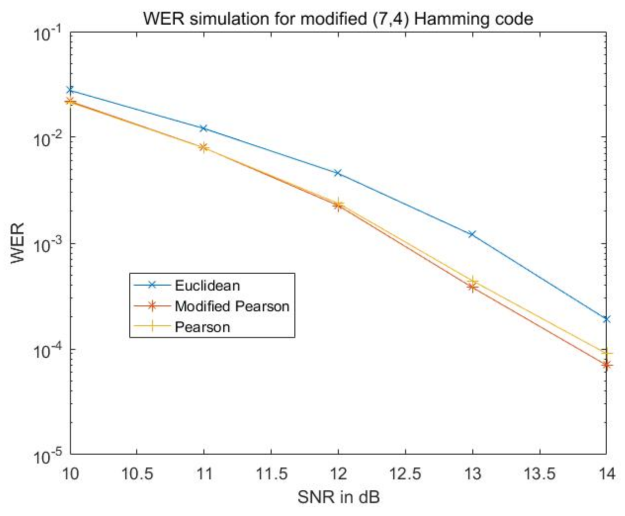

Figure 5.3: Word error rate(WER) computer simulation of Euclidean detector, modified Pearson detector, and Pearson detector for modified Hamming code $(7,4,3)$ with additive white Gaussian noise and channel mismatch: only offset mismatch $b = 0.2$.

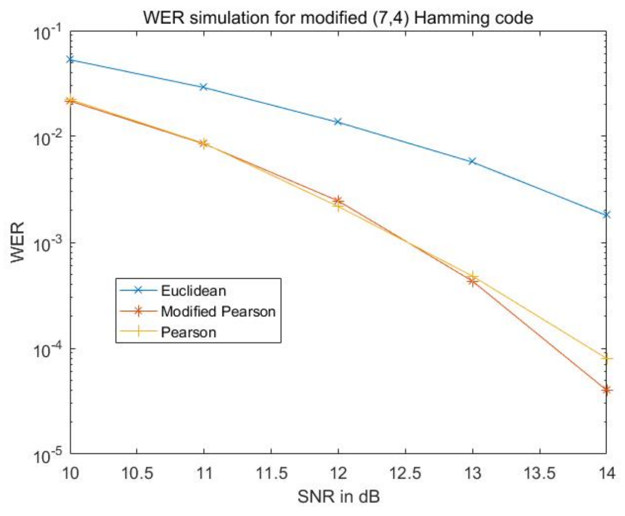

Figure 5.4: Word error rate(WER) computer simulation of Euclidean detector, modified Pearson detector, and Pearson detector for modified Hamming code $(7,4,3)$ with additive white Gaussian noise and channel offset mismatch: both offset mismatch $b = 0.2$ and gain mismatch $a = 2.5$.

codeword $\boldsymbol{b}$ are from the code book of modified Hamming code $(7,4,3)$. So from (2.7) we know:

$$\sigma_a^2 = \sum_{i=1}^{n}(a_i - \overline{a})^2 = \sigma_b^2 = \sum_{i=1}^{n}(b_i - \overline{b})^2 = \sigma_0^2$$

therefore, the squared distance (5.3) in Pearson detection, by definition, is:

$$\begin{aligned} d_P(\boldsymbol{a},\boldsymbol{b})^2 &= 2\sigma_0^2\left(1 - \frac{\sum_{i=1}^{n}(a_i-\overline{a})(b_i-\overline{b})}{\sigma_{\boldsymbol{a}}\sigma_{\boldsymbol{b}}}\right) \\ &= 2\sigma_0^2 - 2\sum_{i=1}^{n}(a_i-\overline{a})(b_i-\overline{b}) \\ &= \sum_{i=1}^{n}(a_i-\overline{a})^2 - 2\sum_{i=1}^{n}(a_i-\overline{a})(b_i-\overline{b}) + \sum_{i=1}^{n}(b_i-\overline{b})^2 \\ &= \sum_{i=1}^{n}\left[(a_i-\overline{a})-(b_i-\overline{b})\right]^2 \\ &= \sum_{i=1}^{n}\left[(a_i-b_i)-(\overline{a}-\overline{b})\right]^2 \\ &= d_{MP}(\boldsymbol{a},\boldsymbol{b})^2 \end{aligned} \tag{5.6}$$

We can see that the squared distance $d_P(\boldsymbol{a},\boldsymbol{b})^2$ in Pearson detection is equivalent to the squared distance $d_{MP}(\boldsymbol{a},\boldsymbol{b})^2$ in Modified Pearson detection for modified Hamming code $(7,4,3)$. Therefore we can conclude that for the Pearson codes which all codewords have the same variance, for example, some constant weight codes like balanced codes, we can use Modified Pearson distance detector instead of Pearson distance detector when facing the channel with both offset and gain mismatch. If the Pearson codes have different variances, here we choose the single parity check T-constrained

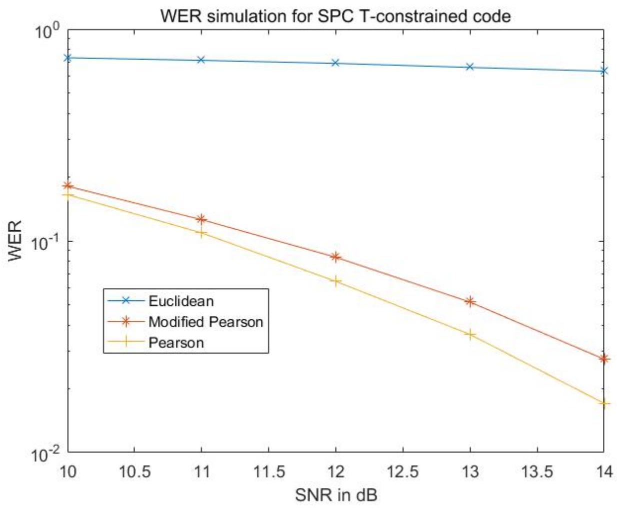

Figure 5.5: Word error rate(WER) computer simulation of Euclidean detector, modified Pearson detector, and Pearson detector for single parity check T-constrained code where $n = 5$, $T = 2.$, with additive white Gaussian noise and channel offset mismatch: both offset mismatch $b = 0.2$ and gain mismatch $a = 2.5$.

code where $n = 5$, $T = 2$. The results in Figure 5.5 shows that in this case, the Pearson detector has the advantage of resistance for both gain and offset mismatch, while the modified Pearson detector can only offer the resistance against the offset mismatch.

