Error control coding for constrained channels

A thesis submitted for the degree of

Doctor of Philosophy

in Electronic and Electrical Engineering

June 1999

Chris Matrakidis

Department of Electronic and Electrical Engineering

University College London

Acknowledgements

I wish to express my thanks to my supervisor, Professor John J. O'Reilly for his advice and guidance throughout the course of this study.

I would also like to thank all my friends and family whose support was essential for the completion of this thesis.

Abstract

Channel coding is an important consideration influencing the design of a communications system. In particular, error control coding is used to detect and/or correct errors and line coding to modify the characteristics of the transmitted signal to suit other constraints of the channel, such as restricted frequency response.

This thesis explores aspects of channel coding for such constrained channels with emphasis given to error control coding.

Specifically, the first chapter of this thesis presents a general overview of channel coding, presents the organisation of the thesis and details the main contributions.

The second chapter gives an overview of the principles of error control coding and line coding and explains a few terms that are commonly used in the remainder of the thesis.

One kind of constrained channel investigated here is the binary asymmetric error channel, where error transitions from one to zero occur with different probability than from zero to one. Error correcting codes for this channel and their properties are investigated in the third chapter.

The fourth chapter introduces disparity control coding, and proposes a new error control coding structure that satisfies disparity constraints for both binary asymmetric and symmetric error channels.

Run length limited channels are the subject of the fifth chapter. A new coding structure is proposed that offers advantages in performance over the one conventionally used for error control in such channels.

The sixth chapter introduces peak power constraints present in multi–carrier systems. Codes that can be used limit the peak to average power ratio of such systems are presented and the application of the coding structure of the fifth chapter is also discussed.

The final chapter brings the thesis to a conclusion by summarising the main results and proposing areas where further work may be fruitful.

Table of contents

List of figures

List of tables

1 Introduction

This thesis addresses a number of points relevant to coding for communications transmission and storage. In general, coding is the mapping of one data sequence into another in such a way that a desired property is achieved.

Figure 1.1 shows the main categories that coding theory deals with. These are cryptography, where the source data are modified to improve security, channel coding where redundancy is added in order to improve reliability and source coding that attempts to remove any redundancy present in the source data. This thesis deals with channel coding, which is also shown as subdivided into error control and line coding.

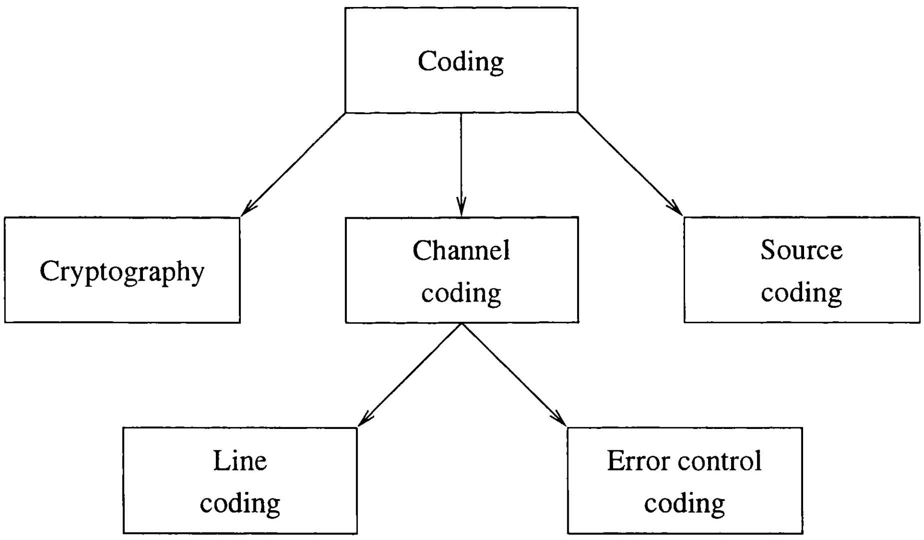

Figure 1.1: Subdivisions of coding

Both line and error control coding are used to improve the reliability of the channel, in two different ways. Line coding modifies the properties of the transmitted sequence in a way that aims to increase the probability of error free transmission. In contrast, error control coding is used to correct and/or detect transmission errors after they have occurred.

The motivation for this work was that the techniques normally designed for generalised channels are not applicable to certain constrained channels. In general, we may wish to match the signal to the channel in more subtle ways than normal, or to incorporate more than one coding objective in a single code. An example of the former is coding to control errors which affect asymmetrically "1's" and "0's" (asymmetric channel). An example of the latter is when a channel requires both line and error control coding: the use of cascaded codes has some disadvantages compared with a single code designed to meet both requirements.

1.1 Organisation of this thesis

Following this introduction, the next chapter provides a brief description of the basic principles of channel coding. Several commonly used terms are explained and some commonly used terms are introduced. Continuing this chapter, line and error control codes are discussed in more detail, and their various subdivisions are described.

Following on from that introduction, the third chapter deals with error correcting codes for an asymmetric channel. This is a channel where the normal assumption that all kinds of errors are equally probable does not hold. The use of specially designed codes gives better results. Several new properties of such codes are introduced and some codes with higher rate than those previously known are presented.

The fourth chapter deals with coding for channels where both disparity control (see section 2.3.1) and error control are desirable. A new error correcting line code structure is proposed that can be applied to codes designed for the binary symmetric and asymmetric channels.

The fifth chapter deals with coding for channels where run length limiting combined with error protection is required. High rate codes that can be combined with systematic error control codes are discussed that exhibit negligible reduction of the error correcting capability of the error control code. Furthermore, a technique to design line codes is proposed that optimises the source word to channel word mapping to allow efficient implementation of the code.

The sixth chapter discusses coding techniques that can be applied in systems where multiple carriers are utilised simultaneously. The main aim here is to reduce the peak to average power ratio of such systems, with a small increase in redundancy and a low level of implementation complexity.

Finally, the seventh chapter concludes the thesis and provides some suggestions for future work.

1.2 Outline of main contributions

The main contributions of this thesis are the following:

- Identification of previously unknown asymmetric error correcting code properties.

- Development of higher rate asymmetric error correcting codes than previously published ones.

• Elaboration of a new disparity error correcting line coding structure.

• Demonstration for the first time of asymmetric error correcting line codes for disparity control.

• Presentation of a coding structure that can achieve overall high coding rate in combined run length limited codes together with conventional systematic error correcting codes, without a noticeable increase in error rate.

• Development of optimisation techniques that provide a reduction in the hardware complexity of line codes.

• Study of the trade–off between implementation complexity and performance in high rate peak to average power ratio reducing codes for multi–carrier systems.

The following papers have been published or have been accepted for publication during the course of this study.

[1] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "A new error correcting line code," in *ITS/IEEE ROC&C International Telecommunications Symposium*, (Acapulco, Mexico), pp. 54–58, Oct. 1996.

[2] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "Exploiting soft decision decoding for error correcting line codes," in *IEEE ICCS/ISPACS Conference Proceedings*, vol. 2, (Singapore), pp. 638–642, Nov. 1996.

[3] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "Soft decision error correcting line code for optical data storage," in *LEOS Conference Proceedings*, vol. 1, (Boston, USA), pp. 201–202, Nov. 1996.

[4] S. Fragiacomo, Y. Bian, A. Popplewell, C. Matrakidis, and J. J. O'Reilly, "An accelerated simulation technique for evaluating communication systems using FEC," in *Proceedings of the European Conference on Networks and Optical Communications*, vol. 2, (Antwerp, Belgium), pp. 145–148, June 1997.

[5] C. Matrakidis and J. J. O'Reilly, "A block decodable line code for high speed optical communication," in *IEEE International Symposium on Information Theory*, (Ulm, Germany), p. 221, June 1997.

[6] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "A class of low complexity line codes," in *IEEE International Symposium on Information Theory*, (Ulm, Germany), p. 219, June 1997.

[7] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "Class of low complexity line codes for optical data storage and communications," in *The Pacific Rim Conference on Lasers and Electro-Optics*, (Chiba, Japan), pp. 92–93, July 1997.

[8] C. Matrakidis and J. J. O'Reilly, "Limiting the maximum run–length of block turbo codes," in *Proceedings of the International Symposium on Turbo Codes & Related Topics*, (Brest, France), pp. 220–222, Sept. 1997.

[9] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "Performance aspects of a class of low complexity line codes," in *International Conference on Signal Processing Applications & Technology*, vol. 1, (San Diego, USA), pp. 632–636, Sept. 1997.

[10] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "Multicarrier transmission peak–to–average power reduction using simple block code," *Electronics Letters*, vol. 34, pp. 953–954, May 1998.

[11] S. Fragiacomo, C. Matrakidis, A. Popplewell, and J. J. O'Reilly, "Novel accelerated technique for low bit error rate communication systems," *IEE Proceedings on Communications*, vol. 145, pp. 337–341, Oct. 1998.

1.3 Summary

This introductory chapter has provided an indication of the motivation for and general orientation of the study of communications coding to be addressed in the remainder of this thesis, together with an outline of the main contributions and details of the results published to date. Given this we turn our attention in the next chapter to reviewing briefly some relevant aspects of channel coding.

2 Aspects of channel coding

2.1 Introduction

Channel coding is the process whereby the reliability of a channel is improved by adding redundancy into the transmitted data. Channel coding is usually subdivided into two categories, line coding and error control coding, that use this redundancy in different ways. Line coding modifies some properties of the transmitted sequence trying to improve the reliability of the transmission, while error control coding attempts to correct and/or detect errors at the receiver.

The main aim of this chapter is to provide a brief introduction to those two channel coding categories. We begin by outlining in section 2.2 the common terms that are going to be used in the remainder of the thesis. In section 2.3 we turn our attention to briefly describing the background and key ideas of line coding. Section 2.4 consists of an overview of the principles of error control coding. Finally, the chapter concludes with a short summary.

2.2 Brief description of common coding terms

There are several common terms relating to channel coding that are used throughout this thesis. The more important ones are as follows.

2.2.1 Code rate

One quantity of interest is code rate. This is a measure of the redundancy added to the transmitted sequence in order to achieve the desired properties. It is the ratio of the number of uncoded information symbols (often bits) divided by the number of encoded symbols that represent the same information. If several coding stages are cascaded, then the overall rate is the product of the rates of all stages.

2.2.2 Hamming distance

The Hamming distance between two sequences is the number of symbol positions in which the two sequences differ. Also of importance is the minimum Hamming distance of a code, which is the minimum Hamming distance calculated between all possible pairs of code–words in the code.

2.2.3 Error correcting/detecting capability

The error correcting/detecting capability of a code is the maximum number of single symbol or bit errors that are guaranteed to be corrected/detected by the code. However, depending on the actual code used and the decoder implementation, there may be specific patterns of more errors that can be corrected/detected.

2.2.4 Error extension

When errors are present in the received sequence that cannot be corrected, the resulting sequence may exhibit more errors after decoding than those that were originally present. This is called error extension. The two measures of error extension commonly used are the average error extension and the maximum error extension. Usually error extension is more important for line codes where typically no error correcting capability exists or is not exploited.

2.2.5 Systematic codes

When the information symbols that are to be transmitted are present unmodified in the encoded sequence then the code is a systematic one, otherwise it is called non–systematic. The symbols of a code–word of a systematic code can be divided into information symbols and redundancy symbols. Many commonly used error correcting codes are systematic, while most line codes are not.

2.3 Line coding

Lines codes are codes designed to modify the characteristics of the transmitted data in a way that suits the transmission channel. This improves the reliability of the transmission. The primary common requirements of a line code are[12]:

- to minimise vulnerability to inter–symbol interference (ISI) and noise;
- to enable extraction of a timing reference; and
- to achieve the first two ends with only modest redundancy.

The two most common categories of line coding are disparity control and run length limiting.

2.3.1 Disparity control line coding

In a binary coding system, disparity is a measure of the imbalance between ones and zeros in the transmitted sequence. It is the difference between the number of transmitted ones and zeros. If disparity is bounded, then the transmitted sequence has no power at very low frequencies. This allows AC coupling of the transmitter and the receiver and facilitates transmission through channels with poor signal to noise ratio at low frequencies[13].

A disparity control line code utilises the added redundancy to limit the disparity of the transmitted sequence between bounds at any point in the sequence. This is usually achieved by bounding the value of the disparity at the end of each code–word.

2.3.2 Run length limiting line coding

A run is a sequence of consecutive identical symbols; run length is the number of consecutive identical symbols. Run length limiting coding is used when the length of the runs needs to be limited. An upper limit allows the extraction of the timing information from the transmitted sequence by guaranteeing the presence of enough signal transitions. A lower limit in the run length is also used on several systems to reduce the inter–symbol interference[14].

2.4 Error control coding

Error control codes (ECCs) are codes designed to detect or correct errors that were inserted during the data sequence transmission. The two main categories of ECC are forward error correction (FEC) where some error correction is performed at the receiver, and automatic repeat request (ARQ) where the receiver detects the errors and requests the retransmission of the corrupted information[15].

Figure 2.1 shows the family of error correcting codes. The first two sub–categories are binary and multi–level codes. Both of these categories can be

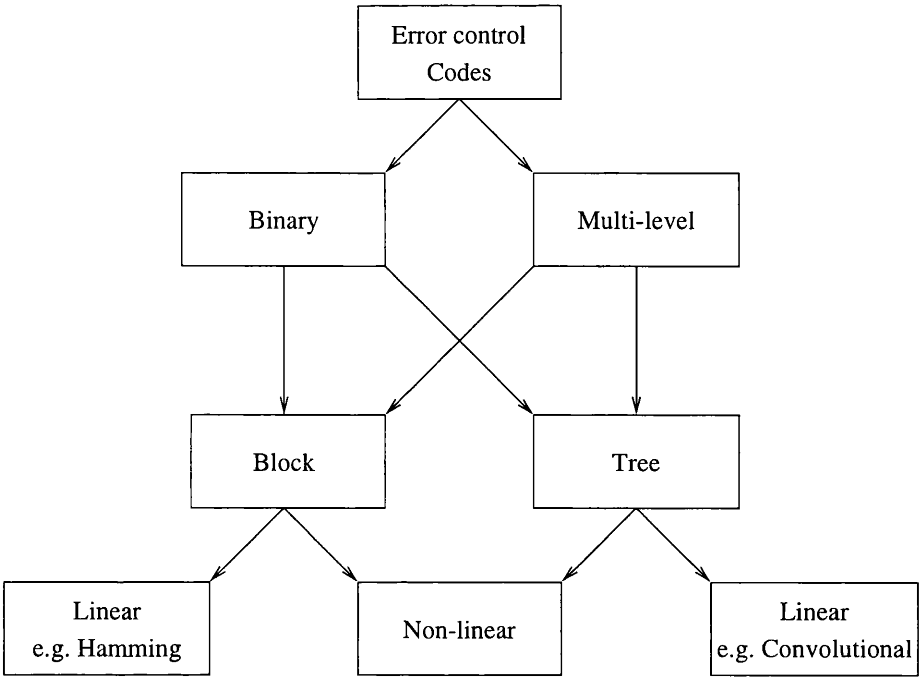

Figure 2.1: Family tree of error correcting codes

split into two other sub–categories, namely block and tree codes. Block codes only depend on the current information word for encoding and decoding, while tree codes utilise memory and the encoding and decoding of each code–word is dependent on a specific number of previous code–words. Furthermore, both tree and block codes can be subdivided into linear and nonlinear codes. Convolutional codes are an important category of linear tree codes, while Hamming, BCH and Reed–Solomon are three well known categories of linear block codes. Reed–Solomon codes are multi–level, while the other are usually binary.

The error correcting capability t of a code is usually (i.e. for symmetric errors) dependent on the minimum Hamming distance d between all pairs of code–words. For such codes this relationships is given by the formula

$$d = 2t + 1.$$

Alternatively, the same code can be used to detect up to $2t$ errors or to correct $2t$ erasures.

Another important distinction in forward error control systems is to do with the way the decoder performs the detection. The simplest decoder architecture only uses the received word hard–limited to ones and zeros (in the binary case) and assumes that possible errors are equally probable in all the symbols. This kind of decoder is called a hard decision decoder. An alternative is a decoder that uses information about the probability of error in all symbols, obtained from the analogue value of the received information. These are called soft decision decoders and can typically correct more (up to $2t$) errors than hard decision ones. However, their implementation is more complicated and under extreme conditions there is the possibility that the may correct fewer errors than a hard decision decoder.

2.5 Summary

This section has introduced some key terms and ideas in channel coding, identifying specifically the usually separate and distinct codes for line and error control coding. With this background we are now in a position to turn our attention to some of the more subtle factors that it may be desirable to address with channel coding. In particular, chapter 3 addresses error control coding designed for the asymmetric error channel.