## 5.3. PERFORMANCE COMPARISON WITH DIFFERENT PEARSON CODES

In this section, we continue to investigate the performance of different kinds of Pearson codes, here we choose 5 codes: modified Hamming code(7,4,3), modified Hamming code(15,11,3), T-constrained code($n = 7, T = 2$), single parity check T-constrained code($n = 7, T = 2$), and modified Reed Muller code RM(1,3), all of these codes are Pearson codes with all one and all zero codewords excluded.

| | Code rate | $\boldsymbol{min}(d_H)$ | $\boldsymbol{min}(d_P^2)$ | $N_P$ | Performance ranking |
|---|---|---|---|---|---|
| Modified Hamming(7,4) | 0.544 | 3 | 2.86 | 6 | 2 |
| Modified Hamming(15,11) | 0.733 | 3 | 1.86 | 0.55 | 3 |
| T-constrained | 0.997 | 1 | 0.61 | 0.67 | 5 |
| SPC T-constrained | 0.854 | 2 | 0.91 | 1.67 | 4 |
| Modified Reed-Muller(1,3) | 0.476 | 4 | 4 | 12 | 1 |

Table 5.1: Comparison of 5 different Pearson codes, the factors are: code rate; minimum Hamming distance $\boldsymbol{min}(d_H)$; minimum Pearson squared distance $\boldsymbol{min}(d_P^2)$; average number of nearest neighbors at $\boldsymbol{min}(d_P^2)$: $N_P$; performance ranking.

The table 5.1 shows the information of theses five Pearson codes in minimum Pearson distance detection, and the Figure 5.6 shows the simulation results of these codes in Pearson detection. Similar to the modified Pearson detection, the code with smaller code rate always have better performance, and for the Modified Reed Muller code, the Hamming distance is also equal to the Pearson squared distance, although the modified Hamming code(7,4,3) have the same Hamming distance with Hamming code(15,11,3), while its shorter code length means less symbol error probability and its minimum Pearson squared distance is larger than the Hamming code(15,11,3), so it has better performance. We also notice that for those Pearson codes with the same code length, the minimum Hamming distance seems also have a positive correlation with the minimum Pearson distance, in the next chapter, we will further investigate the relations between Pearson distance and Hamming distance, and what factors should be considered in the Pearson code design for minimum Pearson distance detection.

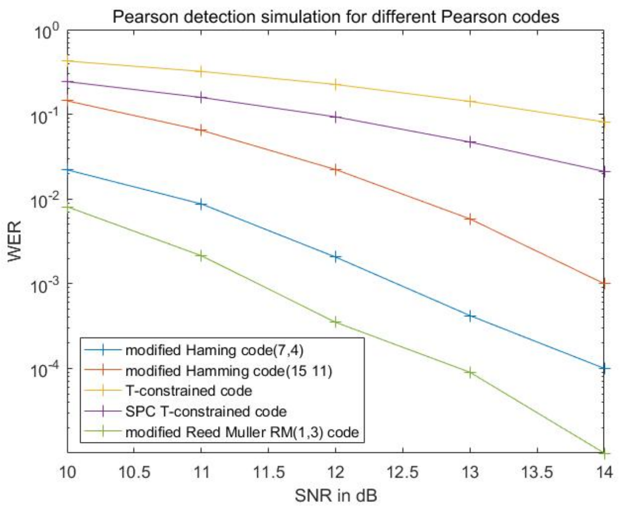

Figure 5.6: Word error rate(WER) performance simulation of the five Pearson codes, the channel is only with additive white Gaussian noise and there is no channel mismatch.

## 5.4. Pearson detection for ternary Pearson codes

In this section, we will investigate the minimum Pearson distance detection for Pearson codes with $q = 3$ and the code length $n = 4$, where the channel contains both offset and gain mismatch. The codewords are chosen over the alphabet $Q = \{0,1, 2\}$ and the size of a full code book is $|S| = q^n = 3^4 = 81$. According to the definition of Pearson code, the largest size ternary Pearson code is the T-constrained code($T = 2$), where symbol '0' and '2' appear at least once in each codeword, so there are only 6 kinds of symbol composition left: {0012}, {0112}, {0122}, {0002}, {0022}, and {0222}. And the code size is $|S_{2,t}| = 50$. While for binary Pearson codes($q = 2$), the Pearson code with largest size is also T-constrained code($T = 2$), and the code size is $|S_{2,b}| = 14$. According to the Matlab computation, the minimum Pearson squared distance of $S_{2,t}$ is 0.53, and average number of nearest neighbors is 0.48; the minimum Pearson squared distance of $S_{2,b}$ is 0.63, and average number of nearest neighbors is 1.71, therefore, $S_{2,b}$ could perform better than $S_{2,t}$ at high SNR because of larger minimum Pearson squared distance. The Figure 5.7 shows the simulation comparison of binary and ternary Pearson code at high SNR, we can see that it matches our computation results very well.

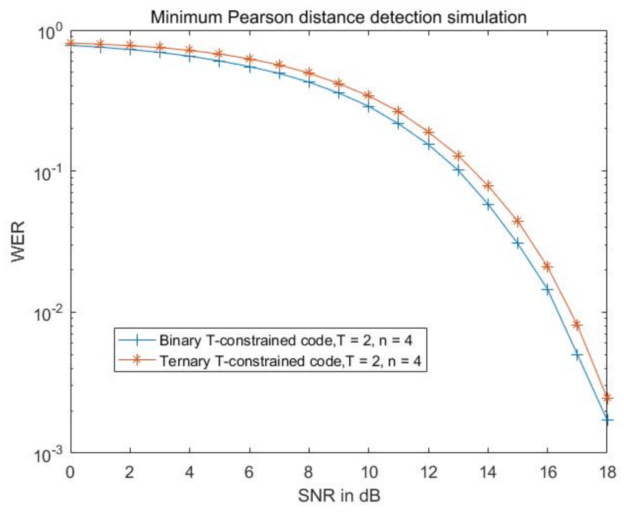

Figure 5.7: Word error rate(WER) performance simulation of the largest binary Pearson code $S_{2,b}$ and largest ternary Pearson code $S_{2,t}$ with same code length $n = 4$, the channel is only with additive white Gaussian noise and there is no channel mismatch.