3 Asymmetric error control codes

3.1 Introduction

This chapter begins with a brief description of the asymmetric channel. Some well known properties of asymmetric Z–channel error correcting codes are then described, followed by a number of properties that were discovered during this study of the channel. Those properties are subsequently used to derive upper bounds on the size of codes that correct asymmetric Z–channel errors.

3.2 Asymmetric errors

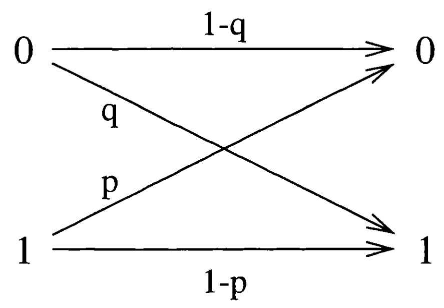

Figure 3.1: A general binary communication channel

The general behaviour of a binary communication channel is shown in figure 3.1. To the left is the input of the channel and to the right is the output. When we transmit a 1 at the input of the channel, we expect to receive a 1 at the output. However there is a (usually small) probability that at the output we will receive a 0. This is called a 1–error, and the probability that it will occur is shown in the figure with the symbol p. On the other hand, if we transmit a 0 and receive a 1, then we have a 0–error, and its probability is shown in the figure with the symbol q.

In most cases, those two probabilities are considered to be independent and equal. In such cases we have what is known as the binary symmetric channel.

There are some cases though, where these two probabilities cannot be considered equal. Then, we have an asymmetric error channel.

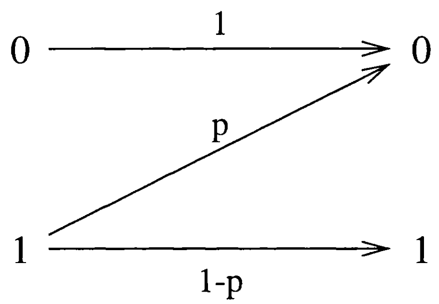

Figure 3.2: Z channel

One such (extreme) case is the Z–channel, which is shown in figure 3.2. In this case, the probability of a 0–error is zero, but the probability of 1–error is p. This is the case in some optical communication systems [16] and some data storage systems [17].

It is clear that the asymmetric errors can be considered as a special case of symmetric errors and treated as such, neglecting the asymmetry. However, if we take advantage of the knowledge we have of the nature of the errors then it may be possible to design more efficient communication systems.

3.3 Asymmetric distance

An important concept for the understanding of Z–channel error–control codes is the asymmetric distance.

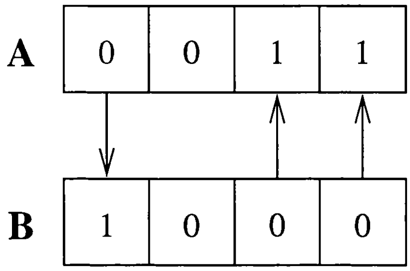

Figure 3.3: Illustrating the quantity $N(\mathbf{x}, \mathbf{u})$ and the asymmetric distance of two binary vectors: $N(\mathbf{A}, \mathbf{B}) = 1$, $N(\mathbf{B}, \mathbf{A}) = 2$ and $d_A(\mathbf{A}, \mathbf{B}) = \max\{N(\mathbf{A}, \mathbf{B}), N(\mathbf{B}, \mathbf{A})\} = 2$

The asymmetric distance is best defined using the quantity $N(\mathbf{x}, \mathbf{u})$ which is defined by $N(\mathbf{x}, \mathbf{u}) = |\{i | x_i = 0 \wedge u_i = 1\}|$, where $\mathbf{x}, \mathbf{u}$ are equal length and with symbols $x_i, u_i, i = 1, 2, \ldots$. This means that $N(\mathbf{x}, \mathbf{u})$ is the number of positions where $\mathbf{x}$ is zero and $\mathbf{u}$ is one.

Figure 3.3 demonstrates the one directional nature of $N(\mathbf{x}, \mathbf{u})$. In the above example, $N(\mathbf{A}, \mathbf{B}) = 1$ and $N(\mathbf{B}, \mathbf{A}) = 2$.

Now, the asymmetric distance $d_A(\mathbf{x}, \mathbf{u})$ is defined as

$$d_A(\mathbf{x}, \mathbf{u}) = \max\{N(\mathbf{x}, \mathbf{u}), N(\mathbf{u}, \mathbf{x})\}.$$

So, in the example shown in figure 3.3, $d_A(\mathbf{A}, \mathbf{B}) = 2$.

This compares to the well known Hamming distance which is defined as

$$d_H(\mathbf{x}, \mathbf{u}) = N(\mathbf{x}, \mathbf{u}) + N(\mathbf{u}, \mathbf{x}).$$

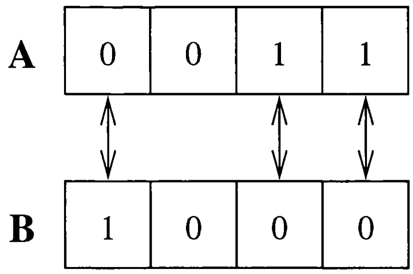

Figure 3.4: The Hamming distance of two binary vectors $d_H(\mathbf{A}, \mathbf{B}) = N(\mathbf{A}, \mathbf{B}) + N(\mathbf{B}, \mathbf{A}) = 3$

In figure 3.4 we see the same two vectors, only this time the arrows are bidirectional, denoting the symmetric nature of the Hamming distance. In this example, $d_H(\mathbf{A}, \mathbf{B}) = 3$.

As we can easily see, $d_A(\mathbf{x}, \mathbf{u}) \leq d_H(\mathbf{x}, \mathbf{u}) \leq 2d_A(\mathbf{x}, \mathbf{u})$. The exact relation between d_A and d_H is

$$2d_A(\mathbf{x}, \mathbf{u}) = d_H(\mathbf{x}, \mathbf{u}) + |w(\mathbf{x}) - w(\mathbf{u})|,$$

where $w(\mathbf{x})$ is the Hamming weight of the binary vector $\mathbf{x}$ and is equal to the number of "1's" in the vector.

3.4 Error correcting capabilities of a code C

The error correcting capabilities of a code C can be described by using two quantities, the minimum asymmetric distance Δ, and the minimum Hamming distance d.

The minimum Hamming distance is defined by

$$d = \min\{d_H(\mathbf{x}, \mathbf{u}) \mid \mathbf{x}, \mathbf{u} \in C;\ \mathbf{x} \neq \mathbf{u}\}$$

and the minimum asymmetric distance by

$$\Delta = \min\{d_A(\mathbf{x}, \mathbf{u}) \mid \mathbf{x}, \mathbf{u} \in C;\ \mathbf{x} \neq \mathbf{u}\}.$$

As is the case between the asymmetric and the Hamming distance of two code–words, it is fairly easy to show that $\Delta \leq d$.

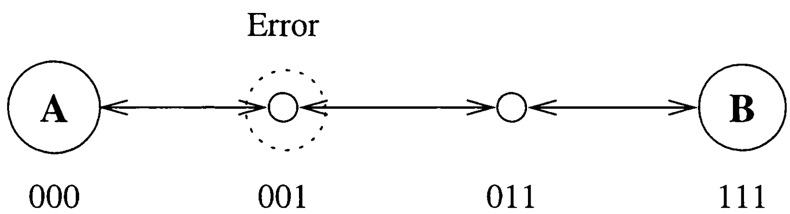

Figure 3.5: An error path between two code–words with $d_H = 3$ in a symmetric error channel

Code C can correct up to K symmetric errors when $2K + 1 \leq d$. An example of why this is the case can be seen in figure 3.5. In this example the two code–words $\mathbf{A} = \{0, 0, 0\}$ and $\mathbf{B} = \{1, 1, 1\}$ are shown along with a possible error path between them. The Hamming distance between these words is 3. If during the transmission through a symmetric error channel, the word $\{0, 0, 1\}$ is received, there are two possible cases. Either **A** was transmitted and one error occurred, or **B** was transmitted and two errors occurred. Assuming the probability of one error occurring is bigger than the probability of two errors occurring, we have to assume that one error occurred. This however means that we can't correct two errors in such a code.

However, C can correct up to T asymmetric Z–channel errors, when $T + 1 \leq \Delta$. An example of this can be seen in figure 3.6. In this example, the same code–words are used as in figure 3.5. The asymmetric distance between **A** and **B**

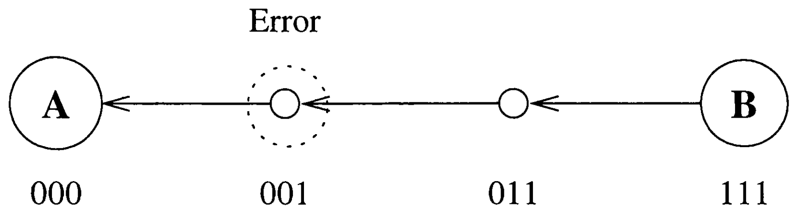

Figure 3.6: An error path between the same code–words as in Fig. 3.5 in the Z–channel. Note that d_A is also 3

is also 3. Now if the word $\{0, 0, 1\}$ is received after the transmission through a z–channel, we know that **B** was transmitted and two errors have occurred. There is no way that **A** was transmitted and one error occurred, due to the one directional nature of the errors in this channel. (Note the arrows in figure 3.6).

We have seen an example, and it can be proved that for a given code the maximum values of T and K have the relationship $T_{max} \geq K_{max}$ i.e. a code can correct at least as many asymmetric Z–channel errors as symmetric ones. This is opposite to the relationship between the symmetric and asymmetric distance, $\Delta \leq d$.

However, Varshamov[18] has proven that most linear codes can correct the same number of symmetric and asymmetric Z–channel errors, so in order to gain from the use of asymmetric codes, nonlinear codes have to be used in most cases.

An interesting point is that the asymmetric errors that can be corrected using a code C don't have to be all 1–errors or all 0–errors, but can be certain combinations. So, a more general expression for the error correcting capabilities of a code C is that the code can correct up to T_1 1–errors and T_0 0–errors if $T_0 + T_1 + 1 \leq \Delta$. As an example, on a generalised asymmetric channel where a single 0–error is more probable than three 1–errors, a code with asymmetric distance 4 can be used to correct any combination of up to a single 0–error and up to two 1–errors.

We can also have codes that can correct either symmetric or asymmetric errors. The necessary and sufficient condition for a code C correcting either K symmetric or T asymmetric errors where $T \geq K$ is given in [19], and is

$$d \geq K + T + 1 \ \wedge \ \Delta \geq T + 1.$$

3.5 Relation between the asymmetric Z–channel error correcting and symmetric error correcting capabilities

A code C with minimum asymmetric distance Δ and minimum Hamming distance d can correct up to $T = \Delta - 1$ asymmetric Z–channel errors or up to K symmetric errors if $d \geq 2K + 1$.

From the definitions of d and Δ it is clear that

$$\Delta \leq d \leq 2\Delta.$$

If K and T are the maximum number of symmetric and asymmetric Z–channel errors that C can correct, then we have the following cases:

If d is odd, then $d = 2K + 1$ and

$$2K + 1 = d \leq 2\Delta = 2(T + 1) \Rightarrow K \leq T + \frac{1}{2}$$

and since K and T are integers,

$$K \leq T.$$

If d is even, then $d = 2K + 2$ and even simpler

$$2K + 2 = d \leq 2\Delta = 2(T + 1) \Rightarrow K \leq T.$$

So, even though $d \geq \Delta$, $K \leq T$ and C can correct at least the same number of asymmetric Z-channel errors as it can correct symmetric ones, which was expected since asymmetric Z–channel errors are a special case of symmetric errors.

3.5 Code word properties

A code C with code–words of length n and minimum asymmetric distance $\Delta(\Delta \leq n)$ has the following properties:

On a code C with minimum asymmetric distance Δ, there can be one and only one code–word with weight less than Δ.

If there is no code–word with weight less than Δ then the all zero code–word $\mathbf{0}$ can be added to the code. For each code word $\mathbf{u}$, where $w(\mathbf{u}) \geq \Delta$ we have $d_A(\mathbf{u}, \mathbf{0}) = w(\mathbf{u}) \geq \Delta$.

Suppose there are two code words $\mathbf{x}$, $\mathbf{u}$ with weight less than Δ. From the definition of d_H it is obvious that $d_H(\mathbf{x}, \mathbf{u}) \leq w(\mathbf{x}) + w(\mathbf{u})$. Then, assuming $w(\mathbf{x}) \geq w(\mathbf{u})$ we have

$$2d_A(\mathbf{x}, \mathbf{u}) \leq w(\mathbf{x}) + w(\mathbf{u}) + |w(\mathbf{x}) - w(\mathbf{u})| = 2w(\mathbf{x}) \leq 2\Delta.$$

So there can be at most one code word with weight less than Δ.

Similarly, there can be one and only one code–word with weight more than $n - \Delta$. This means that if $\Delta \geq \frac{n}{2}$ then C can have a maximum size of two code–words.

If a code C contains a code–word with weight less that the minimum asymmetric distance Δ, then this code–word can be substituted with the all zero code–word $\mathbf{0}$*, without decreasing the minimum asymmetric distance of the code.*

Suppose $\mathbf{x}$ is the code–word with weight less than Δ. Then for each code–word $\mathbf{u}$ in the code, where $\mathbf{u} \neq \mathbf{x}$ we have $w(\mathbf{u}) > w(\mathbf{x})$ and

$$2d_A(\mathbf{u}, \mathbf{x}) \leq w(\mathbf{u}) + w(\mathbf{x}) + |w(\mathbf{u}) - w(\mathbf{x})| = 2w(\mathbf{u}) = 2d_A(\mathbf{u}, \mathbf{0}).$$

In the same way, the code–word with weight more than $n - \Delta$ can be substituted with the all one code–word $\mathbf{1}$.

3.5.1 New properties

In this section, some new properties of the code C with code–words of length n and minimum asymmetric distance $\Delta (\Delta \leq n)$ are presented, together with some constructions for such codes.

A code C with minimum asymmetric distance Δ can be turned into a code with the same minimum asymmetric distance containing at least one code–word of weight Δ.

The code is first transformed so that it includes the all zero code–word $\mathbf{0}$ as described earlier. Then, any code–word with weight Δ has distance Δ from $\mathbf{0}$, which means that potentially it can be part of C. Let's say that the code–word with the next biggest weight is $\mathbf{x}$ and has weight $\Delta + k$. Then, if we set to zero k bits that are one from this code–word, we end up with a code–word $\mathbf{x}'$

with weight Δ. In this case, for any code–word $\mathbf{u} \in C$,where $\mathbf{u} \neq \mathbf{x}, \mathbf{0}$ we have $d_H(\mathbf{x}', \mathbf{u}) \geq d_H(\mathbf{x}, \mathbf{u}) - k$ so

$$2d_A(\mathbf{x}', \mathbf{u}) \geq d_H(\mathbf{x}, \mathbf{u}) - k + |w(\mathbf{u}) - (w(\mathbf{x}) - k)|$$

$$= d_H(\mathbf{x}, \mathbf{u}) + |w(\mathbf{u}) - w(\mathbf{x})| = 2d_A(\mathbf{x}, \mathbf{u}).$$

As in the previous cases, a code can be transformed to have at least one code–word with weight $n - \Delta$.

A code C with code–words of length n and minimum asymmetric distance Δ, with$\Delta \leq \frac{n}{2}$ can have at least two code–words of weight Δ.

As shown in the previous case, C can be transformed to have the all zero code–word $\mathbf{0}$ and a code–word $\mathbf{u}$ with weight Δ. If the code word with the next biggest weight is $\mathbf{x}$ and has weight $\Delta + k$, then it can have at most k common ones with $\mathbf{u}$. If they have m common ones, then $d_H(\mathbf{x}, \mathbf{u}) = 2\Delta + k - 2m$, and we know that

$$2d_A(\mathbf{x}, \mathbf{u}) = d_H(\mathbf{x}, \mathbf{u}) + |w(\mathbf{x}) - w(\mathbf{u})| \geq 2\Delta$$

$$\Rightarrow 2\Delta + k - 2m - |\Delta + k - \Delta| \geq 2\Delta \Rightarrow k \geq m.$$

So, if $k = 0$ we have two code–words in C with weight Δ that have no common ones, which is the necessary condition for their asymmetric distance to be Δ. On the other hand, if $k \neq 0$ then if we delete k ones from $\mathbf{x}$ including all common ones with $\mathbf{u}$, then we end with a code word $\mathbf{x}'$ with weight Δ, which has asymmetric distance Δ from $\mathbf{0}$ and $\mathbf{u}$. For every other code–word $\mathbf{v} \in C$, where $\mathbf{v} \neq \mathbf{0}, \mathbf{x}, \mathbf{u}$, we have $d_H(\mathbf{x}', \mathbf{v}) \geq d_H(\mathbf{x}, \mathbf{v}) - k$ so

$$2d_A(\mathbf{x}', \mathbf{v}) \geq d_H(\mathbf{x}, \mathbf{v}) - k + |w(\mathbf{v}) - (w(\mathbf{x}) - k)|$$

$$= d_H(\mathbf{x}, \mathbf{v}) + |w(\mathbf{v}) - w(\mathbf{x})| = 2d_A(\mathbf{x}, \mathbf{v}).$$

Again, we can transform C if $n \geq 2\Delta$ to have at least two code–words with weight $n - \Delta$.