Recall the definition of single parity check codes, then we divide $S_{2,t}$ into 3 single parity check codes with different symbol compositions as follows:

| $SPC_{2a}$ mod 3 = 0 | $SPC_{2b}$ mod 3 = 1 | $SPC_{2c}$ mod 3 = 2 |
|---|---|---|
| 0 0 1 2 | 0 1 1 2 | 0 1 2 2 |
| 0 2 2 2 | 0 0 2 2 | 0 0 0 2 |

Table 5.2: The symbol compositions of different SPC T-constrained codes with $T = 2$ and $n = 4$

The Table 5.3 shows the information of all the three ternary Pearson codes above. We can see that after adding a single parity check, the minimum Pearson squared distance of all three codes increases, unlike the results of modified Pearson distance detection, here the minimum Pearson squared distance of $SPC_{2a}$ and $SPC_{2c}$ are the same, while that of $SPC_{2b}$ is smaller but with larger code size. We also found that for $SPC_{2a}$ and $SPC_{2c}$, all the nearest codeword pairs are located only in the same symbol composition set of codewords: {0012} and {0122}; while for $SPC_{2b}$ the nearest neighbor are out of the same symbol composition set of codewords, for example, the nearest neighbors of ‘0112’ are ‘0022’ and ‘0202’.

| | Size | Minimum Pearson squared distance | Average number of nearest neighbors |
|---|---|---|---|
| $S_{2,t}$ | 50 | 0.53 | 0.48 |
| $SPC_{2a}$ | 16 | 2 | 2.25 |
| $SPC_{2b}$ | 18 | 1.17 | 1.33 |
| $SPC_{2c}$ | 16 | 2 | 2.25 |

Table 5.3: Comparison of different Pearson codes($q = 3$)

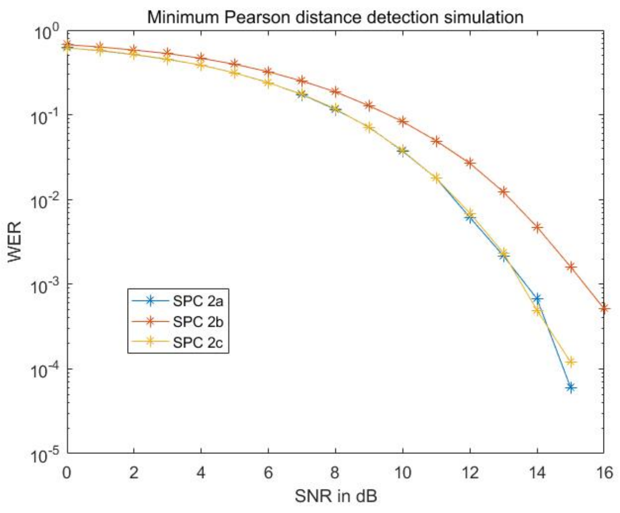

Figure 5.8: Word error rate(WER) performance simulation of ternary single parity check Pearson code $SPC_{3a}$, $SPC_{3b}$, and $SPC_{3c}$. The channel is only with additive white Gaussian noise and there is no channel mismatch.

The performance of these 3 single parity check codes are shown in the Figure 5.8, it matches the computation results of Table 5.3 very well, the $SPC_{3b}$ performs worse than other two codes because its smaller Minimum Pearson squared distance. Compared with modified Pearson distance detection, the relations between codewords in Pearson distance detection are more complicated.

# 6
# Pearson distance vs Hamming distance

In the chapter 4 we have already discuss the relations between modified Pearson squared distance and Hamming distance, but if the channel contains not only offset but also gain mismatch, we should apply the minimum Pearson distance detection. Because the Pearson squared distance $d_P^2$ is not symmetric in each codeword pair, here we study the symmetric part of the squared distance, which is the Pearson distance $\delta_P$, for the code design. In this chapter, we will discuss the relations between Pearson distance and Hamming distance for binary Pearson code, and shed some light on how to utilize hamming distance to design a binary Pearson code for the channel with both offset and gain mismatch.

The Figures 6.1, 6.2, 6.3, 6.4 below illustrate the rough trend of the relation between Pearson distance and Hamming distance in some modified Hamming codes and modified BCH codes, where all zero and one code sequences are excluded from the original Hamming codes and BCH codes, to make it more comparable, the Hamming distance is normalized: it is divided by the codeword length. We notice that there is always a positive correlation between the Hamming distance and Pearson distance, and this is more clear when the codes contain less number of different nonzero Hamming distances, but from the codes have more different Hamming distances, it shows that there are some other factors that have impact on the relations between Hamming distance and Pearson distance so that they can change the slope of the positive correlation or lead to some offsets, which makes the relations much more complicated. So what are these other factors? How many are they? In this chapter we will investigate these questions.

## 6.1. The Pearson distance vs Hamming distance

Consider a full set binary communication code book with all one code sequence and all zero code sequence excluded, S, of chosen binary codewords with length $n$ over the binary alphabet $Q = \{0,1\}$. Suppose any two codewords : $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)$ and $\boldsymbol{b} = (b_1, b_2, \ldots, b_n)$ of the code book, to make it more simple to compare, we can arrange them like the Figure 6.5(can represent all the codewords in the same constant composition subset):

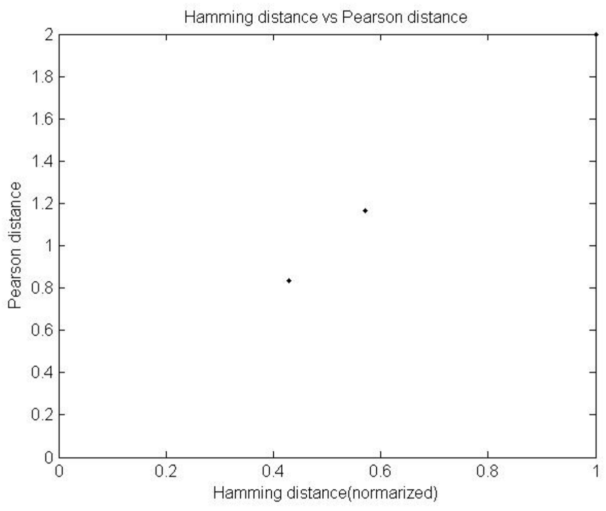

Figure 6.1: Hamming distance *vs* Pearson distance in modified Hamming code, n = 7 and k = 4

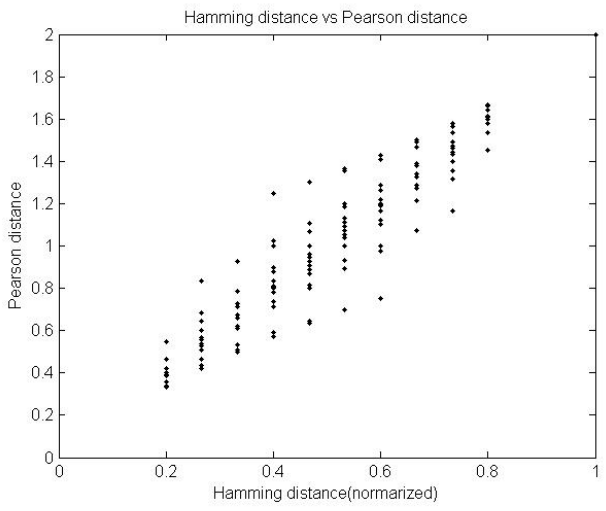

Figure 6.2: Hamming distance *vs* Pearson distance in modified Hamming code, n = 15 and k = 11

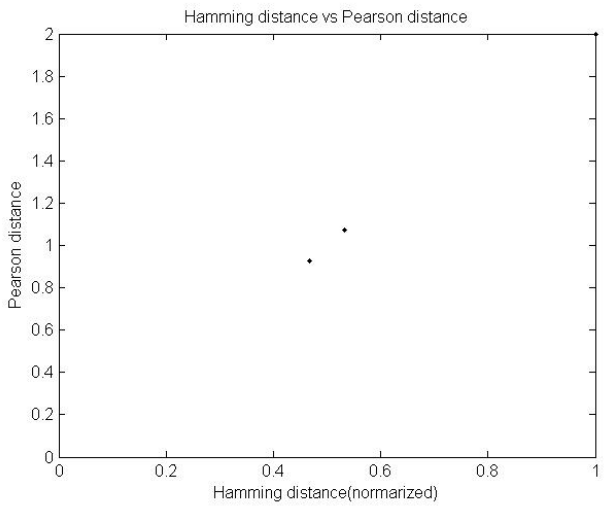

Figure 6.3: Hamming distance *vs* Pearson distance in modified BCH code, n = 15 and k = 5

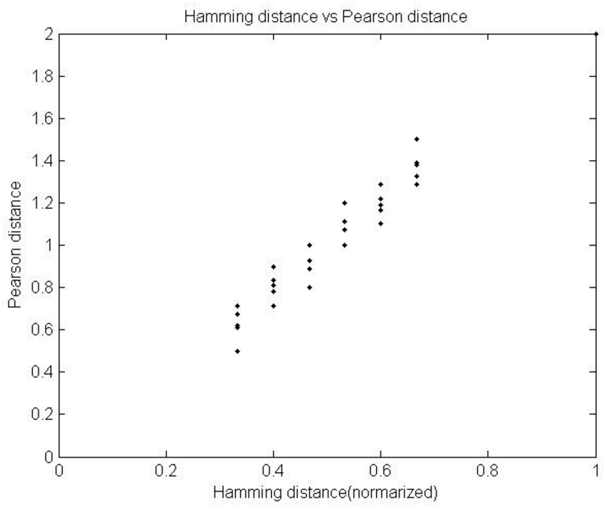

Figure 6.4: Hamming distance *vs* Pearson distance in modified BCH code, n = 15 and k = 7

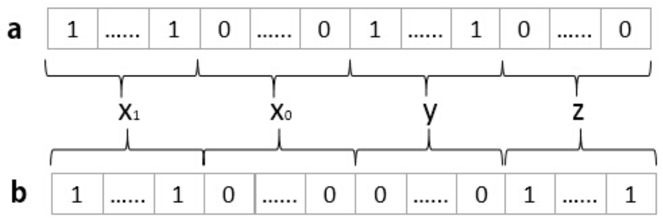

Figure 6.5: Comparison of any codeword pair by rearrange their symbols to four parts: ‘1’ to ‘1’; ‘0’ to ‘0’; ‘1’ to ‘0’; ‘0’ to ‘1’.

According to the definition of Pearson coefficient:

$$\rho_{\boldsymbol{a},\boldsymbol{b}} = \frac{\sum_{i=1}^{n}(a_i-\overline{a})(b_i-\overline{b})}{\sigma_a\sigma_b} \tag{6.1}$$

where

$$\overline{a} = \frac{1}{n}\sum_{i=1}^{n} a_i$$

$$\sigma_a^2 = \sum_{i=1}^{n}(a_i-\overline{a})^2$$

So we have:

$$\rho_{\boldsymbol{a},\boldsymbol{b}} = \frac{n\overline{a}\overline{b}-(x_1+z)\overline{a}-(x_1+y)\overline{b}+x_1}{\sqrt{[x_1+y-2(x_1+y)\overline{a}+n\overline{a}^2][x_1+z-2(x_1+z)\overline{b}+n\overline{b}^2]}} \tag{6.2}$$

Because

$$\begin{cases} x_1+y=n\overline{a} \\ x_1+z=n\overline{b} \\ 2x_1+d_H(\boldsymbol{a},\boldsymbol{b})=n(\overline{a}+\overline{b}) \end{cases} \tag{6.3}$$

where the $d_H(\boldsymbol{a},\boldsymbol{b})$ is the Hamming distance between $\boldsymbol{a}$ and $\boldsymbol{b}$,combine (6.2) and (6.3) we get the expression of Pearson distance $\delta_P(\boldsymbol{a},\boldsymbol{b})$

$$\delta_P(\boldsymbol{a},\boldsymbol{b}) = 1-\rho_{\boldsymbol{a},\boldsymbol{b}} = 1-\frac{\overline{a}+\overline{b}-2\overline{ab}-\dfrac{d_H(\boldsymbol{a},\boldsymbol{b})}{n}}{2\sqrt{\overline{a}\overline{b}[1-(\overline{a}+\overline{b})+\overline{ab}]}} \tag{6.4}$$

Let $m$ denote the Hamming weight difference between $\boldsymbol{a}$ and $\boldsymbol{b}$: $m = |y-z|$, then we can derive

$$\begin{cases} \overline{a}+\overline{b} = \dfrac{x_1-x_0+n}{n} \\ \overline{ab} = \dfrac{(x_1-x_0+n)^2-m^2}{4n^2} \end{cases} \tag{6.5}$$

From (6.4) and (6.5) we know there are three factors that determine the Pearson distance between codewords $\boldsymbol{a}$ and $\boldsymbol{b}$: the Hamming distance $\boldsymbol{d_H(a,b)}$, the weight difference $\boldsymbol{m}$, and the difference between the number of '$0'$ and number of '$1'$ in the same position: $\boldsymbol{x_1-x_0}$. The Figure 6.6 shows the relations between Pearson distance and Hamming distance for a full set of code book when $n$ is 7.

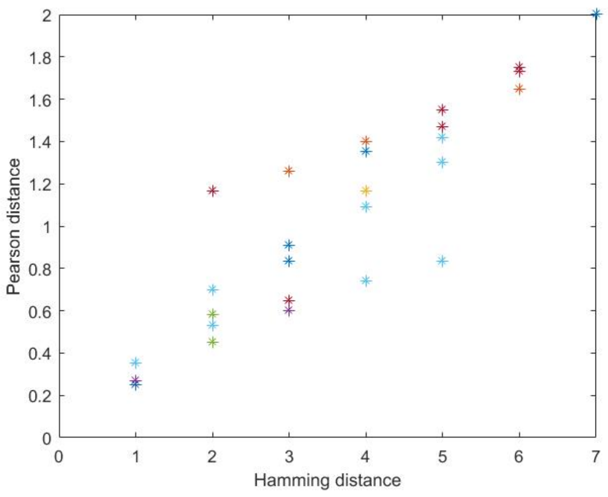

Figure 6.6: The Pearson distance vs Hamming distance for a full set of code book with length $n = 7$, all zero sequence and all one sequence are excluded.