A corollary of the above statement is that if $n = 2\Delta$, then C can have up to 4 code–words, one with weight 0, one with weight n and two with weight Δ.

A code C with code–words of length n and minimum asymmetric distance Δ, with $n \geq \frac{5}{2}\Delta$ can have at least two code–words of weight Δ and at least another two code–words of weight $n - \Delta$.

	$\overbrace{\quad}^{\lfloor \frac{\Delta}{2} \rfloor}$	$\overbrace{\quad}^{\lceil \frac{\Delta}{2} \rceil}$	$\overbrace{\quad}^{\lfloor \frac{\Delta}{2} \rfloor}$	$\overbrace{\quad}^{\lceil \frac{\Delta}{2} \rceil}$	$\overbrace{\quad}^{\lceil \frac{\Delta}{2} \rceil}$
$w = 0$	$0 \cdots 0$	$0 \cdots 0$	$0 \cdots 0$	$0 \cdots 0$	$0 \cdots 0$
$w = \Delta$	$1 \cdots 1$	$1 \cdots 1$	$0 \cdots 0$	$0 \cdots 0$	$0 \cdots 0$
$w = \Delta$	$0 \cdots 0$	$0 \cdots 0$	$1 \cdots 1$	$1 \cdots 1$	$0 \cdots 0$
$w = n - \Delta$	$0 \cdots 0$	$1 \cdots 1$	$1 \cdots 1$	$0 \cdots 0$	$1 \cdots 1$
$w = n - \Delta$	$1 \cdots 1$	$0 \cdots 0$	$0 \cdots 0$	$1 \cdots 1$	$1 \cdots 1$
$w = n$	$1 \cdots 1$	$1 \cdots 1$	$1 \cdots 1$	$1 \cdots 1$	$1 \cdots 1$

Figure 3.7: A code with $n = \lceil \frac{5}{2}\Delta \rceil$

Figure 3.7 shows the construction of a code that has $n = \lceil 5\Delta/2 \rceil$ and has six code–words, with the above mentioned properties. Therefore, any code with $n \geq 5\Delta/2$ can have at least six code–words since according to the properties discussed earlier it can be transformed to one that will have at least two code words with weight Δ and at least another two code–words with weight $n - \Delta$.

Finally, the following construction shows that a code with even n and $n = 3\Delta$ can have at least 12 code–words.

	$\overbrace{}^{\frac{\Delta}{2}}$	$\overbrace{}^{\frac{\Delta}{2}}$	$\overbrace{}^{\frac{\Delta}{2}}$	$\overbrace{}^{\frac{\Delta}{2}}$	$\overbrace{}^{\frac{\Delta}{2}}$	$\overbrace{}^{\frac{\Delta}{2}}$
$w = 0$	$0\cdots0$	$0\cdots0$	$0\cdots0$	$0\cdots0$	$0\cdots0$	$0\cdots0$
$w = \Delta$	$1\cdots1$	$1\cdots1$	$0\cdots0$	$0\cdots0$	$0\cdots0$	$0\cdots0$
$w = \Delta$	$0\cdots0$	$0\cdots0$	$1\cdots1$	$1\cdots1$	$0\cdots0$	$0\cdots0$
$w = \Delta$	$0\cdots0$	$0\cdots0$	$0\cdots0$	$0\cdots0$	$1\cdots1$	$1\cdots1$
$w = n/2$	$0\cdots0$	$1\cdots1$	$0\cdots0$	$1\cdots1$	$1\cdots1$	$0\cdots0$
$w = n/2$	$1\cdots1$	$0\cdots0$	$1\cdots1$	$0\cdots0$	$1\cdots1$	$0\cdots0$
$w = n/2$	$0\cdots0$	$1\cdots1$	$1\cdots1$	$0\cdots0$	$0\cdots0$	$1\cdots1$
$w = n/2$	$1\cdots1$	$0\cdots0$	$0\cdots0$	$1\cdots1$	$0\cdots0$	$1\cdots1$
$w = n - \Delta$	$0\cdots0$	$0\cdots0$	$1\cdots1$	$1\cdots1$	$1\cdots1$	$1\cdots1$
$w = n - \Delta$	$1\cdots1$	$1\cdots1$	$0\cdots0$	$0\cdots0$	$1\cdots1$	$1\cdots1$
$w = n - \Delta$	$1\cdots1$	$1\cdots1$	$1\cdots1$	$1\cdots1$	$0\cdots0$	$0\cdots0$
$w = n$	$1\cdots1$	$1\cdots1$	$1\cdots1$	$1\cdots1$	$1\cdots1$	$1\cdots1$

Figure 3.8: A code with $n = 3\Delta$, when Δ is even

3.5.2 Code size

From the above sections we can derive the following bounds for the size Z of an asymmetric Z–channel error control code with code–words of length n and minimum asymmetric distance Δ, $(\Delta \leq n)$:

$$Z = 2 \qquad n < 2\Delta$$

$$Z = 4 \qquad 2\Delta \leq n < 5\Delta/2$$

$$Z \geq 6 \qquad 5\Delta/2 \leq n$$

The first statement, as well as the third one have been proved above. For the second statement, we already know that $Z \geq 4$ for $n \geq 2\Delta$. Therefore the only thing to show is that after the all zero code–word and the two with weight Δ there is only one more code–word for $n < 5\Delta/2$.

Let $n = 2\Delta + x$ with $x < \Delta/2$ and c_1, c_2 be the two code–words with weight Δ. The minimum weight for a fourth code–word with distance at least Δ from c_1 and c_2 is $2(\Delta - x) + x = 2\Delta - x$. Now, $n - (2\Delta - x) = 2\Delta + x - 2\Delta + x = 2x$ and since $x < \Delta/2$ this code–word is in the area where only one code–word exists. Therefore the code can have up to 4 code–words.

3.6 Integer programming upper bound

This section describes the procedure used to calculate the upper bound $Z(n, \Delta)$ to the size of an asymmetric Z–channel error correcting code C with code–words of length n and minimum asymmetric distance Δ.

For this purpose, relations between the number of code–words with different weight are used. Let A_i be the number of code–words that have weight i. Kløve[20] proves that

$$\sum_{j=w-s}^{w} A(s, 2\Delta, w-j)A_j \leq A(n+s, 2\Delta, w)$$

with $s = 0, 1, \ldots, w$ and $w = 0, 1, \ldots, n$. $A(n, d, w)$ is the maximum number of code–words in a code of length n and minimum distance d that have constant weight w. When the value of $A(n, d, w)$ is not known, the lower bound can be used on the left hand side while the upper bound can be used on the right hand side. Tables of bounds on the size of $A(n, d, w)$ can be found in the appendix.

Furthermore, some additional constraints can be exploited. As we have seen earlier,

$$\begin{aligned}
&A_0 = A_n = 1, && \\
&A_i = 0 && i = 1, \ldots, \Delta - 1, n - \Delta + 1, \ldots, n - 1, \\
&A_\Delta \geq 2 && n \geq 2\Delta \text{ and} \\
&A_{n-\Delta} \geq 2 && n \geq 5\Delta/2.
\end{aligned}$$

It must be noted here that the quantities A_i are integers, therefore integer programming can be used in place of linear programming to give more accurate results. For this purpose, the branch and bound technique was implemented as described by Sultan[21], with additional refinements described by Walukiewicz[22]. Some additional constraints given by Weber et. al. [23] can also be utilised to improve the accuracy of the results.

n,Δ	2	3	4	5	6	7
5	6	2	2	2	–	–
6	12	4	2	2	2	–
7	18	4	2	2	2	2
8	36	7	4	2	2	2
9	62	12	4	2	2	2
10	108–117	18_w	6	4	2	2
11	180_b–210	**30**$^+$	8	4	2	2
12	340_z–410	54–63	12	4	4	2
13	652_b–786	98–**108**	18	6	4	2
14	1204_b–1500	186–**208**	30–34	8	4	4
15	2216_z–2828	266–**384**	44–**46**	12	6	4
16	4232_b–5430_w	364–**734**	66–**88**	16–17	7	4
17	7968_b–10374	647–**1278**	122–**160**	**26**$^+$	8	4
18	14624_b–19898	1218–2380	234–**308**	36–44	12	6
19	28032_b–38008	2050–4242	450–**602**	46–80	15*–**16**$^+$	7
20	53856_b–73174_w	2564–**8068**	860–1144_w	54–138	22*–**25**	9
21	101576_b–140798	4251–**14162**	1628–**2094**	62–230	32*–**36**	12
22	195700_b–271953	8450–26679	3072–**4081**	88–**412**	48*–**64**	14_s–**16**
23	349536–**523584**	16388–50200	4096–**7260**	133–**742**	65*–**110**	19_s–**22**

Table 3.1: Bounds on the asymmetric Z–channel error correcting code size. Upper bounds obtained from linear programming, with new tighter ones shown in bold, except + obtained through exhaustive search and w from [24]. Lower bounds are from [24], except s[25], z[26], b[27] and $*$ found with new search algorithm

Table 3.1 shows the upper bounds obtained in this way, with the exception of the values marked by + that are discussed later and the ones marked with w that were taken from reference [24]. In that reference, there were a few tighter upper bounds for the case $\Delta = 5$. However these depend on the values of $A(n, 10, w)$, which reference [28] states are unreliable, as further discussed in the appendix.

The upper limits noted with + were obtained by performing an exhaustive search on all words that have weight $w > n/2$ with n odd. The upper limit is clearly twice the number of code–words found with this search. The upper bounds that are tighter that any previously published ones are shown in bold.

3.7 Lower bound on the number of code–words

Table 3.1 also shows the lower bound to the number of code–words. Most of these values were taken from [29], except for the values noted with s[25], z[26], b[27] and $*$ that will be discussed later.

All values in the table with up to 12 code–words have been verified as exact by using exhaustive search. However, this search becomes computationally infeasible as the number of code–words increases. In practice, it was impossible to exhaust the search space for any code with more than 12 code–words.

Therefore it is appropriate to use heuristic search algorithms to try and find codes larger than the existing ones. One such procedure was developed by Saitoh[25]. The code space is partitioned into sets that have the same weight. A "greedy algorithm" is then used to search each set, either starting from the one with weight $w/2$ and proceeding towards the sets with weights Δ and $n-\Delta$, or the other way round. Each set was searched in lexicographic order.

Based on this technique, a new procedure was developed. The original algorithm was searching the code space in order, while the new one was modified to search using a random sequence. After a few trials employing different random search sequences, some new codes were obtained. These code sizes are marked with an asterisk in the table.

3.8 Summary

In this chapter, the basic properties of the asymmetric error Z–channel were discussed, together with the properties of the error correcting codes used specifically for this channel.

It was shown, that such a code can be modified to include the all zero and all one code–words. Furthermore, if the code can have more code–words then it can be modified to include at least two with weight equal to the minimum asymmetric distance Δ. Moreover, two code constructions were presented showing that a code with length $n = \lceil \frac{5}{2}\Delta \rceil$ can have at least six code–words and a code with $n = 3\Delta$ can have at least twelve code–words when Δ is even.

Finally, new lower and upper bounds on the number of code–words that are available for a specific code–word length and asymmetric distance were obtained. The upper bounds were calculated by either solving integer programmes incorporating those properties, or by performing exhaustive searches on subsets of the code space. The lower bounds were improved by obtaining new codes using a heuristic search technique.

Having identified some key properties of asymmetric Z–channel error control codes, we will now turn our attention to a different aspect of channel coding, namely disparity control line coding, and in particular combined disparity and error control coding. The properties of asymmetric Z–channel error control codes will be utilised in this respect to investigate disparity control codes for this channel as well.

4 Combined disparity and error control coding

4.1 Introduction

There are two areas of coding that are of particular interest in the design of a communications system. These are error control coding, used to detect and/or correct errors and line coding, used to modify the characteristics of the transmitted signal to match the requirements of the channel.

This chapter deals with the effects of the interaction of those two coding operations when present in the same system. First the disparity control requirements are discussed and a commonly used coding technique is presented. Then the advantages of a combined disparity and error control code are discussed and a new code structure is introduced. Finally, codes for combined disparity and asymmetric Z–channel error control coding are identified. The chapter concludes with a short summary.

4.2 Line coding

Roughly speaking, the objective of a line code is to improve the quality of the transmitted data. In this respect, the primary requirements of a line code are[12]:

- to minimise vulnerability to ISI (inter–symbol interference) and noise;
- to enable extraction of a timing reference; and
- to achieve the first two ends with only modest redundancy.

While the achievement of any two of the above is easy, meeting all three is more complicated.

The two most widely used categories of line codes are the disparity limited line codes and the run length limited line codes.

4.2.1 Disparity limited line codes

Disparity is a measure of the imbalance of the symbols being transmitted, and can be defined as follows (with the definition being applicable to any radix r). Let a_i be the ith digit of a sequence, and $\overline{a_i} = r - 1 - a_i$ be its complementary value. Then the disparity of the digit is $d_i = a_i - \overline{a_i}$, and the disparity of a n–digit word is

$$d = \sum_{i=1}^{n} d_i = 2\sum_{i=1}^{n} a_i - n(r-1).$$

If we are using binary codes, as we will be doing for the rest of this chapter, then the disparity of a code word is given as

$$d = 2\sum_{i=1}^{n} a_i - n.$$

This is equal to the number of ones minus the number of zeros.

If the quantity of interest is the disparity of a single binary code–word c, then $\sum_{i=1}^{n} a_i$ is called the Hamming weight of c, and is represented by $w(c)$. Then,

$$d = 2w(c) - n.$$

In figure 4.1 an example of three code–words of length $n = 4$ along with their corresponding disparity is given.

	Code-word				Disparity
A	0	1	0	1	0
B	1	0	1	1	+2
C	0	1	0	0	-2

Figure 4.1: Some code–words and their disparity

What we are really interested in is that the disparity of a transmitted signal should be bounded. This means that the imbalance of transmitted ones and zeros will never exceed a fixed value. This way we can achieve a frequency response with no DC component[30, 31], which in effect limits the inter–symbol interference[12]. The disparity of the transmitted signal is calculated by accumulating a sum of the disparities of all the transmitted words. This sum is called the running disparity. If the running disparity is bounded, then it is clear that the disparity of the transmitted signal is also bounded.

4.3 Alternate dictionary line codes

One category of line codes that can provide bounded disparity is called alternate dictionary line codes. The idea behind them is straightforward: In the simplest form the encoder has a choice of two code–words, one with positive and one with negative disparity for each input word and selects the appropriate one according to the sign of the running disparity. So, while the running disparity is positive, only code–words with negative disparity are used, and when the running disparity is negative only code–words with positive disparity are used. A simple binary implementation adds one zero bit to each word in the input sequence, and the the alternate words are these and their inverse (which have opposite disparity)[32].

input	+ disparity	−
000	1111	0000
001	1101	0010
010	1011	0100
011	1001	0110
100	0111	1000
101	0101	1010
110	0011	1100
111	1110	0001

Table 4.1: A 3B4B alternate line code with the last bit indicating inversion of the code–word

Such a code is shown in table 4.1. This is a 3B4B code because it encodes 3 bits of input information into 4 bits that are to be transmitted. The encoding is done as follows: At the end of every three bits a zero is added. Then depending on the running disparity the new code–word is transmitted as is or inverted, with the last bit used to indicate the inversion at the decoder.

For the case where the number of bits n in the codeword is even, there exist words that have zero disparity. The transmission of such a word doesn't affect the running disparity, so it can be used in both dictionaries[33].

Table 4.2 gives an example of such a simple alternate dictionary line code. This is also a 3B4B code. The disparity of this code is bounded between 0 and +2, since six of the output words have disparity 0, and the remainder are mapped from two alternate pairs, each with ±2 disparity. This code offers

input	+ disparity −	
000	1011	0100
001	0011	
010	0101	
011	0110	
100	1001	
101	1010	
110	1100	
111	1101	0010

Table 4.2: A 3B4B alternate line code employing

zero disparity words in both dictionaries

tighter disparity bounds than the code in table 4.1 whose disparity is bounded by ± 3.