Therefore, similar with the relations between modified Pearson squared distance and Hamming distance, for two codewords from the same weight pillar or two known weight pillars, their Pearson distance is proportional to their Hamming distance, and smaller Hamming weight difference also means better Pearson distance. However, unlike the modified Pearson distance, the Pearson distance varies for different codeword pairs with the same Hamming distance and weight difference in different position of the weight distribution, from (6.4) we also can write the Pearson coefficient $\rho_{\boldsymbol{a},\boldsymbol{b}}$ as:

$$\rho_{\boldsymbol{a},\boldsymbol{b}} = \frac{\overline{a}(1-\overline{b})+\overline{b}(1-\overline{a})-\dfrac{d_H(\boldsymbol{a},\boldsymbol{b})}{n}}{2\sqrt{\overline{a}(1-\overline{b})\overline{b}(1-\overline{a})}} \tag{6.6}$$

also we know

$$\begin{cases} \overline{a}(1-\overline{b}) = \dfrac{(n+m)^2-(x_1-x_0)^2}{4n^2} \\ \overline{b}(1-\overline{a}) = \dfrac{(n-m)^2-(x_1-x_0)^2}{4n^2} \end{cases} \tag{6.7}$$

plug (6.7) into (6.6) and substitute $t$ for $(x_1 - x_0)^2$, and we know $t \in \left[0, (n - d_H(\boldsymbol{a}, \boldsymbol{b}))^2\right]$, then we can get the relations between $t$ and Pearson coefficient:

$$\rho_{\boldsymbol{a},\boldsymbol{b}} = \frac{n^2 + m^2 - 2nd_H(\boldsymbol{a}, \boldsymbol{b}) - t}{\sqrt{\left[(n+m)^2 - t\right]\left[(n-m)^2 - t\right]}} \tag{6.8}$$

differentiate (6.8) with respect to $t$, we have

$$\begin{aligned}
\frac{d\rho_{\boldsymbol{a},\boldsymbol{b}}}{dt} &= \frac{-\sqrt{\left[(n+m)^2 - t\right]\left[(n-m)^2 - t\right]} - (n^2 + m^2 - 2nd_H(\boldsymbol{a}, \boldsymbol{b}) - t)\dfrac{-\left[(n-m)^2 - t\right] - \left[(n+m)^2 - t\right]}{2\sqrt{\left[(n+m)^2 - t\right]\left[(n-m)^2 - t\right]}}}{\left[(n+m)^2 - t\right]\left[(n-m)^2 - t\right]} \\
&= \frac{-4\left[(n+m)^2 - t\right]\left[(n-m)^2 - t\right] + 2(n^2 + m^2 - 2nd_H(\boldsymbol{a}, \boldsymbol{b}) - t)\left(\left[(n-m)^2 - t\right] + \left[(n+m)^2 - t\right]\right)}{4\left[(n+m)^2 - t\right]\left[(n-m)^2 - t\right]\sqrt{\left[(n+m)^2 - t\right]\left[(n-m)^2 - t\right]}} \\
&= \frac{\left(\left[(n-m)^2 - t\right] - \left[(n+m)^2 - t\right]\right)^2 - 8nd_H(\boldsymbol{a}, \boldsymbol{b})(n^2 + m^2 - t)}{4\left(\left[(n+m)^2 - t\right]\left[(n-m)^2 - t\right]\right)^{\frac{3}{2}}} \\
&= \frac{4m^2n^2 - 2nd_H(\boldsymbol{a}, \boldsymbol{b})(n^2 + m^2 - t)}{\left(\left[(n+m)^2 - t\right]\left[(n-m)^2 - t\right]\right)^{\frac{3}{2}}}
\end{aligned} \tag{6.9}$$

we know the denominator of (6.9) is positive, and from (6.7) we know $t < (n - m)^2$, so the numerator

$$\begin{aligned}
4m^2n^2 - 2nd_H(\boldsymbol{a}, \boldsymbol{b})(n^2 + m^2 - t) &< 4m^2n^2 - 2nd_H(\boldsymbol{a}, \boldsymbol{b})\left[n^2 + m^2 - (n-m)^2\right] \\
&< 4mn^2\left(m - d_H(\boldsymbol{a}, \boldsymbol{b})\right) < 0
\end{aligned} \tag{6.10}$$

therefore, for any $t$

$$\frac{d\rho_{\boldsymbol{a},\boldsymbol{b}}}{dt} < 0 \tag{6.11}$$

Thus we can conclude that if Hamming distance $\boldsymbol{d_H(a,b)}$ and weight difference $\boldsymbol{m}$ are both constant,the Pearson distance $\boldsymbol{\delta_P(a,b)}$ and $\boldsymbol{|x_1 - x_0|}$ are positively correlated. For example, when $n$ is 13, Hamming distance is 7, and weight difference is 3, the comparison of Pearson distance of all possible codeoword pairs is shown in the table 6.1, as we can see, codeword pair with $x_1 = x_0$ has minimum Pearson distance. Let $w_a$ denote the Hamming weight of $\boldsymbol{a}$, and $w_b$ denote the Hamming weight of $\boldsymbol{b}$, we have

$$\begin{cases} x_1 = x_0 = \dfrac{n - d_H(\boldsymbol{a}, \boldsymbol{b})}{2} \\ w_a = x_1 + y \\ w_b = x_1 + z \end{cases} \tag{6.12}$$

then we get

$$\begin{cases} w_a = \dfrac{n+m}{2} \\ \\ w_b = \dfrac{n-m}{2} \end{cases} \tag{6.13}$$

(6.13) shows us that despite any values of Hamming distance, codeword pair with $w_a$ and $w_b$ which is symmetric by $\frac{n}{2}$ always have the minimum Pearson distance compared with any other codeword pairs with same weight difference and Hamming distance, which means for Pearson code designer, if the codewords with length $n$ from weight pillar whose weight is $w$ are chosen, it is better not to choose the codewords from the weight pillar with weight $n - w$.

| codeword pair | $x_1 - x_0$ | Pearson distance | codeword pair | $x_1 - x_0$ | Pearson distance |
|---|---|---|---|---|---|
| 0000001111100<br>0000000000011 | -6 | 1.3371 | 1111111111100<br>1111110000011 | 6 | 1.3371 |
| 1000001111100<br>1000000000011 | -4 | 1.1409 | 1111101111100<br>1111100000011 | 4 | 1.1409 |
| 1100001111100<br>1100000000011 | -2 | 1.0514 | 1111001111100<br>1111000000011 | 2 | 1.0514 |
| 1110001111100<br>1110000000011 | 0 | 1.0250 | | | |

Table 6.1: Comparison of Pearson distance for different codeword pairs

## 6.2. THE SQUARED DISTANCE FOR CONSTANT WEIGHT CODE

As we know the variance is same for all codewords of the same weight pillar, so all codewords of a constant weight code have the same variance. In this section we will discuss the squared distance $d_p^2$ in (5.3) instead of Pearson distance $\delta_P$ for constant weight Pearson code design in minimum Pearson distance detection. From (6.8) we know, for constant weight code, $m = 0$, then we have

$$\rho_{\boldsymbol{a},\boldsymbol{b}} = \frac{n^2 - 2nd_H(\boldsymbol{a},\boldsymbol{b}) - t}{n^2 - t} = 1 - \frac{2nd_H(\boldsymbol{a},\boldsymbol{b})}{n^2 - t} \tag{6.14}$$

and the variance is

$$\sigma_a^2 = \sigma_b^2 = n\overline{a}(1-\overline{a}) = \frac{n^2 - t}{4n} \tag{6.15}$$

we know in the Minimum Pearson Distance detection consider the channel with Gaussian noise, the squared distance determines the performance of Pearson code.

$$d_P(\boldsymbol{a},\boldsymbol{b})^2 = 2\sigma_a^2\delta_P(\boldsymbol{a},\boldsymbol{b}) = 2\sigma_a^2(1-\rho_{\boldsymbol{a},\boldsymbol{b}}) = 2\sigma_b^2(1-\rho_{\boldsymbol{a},\boldsymbol{b}}) \tag{6.16}$$

Therefore, the squared distance is symmetric for the codeword pair of constant weight code. Plug (6.14) and (6.15) into (6.16) we get

$$d_P(\boldsymbol{a},\boldsymbol{b})^2 = d_H(\boldsymbol{a},\boldsymbol{b}) \tag{6.17}$$

thus we know the Pearson squared distance is equivalent to Hamming distance for constant weight code design.

## 6.3. MINIMUM PEARSON DISTANCE BETWEEN WEIGHT PILLARS

We already know that for the codewords in the same weight pillar, improving minimum Pearson distance is the same as improving minimum Hamming distance, but how to pick different pillar combination for a Pearson code design still needs to be discussed, in this section, we will investigate the minimum Pearson distance between every two weight pillars in the weight distribution of a full set of code book.

From the Figure 6.5 we know, for the codeword pair from two different weight pillars we have:

$$\begin{cases} x_1 + y = w_a \\ x_0 + y = n - w_b \end{cases} \tag{6.18}$$

where $w_a$ and $w_b$ are the weight of $\boldsymbol{a}$ and $\boldsymbol{b}$, which are both constant, then we can get:

$$x_1 - x_0 = w_a + w_b - n \tag{6.19}$$

which means $|x_1 - x_0|$ is constant, therefore, the Pearson distance between the codewords from two different weight pillars is also only determined by their Hamming distance, previous section told us Pearson distance is proportional to their Hamming distance, so the codeword pair which can be rearranged like Figure 6.7 have minimum Hamming distance and Pearson distance:

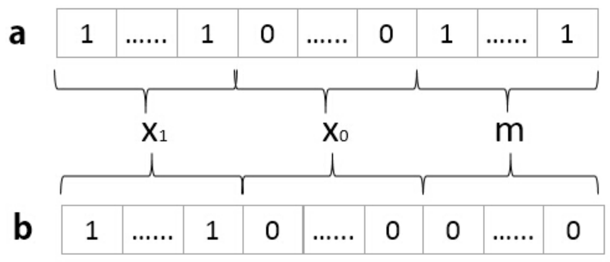

Figure 6.7: Comparison of any codeword pair from two selected weight pillars by rearranging their symbols

Figure 6.7 shows that $d_H(\boldsymbol{a},\boldsymbol{b})$ is equal to $m$. From (6.8) we get:

$$\begin{aligned}
\boldsymbol{min}\big(\delta_P(\boldsymbol{a},\boldsymbol{b})\big) &= 1-\rho_{\boldsymbol{a},\boldsymbol{b}}\Big|_{d_H(\boldsymbol{a},\boldsymbol{b})=m} \\
&= 1-\frac{n^2+m^2-2mn-t}{\sqrt{\big[(n+m)^2-t\big]\big[(n-m)^2-t\big]}} \\
&= 1-\sqrt{\frac{(n-m)^2-t}{(n+m)^2-t}} \\
&= 1-\sqrt{1-\frac{4mn}{(n+m)^2-t}}
\end{aligned} \tag{6.20}$$

we know $t$ is the substitution of $(x_1 - x_0)^2$, so from (6.20) we can see that both $m$ and $|x_1 - x_0|$ are positively correlated with the minimum Pearson distance of two weight pillars, where $m \in [1, n-2]$ and $|x_1 - x_0| \in [0, n-m]$,which means the pillar pair with larger $m$ or $|x_1 - x_0|$ has better minimum Pearson distance, and according to (6.4), (6.5), and (6.8), Pearson distance also increases faster with Hamming distance for pillar pair which has larger $m$ or $|x_1 - x_0|$.

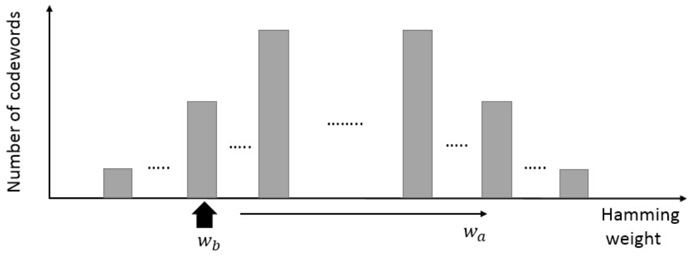

Figure 6.8: Keep $w_b$ fixed, $w_a \in [w_b + 1, n-1]$.

We note that $m$ and the maximum value of $|x_1 - x_0|$ are not independent from each other, therefore, to make it more clear, we can investigate the trend of Pearson distance when we fix the weight of one pillar in the pillar pair, $w_a$ or $w_b(w_a > w_b)$. Firstly, if we keep $w_b$ fixed, $w_a \in [w_b + 1, n-1]$, as the Figure 6.8 shows. According (6.19), we have:

$$x_1 - x_0 = 2w_b + m - n \tag{6.21}$$

where $m > 0$, plug (6.21) into (6.20), we get:

$$\begin{aligned} \boldsymbol{min}\big(\delta_P(\boldsymbol{a},\boldsymbol{b})\big) &= 1 - \sqrt{1 - \frac{4mn}{(n+m)^2 - (m-n+2w_b)^2}} \\ &= 1 - \sqrt{1 - \frac{n}{n - w_b + \dfrac{nw_b - w_b^2}{m}}} \end{aligned} \tag{6.22}$$

we can see that when m increase with $w_a$ increase, and the minimum Pearson distance between the two pillars also increase. Next, let $w_a$ be fixed, $w_b \in [1, w_a - 1]$, like the Figure 6.9 shows.