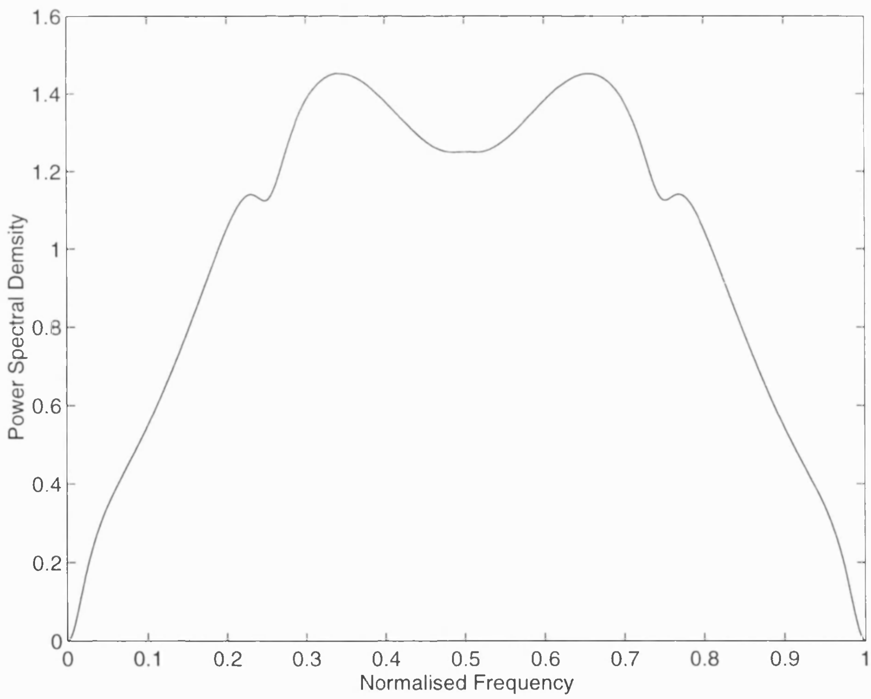

Figure 4.2: Frequency response

of the 3B4B line code of table 4.2

In figure 4.2 we see the frequency response of the 3B4B line code from table 4.2; this was obtained using the procedure described in [34, 35, 36, 37]. There is no DC component in the frequency response and the low frequency content is limited, the main objective which improves the immunity of the coded signal to inter–symbol interference.

4.4 Error correcting line codes

A common requirement in a coding system is to have both error protection and line coding properties.

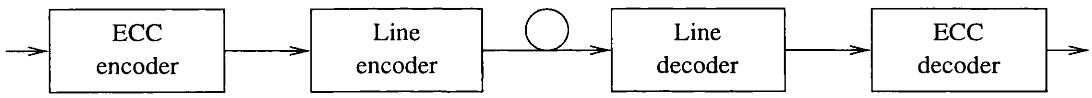

Figure 4.3: A cascaded error control and line coding system

A commonly used arrangement in such a case is shown in figure 4.3. A line code is used to give the required line coding properties to the transmitted sequence, while an outer error control code is used to correct any errors that occur. There are two significant drawbacks in such a system. The first is that both codes add redundancy, and thereby the overall rate of the code may be significantly reduced. Furthermore, many line codes give rise to what is known as error extension: Since the line codes offer no error control, it is possible that an error that occurred in the channel will confuse the line decoder and will result in more errors that the error control code will have to correct.

To avoid both of those problems, the family of error correcting line codes has been proposed[38, 39]. Their main feature is that the transmitted code words have the desired line coding properties, while at the same time they are valid error correcting code words. Therefore, it is possible to effect the error correction before the line decoding, avoiding possible error extensions. Moreover, it is possible to achieve higher coding rates by using such codes,albeit potentially at the expense to some degree of the overall system complexity.

One such code that can be used when bounded disparity is desired together with error protection has been proposed[39] that partitions an error correcting code into two alternate dictionaries that have positive and negative disparity respectively. The code used is a systematic linear transparent code and is divided in such a way that every code–word corresponds to its binary inverse in the alternate dictionary. Using this partition scheme, one bit in the transmitted word is used to indicate whether the code–word has been inverted for transmission, while the code–word is still valid.

To achieve higher efficiency, a scheme with more complicated encoding and decoding has also been proposed[40] where code–words that have zero disparity are used in both of the alternate dictionaries. Using this method, the number of available entries in the dictionary increases.

4.4.1 Proposed technique.

This technique can be further extended by making the following observation: the code–words in the alternate dictionaries that correspond to the same input information need not satisfy the Hamming distance constraint between them. An error that will fail to be corrected this way (provided that it is within the error correcting capabilities of the code) will transform the code–word to the alternate one which will be decoded as the same information. An extreme case is the previously mentioned technique, where both alternate code–words are the same, and therefore have zero Hamming distance.

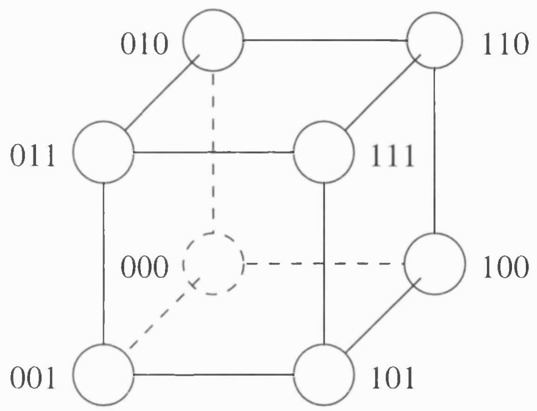

Figure 4.4: The 3 bit code–word space

Figure 4.4 helps illustrate this point. Here, the eight possible 3 bit code–words are displayed as a cube. All code–words that have distance one are connected with a line. It is fairly easy to see that a disparity code with minimum Hamming distance two can only have three code–words.

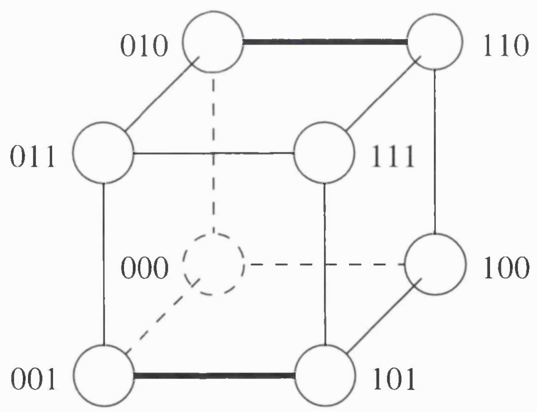

Figure 4.5: Two pairs that form a 3 bit error correcting line code

However, we can select two alternate pairs, with each pair having distance two from the other pair, while the code–words of each pair have distance one. This is shown in figure 4.5, where each pair is connected with a thick line. This simple example is used to demonstrate that higher rate codes are possible using this technique.

4.4.2 Example.

It is possible to design error correcting line codes that have more code–words pairs than the partition into two sets of the longest possible error correcting code of the same length.

A more practical example is given in table 4.3. This is a 9–bit distance 3 code where the possible number of pairs is 21. However, the maximum number of code–words in a 9–bit error correcting code is 40 [28]. Therefore this code can give error correcting line codes with more code–word pairs than is possible using a normal error correcting code. In the table, the pairs where the distance of the pairs is less than 3 are shown with an underlined word number. Only three of the 21 pairs meet the minimum distance constraint.

Figure 4.6 shows the power spectral density of this code.

This code was obtained using a computer search employing a "greedy algorithm". Originally, the set of code–words with positive disparity was searched in order and each word that had minimum distance at least three from all selected words was also selected. Then, for all possible pairs of the initial set of words, the procedure was repeated with the pair having exchanged positions. The best code obtained with this procedure was retained and the set of code–words with negative disparity was searched for the alternate code. This was

word	+ disparity −	
1	001111111	010000000
2	010001111	000001111
3	010110011	000110011
4	010111100	000011100
5	011010101	001010101
6	011011010	001001010
7	011100110	000100110
8	011101001	001101001
9	100010111	100010010
10	100101011	100101000
11	100111101	011001100
12	101001101	101000100
13	101011011	101011000
14	101100111	100000001
15	101110001	001110000
16	110011001	010011001
17	110011110	010010110
18	110100101	010100101
19	111000011	011000011
20	111101010	010101010
21	111110100	110110000

Table 4.3: An error correcting alternate line code with 9 bits and minimum distance 3. The underlined words show the pairs with distance less that three

done with a similar procedure that selected words that had distance less than three from their pair whenever possible. The whole procedure was repeated until no further improvement was obtained.

4.5 Asymmetric error correcting codes

It is also possible to design error correcting line codes suitable for the asymmetric Z–channel. Such a code can be split in two subsets of positive and negative

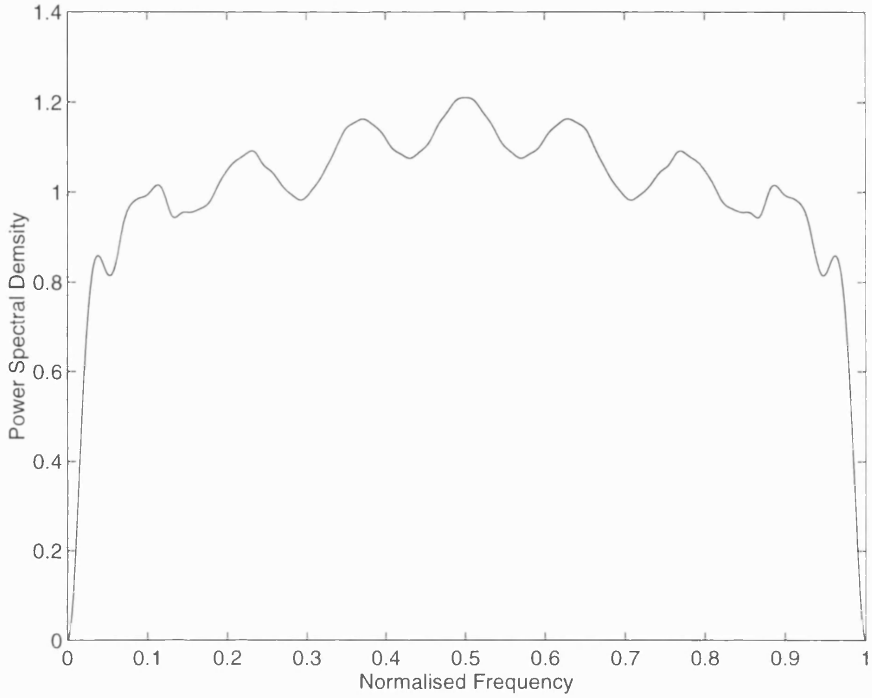

Figure 4.6: Frequency response of the new 9–bit error correcting line code

disparity that can be alternatively used depending on the running disparity of the transmitted sequence.

The case where the asymmetric distance Δ is two (that is, single asymmetric Z–channel error correcting codes), will be studied in more detail.

For a code with even word length, no improvement is possible with the proposed technique. Since the words with zero disparity can be transmitted in any case, the only improvement in code size would be possible if pairs of words with positive and negative disparity were possible with distance less than two. However, the difference in Hamming weight and the Hamming distance between the words of every possible pair are both greater than or equal to two. Therefore, from the formula $2d_A(a,b) = d_H(a,b) + |w(a) - w(b)|$ we get that the asymmetric distance of the pair will be greater than or equal to two.

However, in the case where the length of the code–words is odd, improvements are possible. It can be shown that any single asymmetric Z–channel error correcting code that consists of words of only positive (or negative) disparity can form one half of an alternate asymmetric Z–channel error correcting line code. Specifically, for two single asymmetric Z–channel error correcting codes, one with positive and one with negative disparity, a code–word of one code will have distance less than two with at most one code–word of the other code.

Assume three codewords c_1, c_2 and c_3, with $w(c_1) \leq w(c_2) \leq w(c_3)$. Since $2d_A(a,b) = d_H(a,b) + |w(a) - w(b)|$, we have that

$$2d_A(c_1,c_2)+2d_A(c_1,c_3) = d_H(c_1,c_2)+d_H(c_1,c_3)+w(c_2)-w(c_1)+w(c_3)-w(c_1).$$

However, $d_H(a,b) + d_H(b,c) \geq d_H(a,c)$ gives us

$$2d_A(c_1,c_2) + 2d_A(c_1,c_3) \geq d_H(c_2,c_3) + w(c_2) - w(c_1) + w(c_3) - w(c_1) \Leftrightarrow$$

$$\Leftrightarrow 2d_A(c_1,c_2) + 2d_A(c_1,c_3) \geq 2d_A(c_2,c_3) + 2w(c_2) - 2w(c_1) \Leftrightarrow$$

$$\Leftrightarrow d_A(c_1,c_2) + d_A(c_1,c_3) \geq d_A(c_2,c_3) + w(c_2) - w(c_1).$$

If c_1 has negative disparity, and c_2, c_3 have positive disparity, then $w(c_2) - w(c_1) \geq 1$ and since $d_A(c_2,c_3) \geq 2$ we have

$$d_A(c_1,c_2) + d_A(c_1,c_3) \geq 3.$$

Therefore, only one of c_2 and c_3 – say c_2 – can have asymmetric distance less than two from c_1, and in the same way it can be shown that no other code–word of negative disparity can have distance less than two from c_2.

This helps the search for asymmetric Z–channel error correcting line codes with asymmetric distance at least 2 and an odd word length. For example the procedure that was described earlier can be used with the simplification that only the positive disparity set needs to be searched since the existence of a negative counterpart is guaranteed. Assuming the code with the most code–words is found, a code with the same size but negative disparity words can be generated by inverting all words. Then all pairs of distance less than two can be identified, and the remaining code–words can be paired arbitrarily. The search space can be reduced further by employing the properties of the asymmetric Z–channel error correcting codes discussed in the previous chapter. The all one codeword and two other codewords of weight $n-2$ can be selected as parts of any code, simplifying the search.

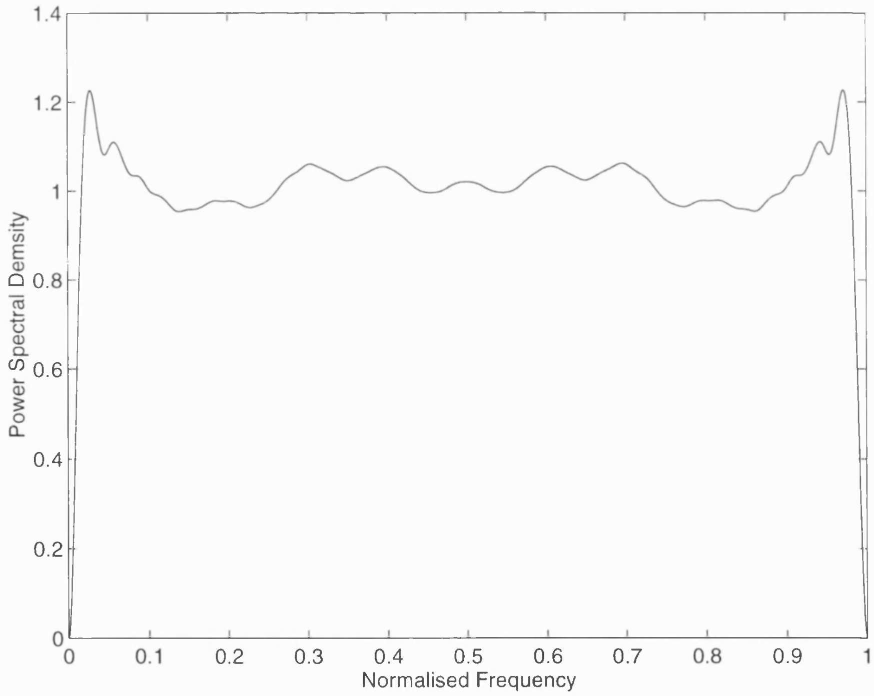

Figure 4.7: Power spectral density of the new 11–bit asymmetric Z–channel error correcting line code

(05f,000),	(0b7,003),	(0eb,00c),	(0fc,120),	(0ff,506),	(12f,410),
(173,133),	(17d,2e4),	(19d,09d),	(1ba,280),	(1c7,0c7),	(1db,645),
(1ee,4b2),	(23b,228),	(267,032),	(27e,552),	(29e,296),	(2ad,111),
(2cf,682),	(2d5,049),	(2f2,405),	(2f9,4e8),	(317,064),	(33f,1e0),
(34d,4c0),	(35a,31a),	(36b,11c),	(374,370),	(38b,250),	(3a6,098),
(3b1,391),	(3bc,474),	(3d6,32c),	(3e8,142),	(3ef,1b4),	(3f5,443),
(3fa,388),	(43d,0a1),	(46e,44e),	(477,6d0),	(48f,48b),	(4bb,598),
(4d3,700),	(4dd,50d),	(4e5,4a5),	(51b,206),	(536,40a),	(54f,459),
(555,155),	(569,561),	(57a,25c),	(57f,0f1),	(59e,235),	(5a3,184),
(5ac,01b),	(5af,344),	(5b5,703),	(5ca,494),	(5d7,0da),	(5f0,056),
(5f9,611),	(61f,5c4),	(64b,530),	(656,078),	(66d,52a),	(671,261),
(67b,2b0),	(699,0a6),	(6aa,2aa),	(6b4,285),	(6bd,192),	(6cc,68c),
(6da,417),	(6df,1a9),	(6e3,3c2),	(6ee,03e),	(70e,0cc),	(725,7a0),
(733,2c9),	(738,638),	(759,253),	(75e,322),	(762,662),	(767,581),
(787,20f),	(792,748),	(79b,624),	(7a9,429),	(7b6,24a),	(7c1,714),
(7cd,06d),	(7e4,125),	(7f3,18e),	(7fc,166),	(7ff,14b)	

Table 4.4: New 11–bit single asymmetric Z–channel error correcting line code with 95 code–word pairs (in hexadecimal). Underlined are the pairs with asymmetric distance less than two

Table 4.4 shows, in hexadecimal, the 95 pairs that form an 11–bit code with minimum asymmetric distance between pairs at least 2. This compares favourably to the code that can be constructed from the best known asymmetric Z–channel error correcting code, that has 180 codewords [27] giving up to 90 pairs. This code has 16 pairs with asymmetric distance less than two, shown underlined in the table. Figure 4.7 shows the power spectral density of this code.