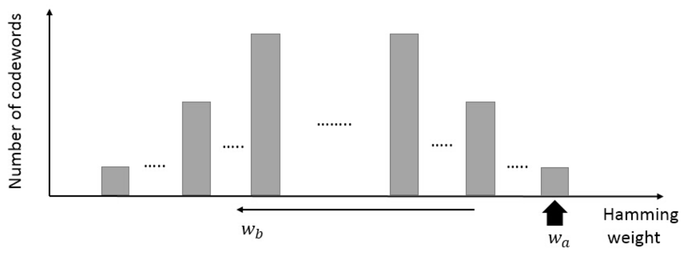

Figure 6.9: Keep $w_a$ be fixed, $w_b \in [1, w_a - 1]$.

Same according (6.19), we have:

$$x_1 - x_0 = 2w_a - m - n \tag{6.23}$$

where $m > 0$, plug (6.23) into (6.20), we get:

$$\begin{aligned} \boldsymbol{min}\big(\delta_P(\boldsymbol{a},\boldsymbol{b})\big) &= 1 - \sqrt{1 - \frac{4mn}{(n+m)^2 - (n+m-2w_a)^2}} \\ &= 1 - \sqrt{1 - \frac{n}{w_a + \dfrac{nw_a - w_a^2}{m}}} \end{aligned} \tag{6.24}$$

we can see that the minimum Pearson distance between the two pillars also increase with $m$ increase. Therefore, unlike modified Pearson distance detection, the minimum Pearson distance between the smallest weight pillar and largest weight pillar is the best among all pillar pairs for a full set Pearson binary code in Pearson detection.

For example, for a full set Pearson code with code length $n = 8$, we compare the pillar whose weight is 1 with every other pillars, their minimum Pearson distance and pillar information are shown in the Table 6.2. It shows that the larger weight pillar($w_a$) we pick from the first pillar($w_b$),

| $\{10000000\}: w_b = 1$ | minimum Pearson distance | $\lvert x_1 - x_0 \rvert$ | $m$ / $d_H$ |
|---|---|---|---|
| $\{11000000\}: w_a = 2$ | 0.3453 | 5 | 1 |
| $\{11100000\}: w_a = 3$ | 0.512 | 4 | 2 |
| $\{11110000\}: w_a = 4$ | 0.6220 | 3 | 3 |
| $\{11111000\}: w_a = 5$ | 0.7072 | 2 | 4 |
| $\{11111100\}: w_a = 6$ | 0.7818 | 1 | 5 |
| $\{11111110\}: w_a = 7$ | 0.8571 | 0 | 6 |

Table 6.2: Comparison of minimum Pearson distance for pillar $w = 1$ and other pillars in the weight distribution of a full set Pearson code with code length $n = 8$.

the better minimum Pearson distance they have, moreover, the previous section already told us both lower $|x_1 - x_0|$ and larger $m$ can lead to decreasing the Pearson distance, while from this table we note that Hamming distance has more influence on Pearson distance than other two factors.

The Table 6.3 shows the minimum and maximum Pearson distance between pillar$(w_1 = 7)$ and all the other pillars$(w_2)$ in the weight distribution of full set Pearson code with length $n = 10$. It also illustrates that larger $|w_1 - w_2|$ can lead to better minimum Pearson distance of the two weight pillars, furthermore,the minimum Pearson distance increase faster with $m$ when $w_2 > w_1$, this is because $|x_1 - x_0|$ also starts to increase. On the other hand, the maximum Pearson distance of two weight pillar is positively correlated with $|x_1 - x_0|$, and the maximum value of it is achieved in the case that $|x_1 - x_0|$ is 0 and it is 2.

| $\{1111111000\}: w_1 = 7$ | min(Pearson distance) | max(Pearson distance) | $\|x_1 - x_0\|$ | $m$ | $d_H$ |
|---|---|---|---|---|---|
| $\{1000000000\}: w_2 = 1$ | 0.7818 | 1.5092 | 2 | 6 | $[6,8]_{even}$ |
| $\{1100000000\}: w_2 = 2$ | 0.6721 | 1.7638 | 1 | 5 | $[5,9]_{odd}$ |
| $\{1110000000\}: w_2 = 3$ | 0.5714 | 2 | 0 | 4 | $[4,10]_{even}$ |
| $\{1111000000\}: w_2 = 4$ | 0.4655 | 1.8018 | 1 | 3 | $[3,9]_{odd}$ |
| $\{1111100000\}: w_2 = 5$ | 0.3453 | 1.6547 | 2 | 2 | $[2,8]_{even}$ |
| $\{1111110000\}: w_2 = 6$ | 0.1982 | 1.5345 | 3 | 1 | $[1,7]_{odd}$ |
| $\{1111111100\}: w_2 = 8$ | 0.2362 | 1.3273 | 5 | 1 | $[1,5]_{odd}$ |
| $\{1111111110\}: w_2 = 9$ | 0.4908 | 1.2182 | 6 | 2 | $[2,4]_{even}$ |

Table 6.3: Comparison of minimum and maximum Pearson distance for pillar $w = 7$ and other pillars in the weight distribution of a full set Pearson code with code length $n = 10$.

# 7 CONCLUSION AND FUTURE WORK

## 7.1. CONCLUSION OF OUR WORKS

In this thesis, we studied the Pearson codes and the Pearson-distance-based detection. In chapter 3 and chapter 5, we focus on comparing the Pearson-distance-based detection of different Pearson codes, and finding the factors that can determine the performance of Pearson code in different situations. we found that:

1) In Pearson-distance-based detection, there are two factors for measuring the performance of a Pearson code in the AWGN channel at high signal-to-noise ratios: the minimum (modified) Pearson squared distance ($d^2_{min,MP}$ or $d^2_{min,P}$) and the average number of nearest neighbors ($N_{MP}$ or $N_P$) of every codeword at the minimum squared distance, and the former factor is more important than the latter factor. The Pearson code of the same code length with better performance always has larger $d^2_{min,MP}$ and $d^2_{min,P}$, or larger average number of nearest neighbors when minimum squared distance is the same, and the code rate is relatively small.