This code was generated with the above procedure, with an added step. After a code of 94 code–words was generated, an added optimisation step was used. This tried to improve all subsets of the code with size $94 - k$, where k was progressively increased, by exhaustively searching all possible combinations of the remaining words with positive disparity. The result of 95 words was obtained for $k = 5$. For values of k higher than 6 this optimisation was found to be computationally infeasible.

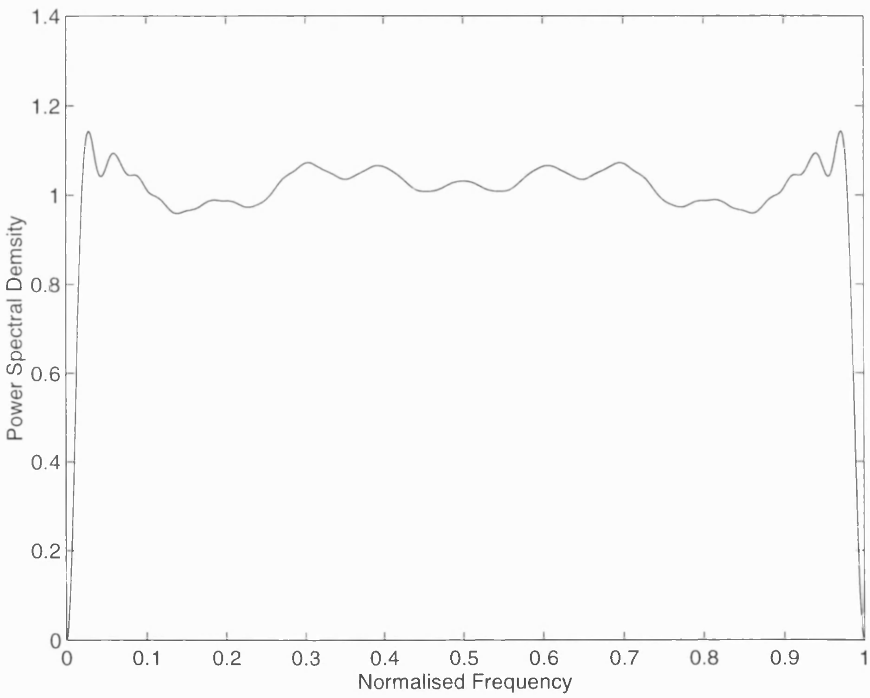

Figure 4.8: Power spectral density of the reduced 11–bit asymmetric Z–channel error correcting line code

This code can be transformed to satisfy a tighter disparity bound, by removing the all one and all zero code–words. The power spectral density of the resulting 94 word code is shown in figure 4.8, where a small reduction of the low frequency content can be seen.

4.5.1 Alternative encoding scheme.

Another improvement that requires more complicated encoding is possible using a technique similar to that described in [9]. This requires that the code–words are paired in such a way that words with high positive (negative) disparity are paired with words with low negative (positive) disparity. The encoder can then select the code–word that gives a value of disparity closer to zero which on occasion will have the same sign as the running disparity. A small example will illustrate the point. If the current disparity is -1 and the words in the pair that is to be transmitted have disparities of $+7$ and -1, selecting based on the sign the disparity would become $+6$, while selecting the other one the disparity becomes -2.

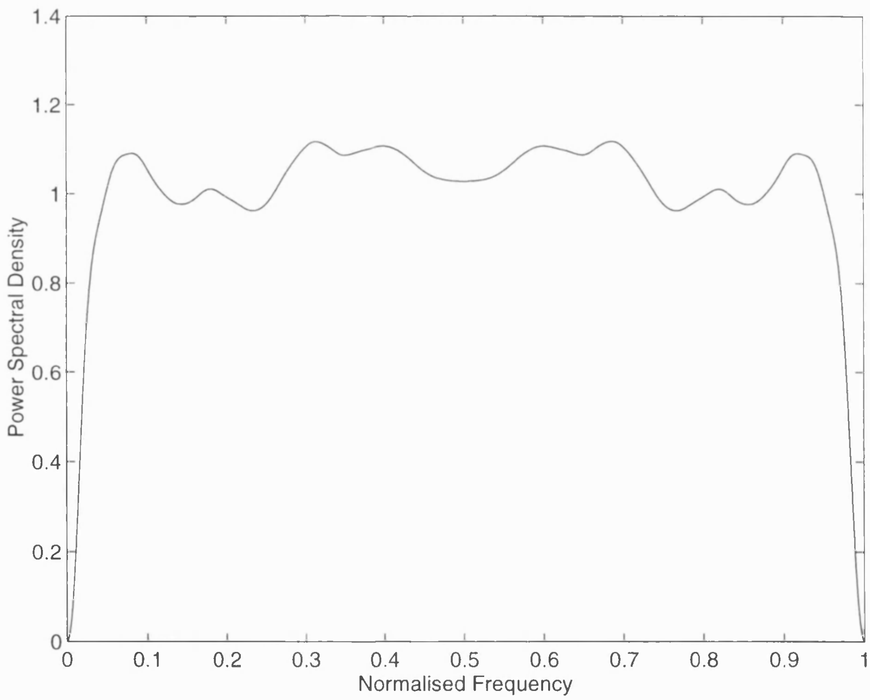

Figure 4.9: Power spectral density of the same 11–bit code using the more complicated encoder

The code that was given in table 4.4 was designed for use with this technique, and the resulting power spectral density is shown in figure 4.9. A considerable improvement in the low frequency content is evident from this graph. Removing the all one and all zero words from this code makes no apparent difference in the low frequency content, which is an indication that the variance of the disparity is very low.

4.6 Summary

In this chapter, the basic forms of line coding were discussed, as well as the need for combined error control and line coding schemes. The family of error correcting line codes that limits the low frequency content of the transmitted sequence by limiting the disparity of the transmitted sequence was presented. The requirements of such codes were discussed in the context of channels displaying both symmetric errors and asymmetric Z–channel errors and a new class of such codes was introduced that is shown that to have more code–words than previously published ones for some specific cases.

Two such examples of disparity limiting error correcting codes were presented, that achieve higher rate than the presently known ones. One is a 9–bit single symmetric error correcting code with 21 pairs of code–words. The other is a 11–bit single asymmetric Z–channel error correcting code with 95 code–word pairs.

Finally, an alternative encoding scheme was proposed that uses more complicated encoding rules than the traditional one. This is shown to achieve tighter disparity bounds and consequently better low frequency suppression, using the above mentioned single asymmetric Z–channel error correcting code.

Disparity control is but one common feature which may be incorporated into a line code. Another, and sometimes quite separate, requirement can be to limit the lengths of "runs" of identical symbols. A category of codes that can be used in such situations are called run length limited (RLL) codes. The following chapter considers this class of codes and their combination with error control coding.

5 Combined run–length limited and error control coding

5.1 Introduction

There are certain channels, where it is advantageous to transmit sequences with limited runs of consecutive identical symbols. As an example, there is evidence that the error performance of long–haul high speed optical communication systems, employing optical amplifiers, can be improved if maximum run–length limited sequences are transmitted. In an experimental system, the bit error rate of a link was found to be markedly higher when a long m–sequence (with runs of the order of 30 bits) was used to generate the input data, compared with the results obtained using a short m–sequence (with runs of the order of 7 bits)[41]. This indicates that there is potential benefit in using run–length limited coding.

Since systematic error control, typically based on Reed–Solomon (RS) codes, is now employed for such systems, it is appropriate to devise arrangements that enable run–length limiting to be incorporated. In this particular context it is important to note that the data rates are very high (1 Gb/s or greater) so that any proposed code must be of relatively low complexity if it is to be practically realisable [42].

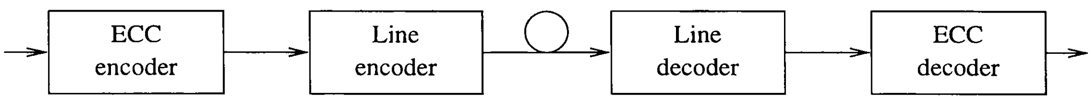

Figure 5.1: Conventional cascaded coding arrangement

A conventional coding arrangement used in such a system is shown in figure 5.1. However, this technique cannot be used with a soft decision decoder for the error control code, since its line decoding operation eliminates the analogue information from the channel. In such a system, a systematic line code has to be used so that the decoding consists of extracting the information bits retaining the analogue information. The disadvantage of this approach is that the maximum run is longer than what is possible with a non–systematic code.

Furthermore, this arrangement has the disadvantage that error extension is present from the line decoder before the error control decoder. Since a typical line code has no error control capabilities, any errors in the channel during the transmission of a code–word may result in a completely different code word after decoding. However, when RS coded data are used with a block decodable line code, the input size of the line code word can be selected to match the symbol size of the error control code. This avoids error extension due to the line code since it is restricted in only one n–bit symbol of the RS code [43]. The disadvantage of this approach is that the maximum rate of the line code that can be used is n/(n+1). If a higher–rate line code is used, the error extension of the line code will affect more symbols and in order to achieve the same residual bit error rate, the use of a more powerful RS code is required.

5.2 Overview

To combat those problems, several attempts have been made to invert the order of the error correcting and line codes. In such schemes the error correction takes place before the line decoding.

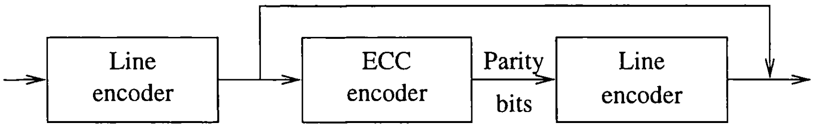

Figure 5.2: Bliss' coding scheme

In an attempt to solve the error extension problem, Bliss[44] proposed encoding the information with a line code before the error control encoding. Then, the parity information is encoded with a second line code. The encoding arrangement is shown in figure 5.2. Immink[45, 46] proposes a scheme that employs an added lossless compression stage before the error control encoding to improve the tolerance to burst errors. A comprehensive overview of such schemes with emphasis to applications in magnetic recording is presented in [47].

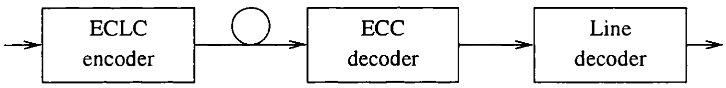

Figure 5.3: Error correcting line code
encoding and decoding arrangement

Furthermore, Popplewell[48, 49] proposed a class of codes called error control line codes (ECLC) where the code–words transmitted in the channel are simultaneously valid error correcting code–words, and line code–words. Therefore they can be decoded first by the error correcting code and then by the line code as shown on figure 5.3.

5.3 Proposed line code

In this chapter a new procedure which allows block decodable line codes which exhibit tighter run–length bounds to be used is presented[5]. This technique also utilises two cascaded codes with the line code encapsulating the error control code, allowing error control decoding before line decoding. Since the analogue value for each bit is available, soft decision error control decoding is possible. However, before the transmission of the coded sequence, an added step takes place. This step distributes a part of each error control code word in a predefined way that guarantees the limiting of the maximum run–length.

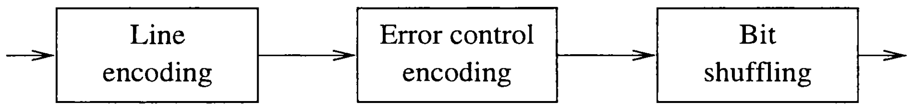

Figure 5.4: Proposed coding arrangement

This procedure requires that the error control code is a systematic one. Figure 5.4 shows the steps used during encoding.

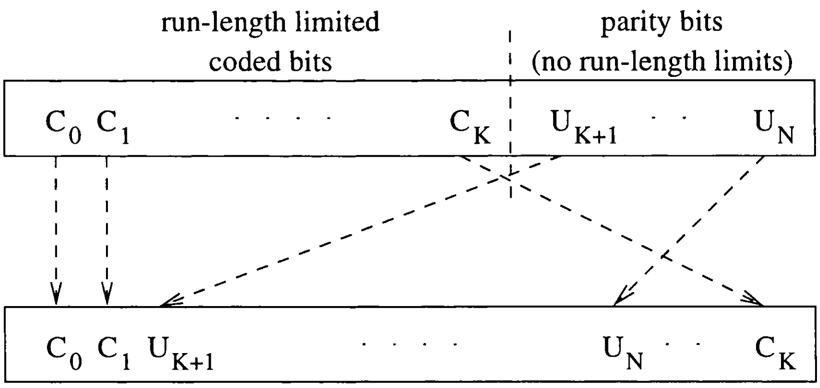

Figure 5.5: Shuffling of the bits to satisfy run–length bounds

During line encoding, the information symbols are encoded, generating a block of run–length limited data. This block is then encoded using an error control code. The effect is that part of the resulting code word is encoded with a run–length limited code, while the parity bits have no run–length limitations. To combat this we shuffle the bits so that the parity bits are distributed as single bits in appropriate positions in the line coded part. Such an example can be seen in figure 5.5. In the worst case, using a regular line code and providing any two parity bits have a sufficient number of run–length limited bits between them, this stage will increase the maximum run by one.

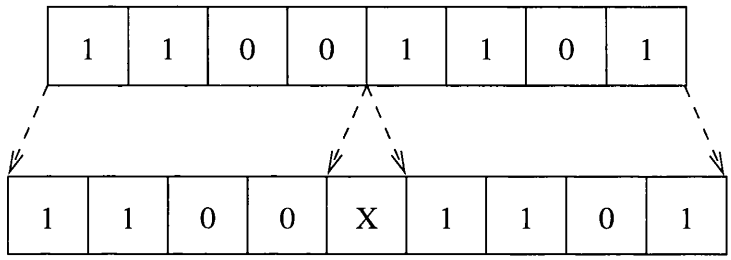

Figure 5.6: Example of bit insertion in a code–word

In figure 5.6, this increase of the run length is illustrated. One codeword of such a code is displayed. A 8–bit long code–word has one bit inserted in the middle. The maximum run–length in this code word was two before the insertion and it becomes three afterwards.

However, by correctly designing the line code, it is possible in most cases to incorporate the inserted bits within the maximum run–length. This can be achieved by designing the line code to have unconstrained bit positions where single bits can be inserted without affecting the maximum run[50]. Such a code has the property that it can be encoded and decoded independently of the value of those bits.

Furthermore, in certain error correcting and line code combinations, it is possible to leave some of the information symbols uncoded and to distribute them, together with the parity symbols, in available unconstrained bit positions of the run–length limited sequence. Wherever this is feasible, it results in increased overall rate.

This procedure allows the transmission of the uncoded control symbols with a small increase in the complexity of the coding system.

5.3.1 Implementation details

Block decodable line codes are commonly referred to as nBmB codes, with n the number of information bits and m the length of the code. This study is restricted to line codes with rate of the form $(m-1)/m$, i.e. $n = m - 1$.

As an example consider an RS(255,245) five error correcting code. For each code word 2040 bits are transmitted. With our procedure, 238 of the input symbols can be encoded using a 13B14B line code giving 136 line codewords, and leaving 136 bits uncoded (10 error control symbols and 7 information symbols) Those bits can be distributed one after each line code word, giving a sequence with a maximum run of 6 identical symbols.

If error control codes of a lower rate are to be used, then line codes with more unconstrained bit positions per code word are necessary. The error control code has K information bits encoded into N bits. We want the line code length m plus the number of unconstrained bit positions s to be a divisor of N so that the line code is aligned with the error control code, resulting in simplified decoding. Let $x = N/(m+s)$. Then for this technique to work,

$$x \times m < K \Rightarrow \frac{m}{m+s} < \frac{K}{N}$$

For each code word, the number of information bits encoded in the run–length limited sequence is $x \times (m-1)$, while the number of information bits left uncoded is $K - (x \times m)$. Therefore, the overall rate R of the transmitted sequence is

$$R = \frac{K - x}{N} \Rightarrow R = \frac{K}{N} - \frac{1}{m+s}$$

Therefore, this coding procedure reduces the rate by only $1/(m+s)$.

Code	Run length		
	$s = 0$	$s = 1$	$s = 2$
7B8B	4	5	5
8B9B	4	5	5
9B10B	5	5	5
10B11B	5	5	5
11B12B	5	5	5
12B13B	5	5	5
13B14B	5	5	5
14B15B	5	5	5
15B16B	5	5	6
16B17B	5	6	6

Table 5.1: Maximum run–length of various codes.

s is the number of unconstrained bit positions

Table 5.1 lists the maximum run–length of codes with zero, one and two unconstrained bit positions, obtained by employing an exhaustive search algorithm.

This table illustrates the cases where using codes with unconstrained bits gives a smaller run length compared with just inserting the bits in a regular run length limited code. For example, if a normal 10B11B run length limited line code is selected, the maximum run length is 5. If one bit is inserted in the middle of the code–word (or possibly between successive code–words), then the maximum run length becomes 6, and the code is equivalent to an 11B12B code with one unconstrained bit position. However, an 11B12B code with one unconstrained bit can have a maximum run length of 5.

Code	Run length	
	$s = 1$	$s = 2$
10B11B	5	
11B12B	5	5
12B13B	5	5
13B14B	5	5
14B15B	5	5
15B16B	5	

Table 5.2: Codes of table 5.1 presenting an improvement in run–length. s is the number of unconstrained bit positions

Table 5.2 presents the cases where such improvements are possible. However, even in the cases where there is no apparent improvement in run length, there is likely to be an improvement in terms of hardware implementation, since the new code will have more relaxed run–length constraints than would a normal run–length limited code.

5.4 Implementing a line code

One problem encountered during the design of a line code is the selection of code–words from those available, as well as the problem of assigning the code-words to input words. This section presents a technique that was developed to aid this assignment.

The main characteristic that was required of the designed code was implementation simplicity. This translates to low hardware requirements for the encoder and the decoder. Another method with similar aims but for a different class of line codes is described in [50, 51].

A first approach to the problem was to use Karnaugh maps. The input bits were to be the input to a logic function that was to be designed, and one map (function) was to be used for each bit of the output codeword, for a random selection of codewords. Then an optimisation technique was to be used to modify the Karnaugh maps by replacing the selected code–words with others until a satisfactory function for each bit was achieved. This approach posed several difficulties in implementation and was soon abandoned. However, it was helpful in obtaining the following insight:

Obviously, for a logic function to be easy to implement, many neighbouring positions in the Karnaugh map have to have the same state (zero or one). Since the inputs of the function are the input words, this is roughly equivalent to input words that have small Hamming distance giving the same output. And a suitable requirement over the whole of the output code word may be that input words with Hamming distance equal to one correspond to output code words that have small Hamming distance between them. Therefore, the cost function C used was

$$C = \sum_{k.l} d_H(w(k), w(l)) \quad | \quad d_H(k,l) = 1,$$

where k and l are input words, while $w(k)$ and $w(l)$ are the corresponding code–words.

This requirement was used in a computer programme that used simulated annealing to minimise the average distance of the output code–words that correspond to all pairs of input words that have Hamming distance one between them. Simulated annealing is a global optimisation technique that was proposed by Metropolis[52] and is considered a good choice for a wide variety of problems. It attempts to minimise a cost function C by doing a random change in the initial configuration and then accepting the change if the difference $\Delta = C_1 - C_0$ of the new cost C_1 and the old cost C_0 is negative, or with probability

$$p = e^{-\frac{\Delta}{T}}, \quad \Delta > 0$$

where T is called the "temperature", and is progressively reduced. This way the probability of accepting a new configuration that increases the cost function is reduced as the optimisation progresses, while smaller increases are always more likely to be accepted than larger ones.

In channel coding, simulated annealing has been used to obtain large error correcting codes[53] and constant weight codes[28]. However, stochastic optimisation techniques are not known to have been used in the design of line codes.

Several other more complicated cost functions were used, that did not result in better performance. Those include cost functions where the difference

$$C = \sum_{k,l} d_H(w(k), w(l)) - d_H(k, l)$$

or the ratio

$$C = \sum_{k,l} \frac{d_H(w(k), w(l))}{d_H(k, l)}$$

of the Hamming distance of the output codewords with the Hamming distance of the input words were used. Also the squared value of the Hamming distance, as well as the Hamming distance raised to a higher power

$$C = \sum_{k,l} (d_H(w(k), w(l)))^n \quad | \quad d_H(k, l) = 1, \quad n \geq 2$$

were considered in an effort to penalise higher values. However, those cost functions required increased computation, and the resulting codes presented no advantages compared to those obtained with the original cost function, while many of them required a significantly more complex implementation.

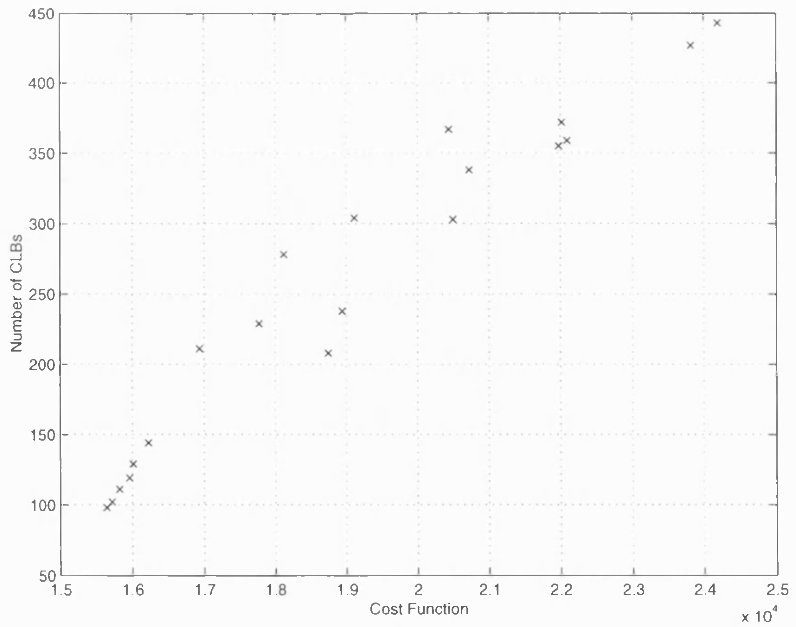

Figure 5.7: Number of configurable logic blocks for different values of the cost function

Figure 5.7 displays the hardware complexity of a line code as a function of the value of the cost function. Each of these is the lowest value obtained from several optimisation trials. The hardware complexity in this case is represented by the number of configurable logic blocks (CLBs) of a Xilinx field programmable gate array (FPGA)[54] that were required. This number was obtained using the SIS design software package[55]. The code used in this example is a simple run length limited code, that had 11 input bits, a 12 bit codeword and a maximum run of 5 consecutive identical symbols.

In this graph the relationship between the cost function and the resulting code complexity can be seen. Lower cost functions give, in most cases, codes of lower complexity. This is always the case with lower cost values, while some discrepancies occur at higher cost values. This may be explained by the increased difficulty of obtaining a good hardware implementation using SIS or any other computer design software when the requested functions become more complex.

5.3.2 Improvement

A further refinement of the technique is the following. The original optimisation algorithm uses only the Hamming distance of the code words as a measure. This in general required one logic function for each input bit. It is possible to further reduce the hardware requirements if we arrange for some of the input bits to be present directly in the output.

Table 5.3 shows the number of input bits that can be present uncoded in the code–word without sacrificing the maximum run constraints. Of course, the encoding of the remaining bits of the codeword are dependent on all bits of the

Code	Uncoded bits
7B8B	3
8B9B	2
9B10B	5
10B11B	4
11B12B	5
12B13B	5
13B14B	6
14B15B	6
15B16B	5

Table 5.3: Maximum number of uncoded bits that can be present in line code

input word. This is different from the unconstrained bits that were mentioned earlier, whose values do not affect the remainder of the code–word.

A modified version of the algorithm was designed that, prior to optimisation, selected from all the possible code–words the subset into which the highest possible number of input bits could be used as output bits. Then the optimisation function was slightly modified to preserve those relationships. The end result was codes that required fewer CLBs compared with codes that had the same cost value.

Figure 5.8 shows the results obtained for this modified cost function for several optimisation trials. Again there is a clear overall trend for better results with lower values of the cost function. However, this graph is more erratic than the one using the original method. One possible explanation may be that again the demands on the logic minimisation software were increased, since the resulting

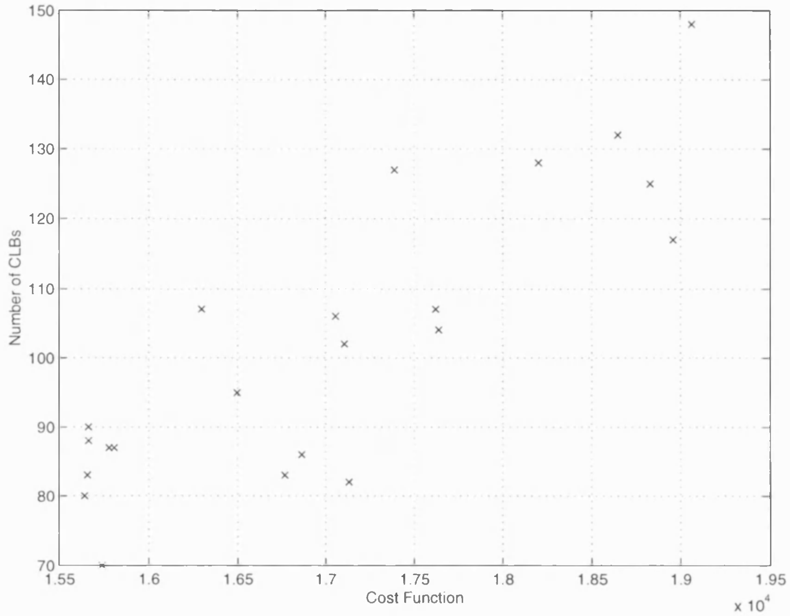

Figure 5.8: Number of configurable logic blocks
for different values of the modified cost function

functions were more complicated compared to the previous case, even though the overall result is better since there are fewer of those functions. Furthermore, it is clear that this method gives improved results compared to the original one.

Using the original approach, the best result was 98 CLBs, giving an average of $8\frac{2}{12}$ CLBs per output bit. With the improved approach, 5 of the input bits are left uncoded in the output code-word, leaving only 7 out of the 12 output bits encoded. In this case, the best result is 70 CLBs, giving an overage of 10 CLBs per output bit. This supports the above statement that each output bit is encoded with a more complicated function, but the overall complexity is lower.

5.3.3 Error extension

Another consideration in many line codes is the error extension present in the code. This is the number of errors present in the output after the decoding of a codeword with a single error. Since most line codes have no error correcting capabilities, this error extension can take large values. The above technique can also be used to reduce error extension. A modified cost function is required for this case. The cost function tested was the sum of the Hamming distances of all pairs of input words whose corresponding code–words have a Hamming distance of one. Minimising such a cost function gives a code where single errors in the channel decode to code–words similar to the transmitted ones. This is effectively the inverse of the previous cost function. However, this cost function minimises the average error extension. If the maximum error extension needs to be minimised, then the Hamming distance of the codewords that are to be summed can be raised to a power, so that the optimisation process will favour smaller values to a greater extent.

Only code–words that differ in one bit were considered, since the error extension is more pronounced in this case.

Finally, both cost functions can be used together to give a code with a reasonable hardware implementation, while keeping the error extension of the code low.

5.4 Examples of designed codes

Using the proposed technique to design codes with a single unconstrained bit position we get the following results:

Table 5.4 shows the number of bits that can be present uncoded at the output code–word of a line code that incorporates one unconstrained bit position. This differs from table 5.3 since the presence of the unconstrained position restricts the number of available–codewords, and in certain cases increases the maximum

Code	Uncoded bits
6B7B	4
7B8B	5
8B9B	5
9B10B	4
10B11B	5
11B12B	5
12B13B	5
13B14B	5
14B15B	4

Table 5.4: Uncoded bits for various codes including the unconstrained one

run length. Since the unconstrained bit is not part of the code as such, the entries in table 5.4 are one bit shorter than the equivalent entries in table 5.3. however, in order to compare the number of uncoded bits, the unconstrained bit is added to the uncoded bits.

Figure 5.9 shows the number of Xilinx 3000 CLBs that were required to implement line codes with one unconstrained bit position, for several different code–word lengths. Clearly, the complexity increases rapidly with the number of code–word bits, the number of CLBs roughly doubling with each added bit. This provides an indication that the procedure can scale to even longer codes.

5.5 Summary

A new coding scheme has been described that allows the use of block codes with a simple line code that gives tight run–length bounds with a small decrease in

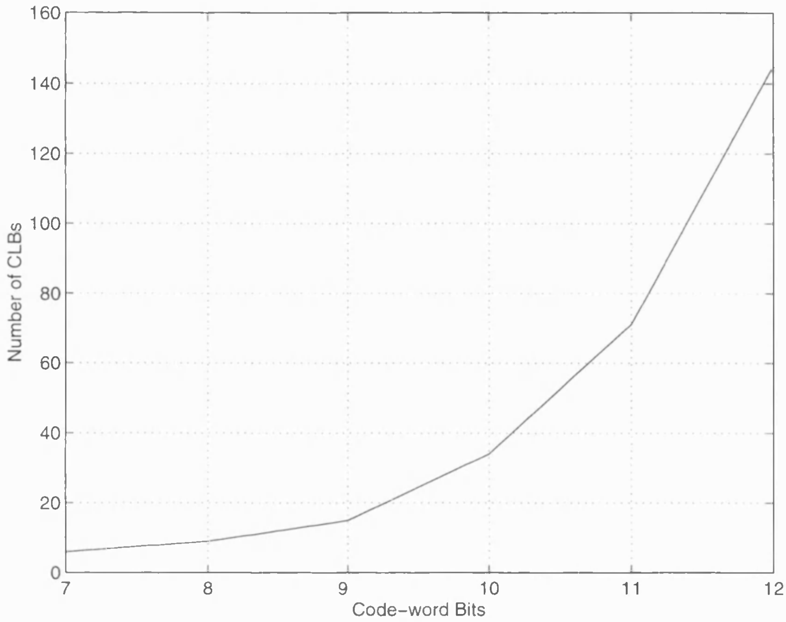

Figure 5.9: Number of CLBs for different code–word lengths

the overall code rate and without limiting the decoding power of the error control code. This was achieved by reversing the usual cascaded error control and line codes, so that the error control code is the outer one, in such a way that the properties of the line code are retained.

Furthermore, a stochastic optimisation technique was employed to select the source to channel code–words mapping in an efficient way, in order to implement those codes. Clearly, better optimised codes required less complex hardware implementations than unoptimised ones. Several different cost functions were investigated and the most promising one was identified.

Finally, several line codes suitable for use in the cascaded structure described above were designed and their relative hardware complexity was assessed.

In the following chapter a departure from the traditional line coding constructions is addressed; line codes are designed that can constrain the peak to average power ratio of a multi–carrier communication system.

6 Peak power constrained coding for multi–carrier systems

6.1 Introduction

Multi–carrier modulation is the technique where the information is divided into several parallel data streams that are transmitted simultaneously using several sub–carriers.

One such technique is orthogonal frequency division multiplexing (OFDM) that involves several sub–carriers with overlapping modulation spectra. The waveforms used are selected in such a way as to guarantee the orthogonality of the sub–carriers and can be generated using fast Fourier transforms (FFTs) at the transmitter and the receiver.