2) For the Pearson code where all codewords have same variance, the minimum modified Pearson distance detector is the same as minimum Pearson distance detector, but for those have different variances, the minimum Pearson distance detector outperforms the modified Pearson distance detector in the channel with both gain and offset mismatch.

As we know, in the traditional Euclidean distance detection, the Hamming distance is the simple and key factor for the performance of traditional binary block codes, so in chapter 4 and chapter 6, we investigate the relations between Hamming distance and the modified Pearson squared distance or Pearson distance to reveal the key factors for Pearson code design in the Pearson-distance-based detection:

1) For the offset only channel, where minimum modified Pearson distance decoding is applied, the squared distance is symmetric for every codeword pair, we investigate the relations between the modified squared distance $d^2_{MP}$ and Hamming distance $d_H$ in chapter 4, and we found that:

$$d^2_{MP} = d_H - \frac{m^2}{n} \tag{7.1}$$

so there is two factors that can determine the modified Pearson squared distance between two codewords: their Hamming distance $d_H$ and weight difference $m$: larger Hamming distance and smaller weight difference can lead to better squared distance. So the furthest code pair in a full set of Pearson code book is located in the central weight pillar of the weight distribution, meanwhile we know, the squared distance is equivalent to Hamming distance for constant weight code. On the other hand, for the Pearson code design in the offset only channel, the better Pearson code always has larger minimum Hamming distance, or small weight range with sparse weight distribution.

2) If the channel mismatch are gain and offset, where minimum Pearson distance detection is applied, the Pearson squared distance $d^2_{MP}$ is only symmetric for codeword pair of constant weight Pearson code and it is equal to their Hamming distance. While for other Pearson codes, $d^2_{MP}$ is not symmetric, because it also depends on the sent word's symbol value variance, in this thesis we investigate the relations between the Pearson distance $\delta_P$(which is the symmetric part of the squared distance) and Hamming distance $d_H$ in chapter 6. We found that there are three factors determining the Pearson distance between two codewords: not only their Hamming distance $d_H$ and weight difference $m$, but also the difference between number of '$1'$ and number of '$0'$ in their same symbol value positions: $|x_1 - x_0|$. The Pearson distance $\delta_P$ is positively correlated with their Hamming distance $d_H$ and $|x_1 - x_0|$, but negatively correlated with the weight difference $m$, and the Hamming distance is more important than other two factors for improving the Pearson distance. Moreover, we also found that the minimum Pearson distance of two weight pillar in the weight distribution could be better if their weight difference is larger. According to these Properties, we can more easily choose good codewords and remove bad codewords to improve the error control capacity of a Pearson code.

## 7.2. FUTURE WORKS

In this thesis, we have studied the Pearson-distance-based detection, and investigated the factors that determining the (modified) Pearson distance for Pearson code design. Based on our methods and results, some possible extension could be carried out with respect to the following directions:

1) We can continue to investigate the relations between the Pearson squared distance with Hamming distance given the asymmetrical component: the variance of sent codeword, and combine with the results in our thesis for the Pearson code design.

2) Instead of study binary Pearson code, we can investigate the relations between (modified) Pearson distance and Hamming distance in Pearson-distance-based detection for q-ary Pearson codes, and find out what factors of a codeword pair we can use for q-ary Pearson code design($q \geq 2$).

3) We know this thesis is based on minimum distance decoding, It is possible to find other different new decoding scheme for some Pearson codes for the better performance.

# Bibliography

[1] K. A. S. Immink and J. H. Weber, *Minimum pearson distance detection for multilevel channels with gain and/or offset mismatch,* IEEE Trans. Inform. Theory **60**, 5966–5974 (2014).

[2] J. Irvine and D. Harle, *2.4.4 types of coding,* Data Communications and Networks , 18 (2002).

[3] R. Gray, *Source coding theory,* Kluwer Academic, Boston, Mass. (1990).

[4] C. E. Shannon, *A mathematical theory of communication,* Bell System Technical Journal , 379–423 (July 1948).

[5] J. H. Weber and K. A. S. Immink, *Pearson codes,* IEEE Trans. Inform. Theory **60**, 3993–3998 (2016).

[6] K. A. S. Immink, *Coding schemes for multi-level flash memories that are intrinsically resistant against unknown gain and/or offset using reference symbols,* Electron. Lett **50**, 20–22 (2014).

[7] A. Jiang, R. Mateescu, M. Schwartz, and J. Bruck, *Rank modulation for flash memories,* IEEE Trans. Inform. Theory **IT-55**, 2659–2673 (2006).

[8] K. A. S. Immink, *Coding schemes for multi-level channels with unknown gain and/or offset,* Proc. IEEE Int. Symp. Inf. Theory (ISIT) , 709–713 (2013).

[9] A. Gayen, *The frequency distribution of the product moment correlation coefficient in random samples of any size draw from non-normal universes,* Biometrika **38**, 219–247 (1951).

[10] K. A. S. Immink and J. H. Weber, *Hybrid minimum pearson and euclidian distance detection,* IEEE Trans.Commun **63**, 3290–3298 (2015).

[11] W. C. Huffman and V. Pless, *Fundamentals of error-correcting codes,* Cambridge (2003).

[12] C. Scott and R. Nowak, *The Q-function,* `http://cnx.org/contents/hDU5uzaA@2/The-Q-function` (2016).

[13] T. K. Moon, *Error correction coding,* New Jersey: John Wiley & Sons (2005).

[14] D. Bhattacharryya and S. Nandi, *An efficient class of sec-ded-aued codes,* 1997 International Symposium on Parallel Architectures, Algorithms and Networks **ISPAN '97**, 410–415 (1997).

[15] G. D. Forney, *Lower bounds on error probability in the presence of large intersymbol interference,* IEEE Trans. Commun. Technol. (Corresp.) **COM-20**, 76–77 (Feb. 1972).

[16] W. Chu, C. J. Colbourn, and P. Dukes, *On constant composition codes,* Discrete Appl. Math. **154**, 912–929 (Apr. 2006).

[17] R. L. Graham and J. A. Sloan, *Lower bounds for constant weight codes,* IEEE Trans. on Information Theory **26**, 37–43 (Jan. 1980).

[18] E. Agrell, A. Vardy, and K. Zeger, *Upper bounds for constant-weight codes,* IEEE Transactions on Information Theory **46**, 2373–2395 (2000).

[19] A. E. Brouwer and T. Verhoeff, *An updated table of minimum-distance bounds for binary linear codes,* IEEE Trans. Inform. Theory **39**, 662–677 (Mar. 1993).

[20] K. A. S. Immink and J. H. Weber, *Very efficient balanced codes,* IEEE Journal on Selected Areas in Communications **28**, 188–192 (2010).