One problem of such multi–carrier techniques is that the peak transmitted power can be many times larger than the average power, giving a large peak to average power ratio (PAPR).

Increased interest in multi–carrier transmission has resulted in a number of solutions being proposed for the reduction of peak transmitted power. A recent paper by Jones *et al.* [56] offers a simple block coding scheme for power reduction, where the block coding is applied across the carriers. This code uses only the code–words that would give the lower values of the PAPR, thereby giving very good results. In the four carrier case, this can be achieved by using three information bits with an added parity bit, giving a very simple solution. For a larger number of carriers, though, no such simple solutions were found, and the only alternative proposed is to use look–up tables, with all that entails in terms of complexity.

Because of the perceived complexity of this scheme, several other techniques have been proposed; these achieve lower complexity, but yield inferior results in terms of PAPR reduction and code rate. In this chapter, new codes that achieve acceptable reduction in the PAPR value without requiring the complexity of a large look–up table are developed.

6.2 Multi–carrier modulation

A bandpass signal $v(t)$ is commonly represented by the following equation

$$v(t) = x(t)\cos(2\pi f_c t) - y(t)\sin(2\pi f_c t),$$

where f_c is the carrier frequency, while $x(t)$ is the in–phase and $y(t)$ is the quadrature part of $v(t)$.

An equivalent representation of $v(t)$ is

$$v(t) = \Re(g(t)e^{j2\pi f_c t}),$$

where $\Re(x)$ is the real part of x, and $g(t)$ is the complex envelope of $v(t)$. The complex envelope can be given as a function of the in–phase and quadrature parts of $v(t)$ by

$$g(t) = x(t) + jy(t),$$

where $x(t)$ and $y(t)$ are real baseband waveforms, while $g(t)$ is obviously a complex baseband waveform.

For a multi–carrier signal with N sub–carriers,

$$v(t) = \sum_{k=0}^{N-1} x_k(t)\cos(2\pi(f_c + \frac{k}{T})t) - y_k(t)\sin(2\pi(f_c + \frac{k}{T})t),$$

where T is the symbol period, and $x_k(t)$, $y_k(t)$ are the in–phase and quadrature parts of the k–th sub–carrier. The complex envelope representation is

$$v(t) = \frac{1}{\sqrt{N}} \Re(\sum_{k=0}^{N-1} g_k(t) e^{\frac{j2\pi kt}{T}} e^{j2\pi f_c t}),$$

where now $g_k(t)$ is the complex envelope of the k–th sub–carrier, and as before $g_k(t) = x_k(t) + jy_k(t)$.

The sum of the several sub–carriers can be considered as a single baseband signal, giving

$$g(t) = \sum_{k=0}^{N-1} g_k(t) e^{\frac{j2\pi kt}{T}}.$$

For the transmission of digital signals, both $x_k(t)$ and $y_k(t)$ take values from the discrete set of possible values that are held constant for the duration of the symbol period. The discrete complex envelope for the k–th sub–carrier will be identified as $d_k(t)$. Obviously, $d_k(t) = x_k(t) + jy_k(t)$. Then, the complex envelope of an N sub–carrier system is

$$g(t) = \sum_{k=0}^{N-1} d_k(t) e^{\frac{j2\pi kt}{T} + \phi_k}$$

where ϕ_k is the initial phase of the k–th sub–carrier. The initial phase of each sub–carrier is explicitly mentioned in this expression since, for all sub–carriers, the $g_k(t)$ are limited to values from the same discrete set, while under the original formulation the initial phase was incorporated in the value of $g_k(t)$. Moreover, for simplicity the initial phase ϕ_k of each sub–carrier is assumed to be equal to zero for the rest of the discussion.

6.3 Peak to average power ratio

Since $x(t)$ and $y(t)$ are baseband waveforms, it is reasonable to assume that they are constant over one period of the carrier signal. Therefore, the instantaneous envelope power of $v(t)$ is given by

$$P(t) = f_c \int_t^{t+\frac{1}{f_c}} v^2(\tau) d\tau =$$

$$= f_c \int_t^{t+\frac{1}{f_c}} \left(x(t)\cos(2\pi f_c \tau) - y(t)\sin(2\pi f_c \tau)\right)^2 d\tau =$$

$$= \frac{x^2(t) + y^2(t)}{2} = \frac{g(t)g^*(t)}{2}$$

The peak envelope power (PEP) is defined as "the average power that would be obtained if $|g(t)|$ were to be held constant at its peak value"[57]. Therefore, in our case,

$$P_{PEP} = \frac{\max(g(t)g^*(t))}{2},$$

where T is the duration we are interested in.

The peak to average power ratio (PAPR) (sometimes referred to as peak factor[58]) is given by

$$PAPR = \frac{\max v^2(t)}{E(v^2(t))} = \frac{\frac{1}{2}\max(g(t)g^*(t))}{E(g(t)g^*(t))},$$

where $E(x)$ is the expected value of x. Sometimes a different quantity, called the crest factor, is used. The crest factor is defined as

$$CF = \frac{\max |v(t)|}{\sqrt{E(v^2(t))}},$$

which is the square root of the PAPR.

The case we are interested in is the transmission of digital data. Therefore, for each period only a limited sum of different waveforms is possible. For the study of coding schemes that can be used to affect the PAPR, only one such period is studied neglecting the effects of the transition to a different waveform. The peak power can be calculated by either sampling at a high frequency the multi–carrier signal, or by using the property that the peak value of a continuous function $s(t)$ in the interval a to b is given by

$$\lim_{n\to\infty} \sqrt[n]{\frac{1}{b-a}\int_a^b |s(t)|^n dt}$$

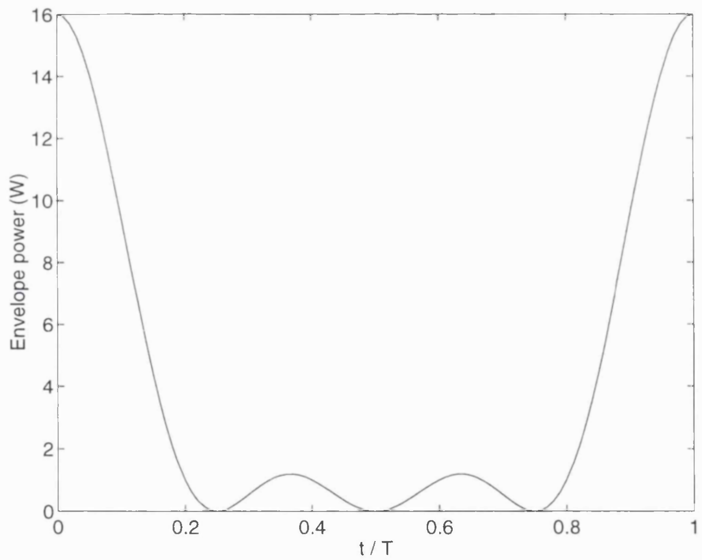

Figure 6.1: Envelope power of a four sub–carrier signal

Assuming that each sub–carrier has a power normalised to 1 Watt, then the average power is equal to N Watts. However, when all combinations of inputs are allowed, the peak value of the power is N^2 Watts. This can be seen in figure 6.1

where the envelope power of the four sub–carrier signal $\sqrt{2}\sum_{k=0}^{3} e^{2\pi \frac{k}{T} t}$ is plotted over one period. The PAPR in such a case has a value of N. This poses several problems, mainly that an amplifier that is linear up to this power is required, if the intermodulation distortion must be kept low, adding significantly to the cost of the system.

To avoid these problems, several authors have proposed to modify the initial phase of each carrier [59, 60, 61, 62] in several ways that avoid the large values of PAPR. However, most of this work is not concerned with modulation. When the carriers are independently phase modulated such phasing schemes become ineffective[63] because of the carriers' uncorrelated phases.

Another technique that has been proposed is to vary the level of all carriers in such a way that the peak transmitted power for every combination of inputs remains the same[58]. This method can limit the peak power to any value. However, the resulting bit error rate is higher than that of a standard modulation scheme.

A more promising idea is to add some redundancy in the transmitted sequence in such a way as to avoid the input combinations that give large values to the PAPR[56, 64, 65]. Several such systems have been proposed.

One that gives very good results in both code rate and PAPR reduction is to only transmit the better half of the possible input combinations, which requires only one bit of redundancy. However, only for a four carrier system is there a simple implementation; for higher values of N the use of lookup tables is proposed.

Several other schemes have been proposed that trade PAPR reduction or code rate for implementation simplicity[66, 67, 68, 69, 70, 10]. Furthermore, several attempts have been made to combine the PAPR reduction with error correction, for example by using Golay sequences and Reed–Muller codes[71] or m–sequences[72, 73].

6.4 Encoder structure

This chapter focuses on the tradeoff between PAPR reduction and implementation complexity for encoding schemes. This section outlines the encoder structure that will be used.

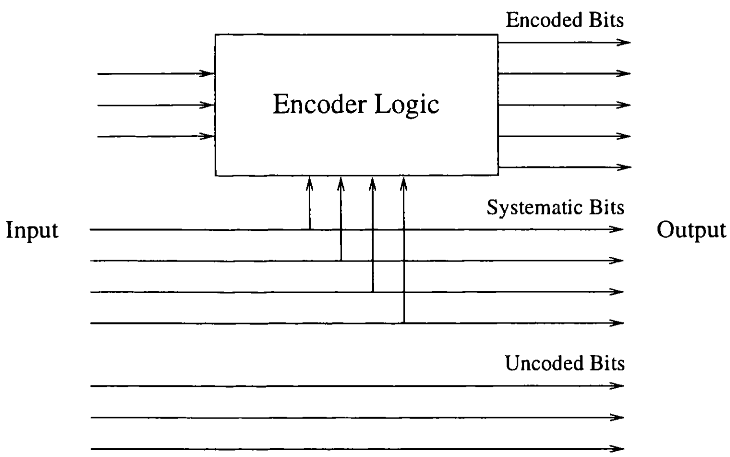

Figure 6.2: Encoder structure.

This encoder structure is shown on figure 6.2. Here, the encoder output can be divided into three parts: the encoded bits, the systematic bits and the uncoded bits. The uncoded bits are the output bits that are passed from the input to the output without affecting the encoding process in any way. The encoded bits are the ones whose value depends on a function of several input bits. Finally, the systematic bits are those that are present in the output in the same way as in the input, while their value affects the encoding of some (or all) of the encoded bits. The figure shows an example of a ten input to twelve output decoder. The five encoded bits, depend on the value of seven of the input bits, including the four systematic bits. This leaves three uncoded bits. Of course, even though these bits are shown in order in the diagram, depending on the actual code their order may be different.

The number of encoded and systematic bits is used as a measure of the implementation complexity of the encoder logic.

The remainder of this chapter will focus on the tradeoff between PAPR reduction and implementation complexity for various modulation schemes, using for the most part just one single bit of redundancy.

6.5 Binary phase shift keying (BPSK)

Using this modulation method only one bit of information per sub–carrier is transmitted at each period. The two different values are represented by a 180 degree phase shift of the sub–carrier. The modulation data d_k take the values from the set $\{\sqrt{2}, -\sqrt{2}\}$ when each carrier's power is normalised to 1 Watt. The resulting complex envelope is of the form

$$g(t) = \sqrt{2} \sum_{k=0}^{N-1} \pm e^{\frac{j2\pi k t}{T}}.$$

Using Jones' scheme to achieve a good reduction of the PAPR, only half of the available combinations are used, thereby effectively using one redundant bit. This is achieved by using a block code that encodes $N-1$ bits to an N–bit

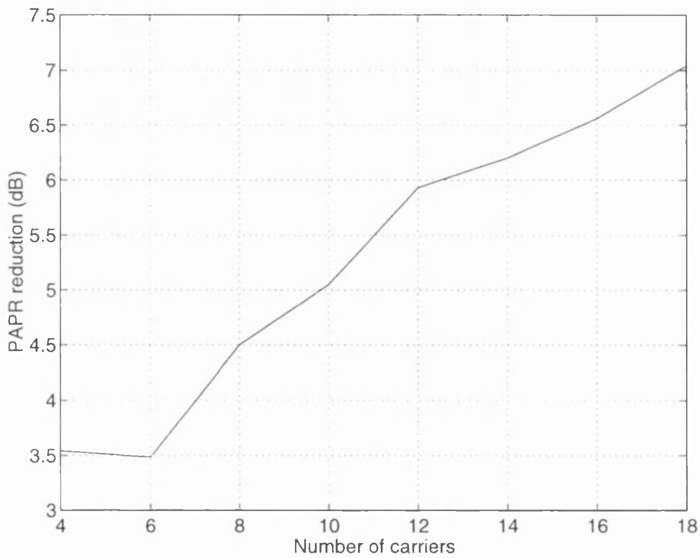

Figure 6.3: PAPR reduction possible with Jones' scheme

code–word, where each bit is used to modulate one of the sub–carriers. The PAPR reduction that can be achieved using this scheme is shown on figure 6.3. Obviously, this scheme results in a very low overhead, while obtaining the best possible improvement for the specified overhead. However, the only case where such a technique is also simple to use is the 4–carrier system, where the required code is three bits with an added parity check bit. For more complicated codes, the best proposed alternative is to use a look–up table for the encoding and decoding.

However, we shall show that if one is willing to sacrifice some of the improvement of the PAPR, then simpler codes are possible without any reduction in the code rate. The number of encoded bits required to achieve the best result is shown on table 6.1. In most of the cases, to achieve the best possible result it is necessary to encode and decode most of the bits. However, in the 4–carrier system only one single bit needs to be encoded while the other three are systematic, and

Carriers	Encoded Bits	Gain(dB)
4	1	3.54
6	2	3.48
8	4	4.50
10	8	5.05
12	8	5.93
14	12	6.20
16	14	6.56
18	12	7.04

Table 6.1: Number of encoded bits required for Jones' scheme

the decoding can be achieved by just ignoring the value of the encoded bit. Similarly, for a larger number of carriers it is possible to design codes that need less bits to be encoded and decoded.

Furthermore, it is possible to design codes that have the property that the encoding and decoding of the coded bits is independent of the value of some uncoded bits (as shown in figure 6.2). However, this is not always the case. As an example, in the 4–bit code discussed earlier, the encoding of the fourth bit (*i.e.* the parity bit) depends on the value of all of the three systematic bits.

The simplest case is where one bit is encoded and its value is dependent on only one other systematic bit. In most cases this can be achieved by setting the added bit at the end of the code–word to the inverse of the second to last information bit[10]. The decoding can be as simple as ignoring the encoded bit, or its value can be used to improve the reliability of the information of the systematic bit.

Following the analysis in [64], the substitution $z = e^{\frac{j2\pi}{T}}$ results in

$$g(t) = \sqrt{2} \sum_{k=0}^{N-1} d_k z^{kt}$$

where d_k is the signal modulated on the k–th subcarrier and in this case takes values from the set $\{-1, 1\}$. The transformations of multiplication by -1, or of substitution of z with $-z$, z^{-1} and $-z^{-1}$ give other sequences with the same peak value. Multiplication by -1 is equivalent to inverting the code–word. Substitution of z with $-z$ is equivalent to inverting the signal of the odd sub–carriers. Finally, substituting z with z^{-1} is equivalent to reversing the order of the sub–carriers. However, not all combinations of those transformations correspond to distinct values of the information sequence, but there will be at least three that will. Therefore, for each code–word, there will be another three and possibly more combinations that give the same peak value.

Carriers	Gain(dB)	Possible(dB)
4	2.27	3.54
6	2.67	3.26
8	2.50	2.50
10	1.94	1.94
12	1.58	1.58
14	1.34	1.34
16	1.16	1.16
18	1.02	1.02

Table 6.2: PAPR gain possible by using simple code compared with best possible gain using comparable scheme

One of the code–words that gives the worst PAPR values is the one where $d_k = 1$ for all k. Using combinations of the above transformations we get another three code–words that give the same peak value. These are the $d_k = -1$ for all k and the $d_k = 1$ for k odd (even) and $d_k = -1$ for k even (odd). It turns out that there are no other code–words that give the same peak value. Those four code–words can be avoided with the simple code described above. For 8 carriers and more, this code gives the best result that is achievable when only one bit is encoded. The improvement possible is shown in table 6.2 compared to the best possible improvement when only a single bit is encoded. However, for a large number of carriers the possible PAPR improvement becomes insignificant[74].

6.5.1 Search algorithms

To obtain the number of bits that can be systematic or uncoded while the required PAPR improvement is achieved for a given code size and rate, two exhaustive search procedures were used. Initially, the subset of the possible code–words whose peak power is below the required maximum is selected. The first search algorithm is used to find the maximum number of uncoded bits, while the second one is used to find the maximum number of systematic bits possible for such a code. Moreover, the two algorithms can be cascaded to find the value of the sum of the number of uncoded and systematic bits for this code.

The first algorithm searches for the maximum possible number of uncoded bits. The binary tree representing all possible combinations when every bit is either considered or not is traversed. For each bit under consideration, all pairs of codewords that differ only in that bit position are selected and only the code–word with zero at that bit position is retained. The procedure is repeated with the selected words until all bit positions are examined, or until the code rate requirements can not be satisfied. The combination with the maximum number of bits is the final result.

The second algorithm searches for the maximum possible number of systematic bits. Again, the binary tree representing all possible combinations when every bit is either considered or not is traversed. For each bit under consideration, the code–words are subdivided into two sets depending on the value of that bit. These sets are further subdivided for every other bit explored until all the bits are examined or the code rate requirements can not be satisfied. The combination with the maximum number of bits is the final result.

6.5.2 Results

Figures 6.4 to 6.7 show the number of bits that need to be coded as a function of the PAPR improvement for several values of N. Each graph has three lines. The bottom solid line shows the maximum number of uncoded bits possible for the specified gain. The top solid line shows similarly the maximum number of systematic bits attainable. Finally, the dotted trace in the middle shows the sum of systematic and uncoded bits when the number of uncoded bits has its maximum feasible value (bottom trace).

From these figures the requirements for encoding and decoding hardware can be obtained. As an example consider the case where a 5dB gain in PAPR is required from a 16 carrier system. The best options can be seen in figure 6.7. From that diagram there is a choice of two codes. One is where 9 bits can be systematic, leaving the remaining 7 bits to be encoded as a function of most of the fifteen input bits, and one is where three of the input bits can be left uncoded while another five bits are systematic for a total of 8. This leaves

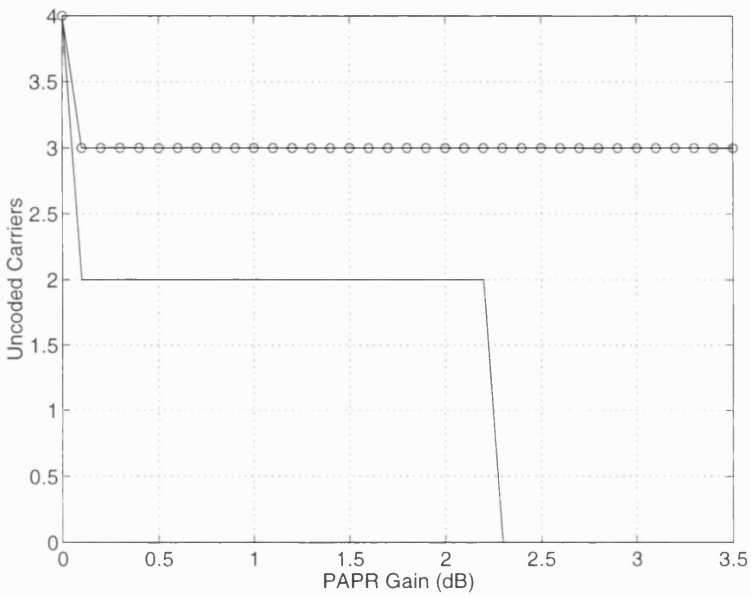

Figure 6.4: Uncoded and systematic bits as a function of the PAPR gain for a 4 sub–carrier system

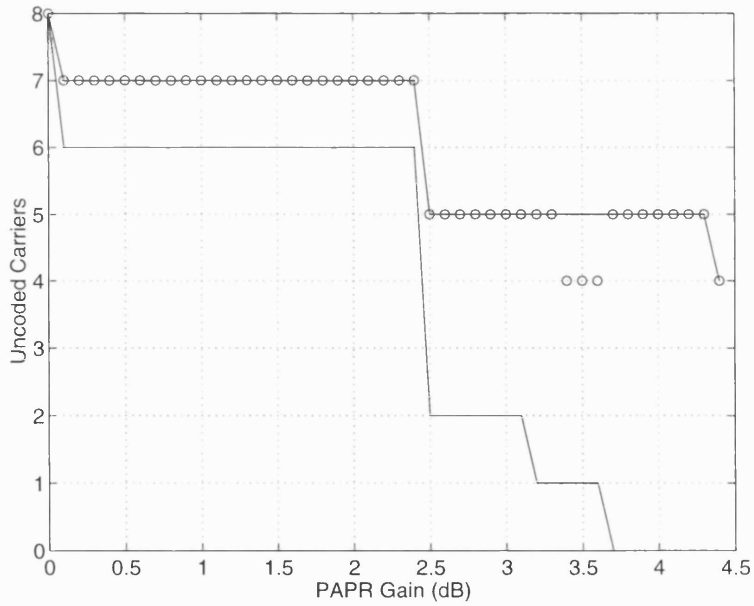

Figure 6.5: Uncoded and systematic bits as a function of the PAPR gain for an 8 sub–carrier system

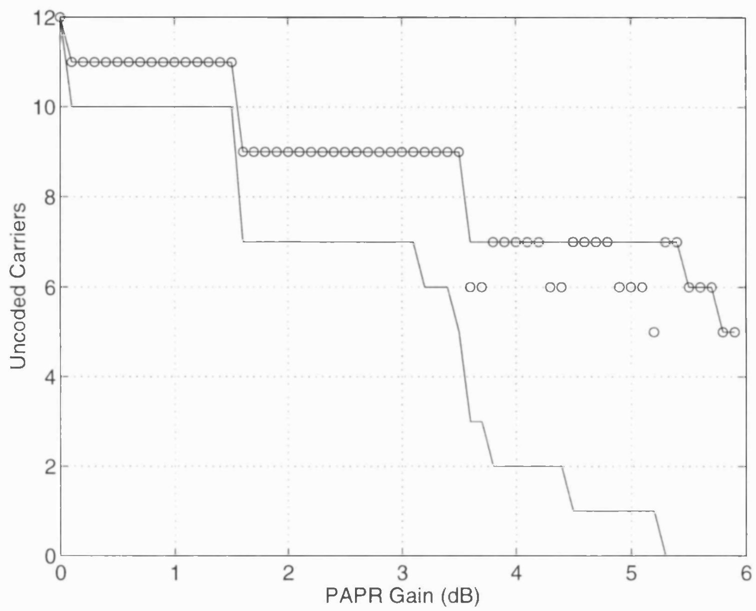

Figure 6.6: Uncoded and systematic bits as a function of the PAPR gain for a 12 sub–carrier system

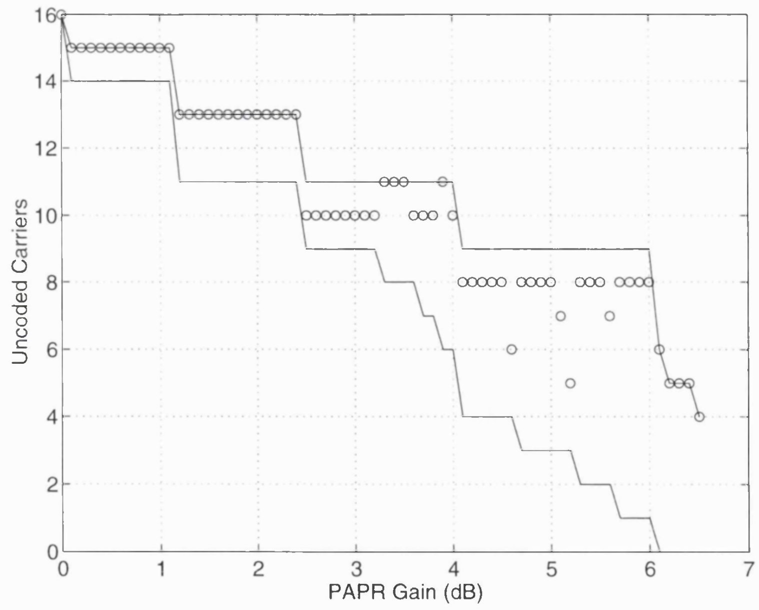

Figure 6.7: Uncoded and systematic bits as a function of the PAPR gain for a 16 sub–carrier system

another 8 bits to be encoded as a function of 12 bits. Selecting the last code, the requirements are for a 12–bit to 8–bit encoder and for a matching 13–bit to 7–bit decoder. This is compared to the 15–bit to 14–bit encoder and 16–bit to 13–bit decoder required for the 6.56dB improvement when using Jones' scheme.

Carriers	Encoder	Decoder
4	3→1	–
6	3→1	–
8	5→3	6→2
10	5→3	6→2
12	4→3	5→2
14	6→6	7→5
16	6→6	7→5
18	6→6	7→5

Table 6.3: Encoder and decoder arrangement for 3dB gain

Table 6.3 shows the number of inputs and outputs required for the encoder and decoder of codes that achieve a 3dB reduction in the PAPR for several carrier numbers. These codes use the maximum possible number of uncoded bits, as well as the maximum number of systematic bits available out of the remaining ones. The number of inputs and outputs is depicted in the table as *inputs→outputs*.

6.6 Quadrature phase shift keying (QPSK)

With a QPSK modulation scheme, two bits of information can be transmitted per carrier in one period. The modulation data d_n take values from the set

$\{1+j, 1-j, -1+j, -1-j\}$ when each sub–carrier's power is normalised to 1 Watt. The resulting complex envelope is of the form

$$g(t) = \sum_{k=0}^{N-1} (\pm 1 \pm j) e^{\frac{j2\pi kt}{T}}.$$

The substitution $z = e^{\frac{j2\pi}{T}}$, results in

$$g(t) = \sum_{k=0}^{N-1} d_k z^{kt},$$

where $d_k = \pm 1 \pm j$. According to [64], there are three sets of transformations of $g(t)$ that preserve the peak value. The first one consists of the multiplication by -1, by j or by $-j$. The second set consists of substitution of z by $-z$, by jz or by $-jz$. The third set consists of substitution of z by z^{-1}, by jz^{-1}, by $-z^{-1}$ or by $-jz^{-1}$. By combining transformations from those three groups, the equivalence of several input code–words with respect to the peak value can be shown.

Carriers	Gain(dB)	Possible(dB)
4	–	0.82
5	0.65	0.88
6	0.62	0.76
7	0.66	0.83
8	0.65	0.79
9	0.67	0.75
10	0.67	0.82

Table 6.4: PAPR gain possible by using simple code compared to the best possible gain using comparable code (QPSK)

There is a simple code that can avoid the input combinations that give the worst value of PAPR. In this case the code consists of adding at the end of the word a single encoded bit which is the inverse of the bit that is seventh from the last. This is so because due to the code–word equivalence, any other code that adds a bit that is a function of a single bit closer to the end will give a code–word that exhibits the worst case peak to average power ratio. The gain that can be achieved with this code is shown in table 6.4 compared to the best PAPR gain achievable with one single added bit.

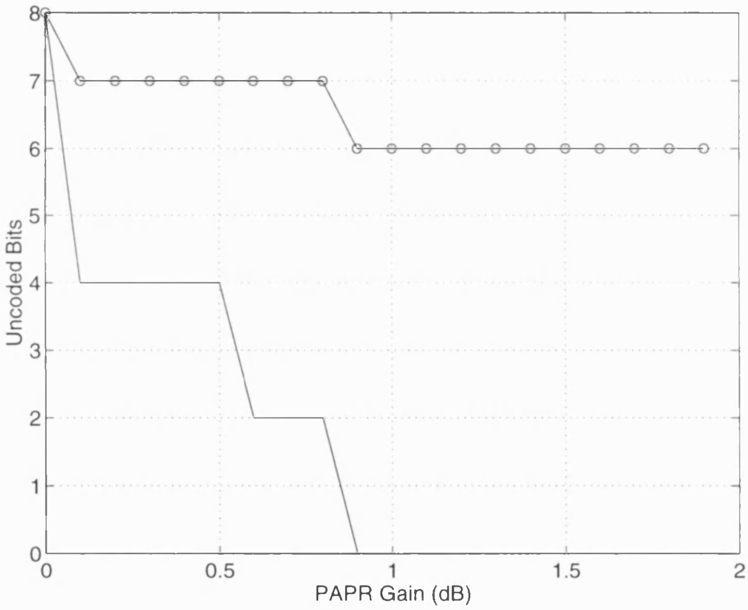

Figure 6.8: Uncoded and systematic bits as a function of the PAPR gain for a 4 sub–carrier QPSK system

Figures 6.8 to 6.10 show the number of bits that need to be coded as a function of the PAPR improvement for several values of N. As in the previous section, each graph has three lines. The bottom solid line shows the maximum number of uncoded bits possible for the specified gain. The top solid line shows similarly

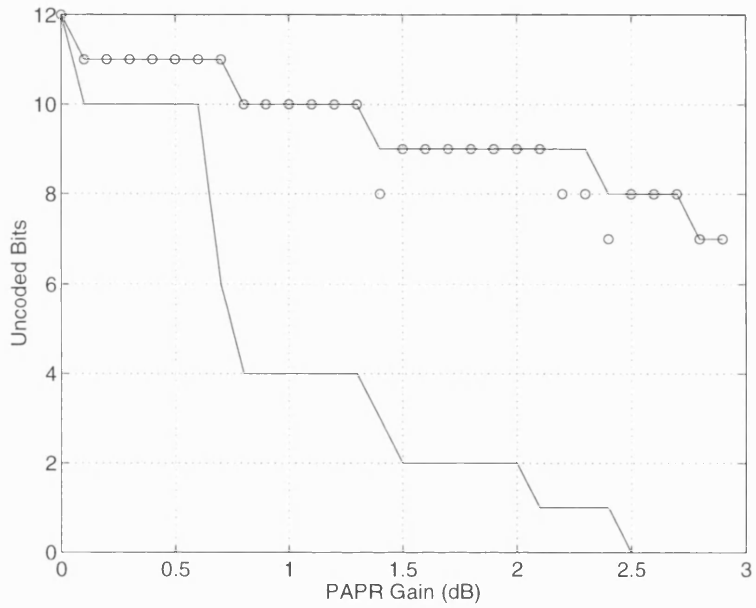

Figure 6.9: Uncoded and systematic bits as a function of the PAPR gain for a 6 sub–carrier QPSK system

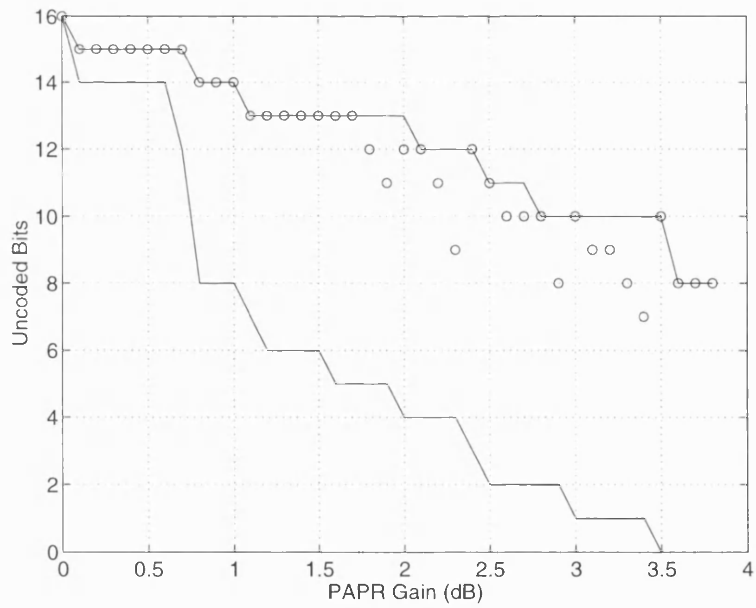

Figure 6.10: Uncoded and systematic bits as a function of the PAPR gain for an 8 sub–carrier QPSK system

the maximum number of systematic bits attainable. The dotted trace in the middle shows the maximum value of the sum of systematic and uncoded bits when the number of uncoded bits reaches its maximum attainable value (bottom trace).

As an example, the "Possible" column in table 6.4 can be obtained from such diagrams by noting the PAPR gain where the leftmost segment of the top line ends. This is the point where two encoded bits are required instead of one, while the rest are systematic.

Carriers	Encoder	Decoder
5	9→3	10→2
6	9→3	10→2
7	10→4	11→3
8	11→4	12→3
9	11→6	12→5

Table 6.5: Encoder and decoder arrangement for 2dB gain

Table 6.5 shows the encoder and decoder arrangements for codes that result in a 2dB gain compared to the uncoded system.

A different approach that requires added redundancy is to code one carrier (2 bits) in order to minimise the value of the PAPR. This is useful when each sub–carrier is modulated independently. In such a case, the encoder has no control over the individual carriers, therefore one feasible solution is to add an additional carrier to reduce the peak value.

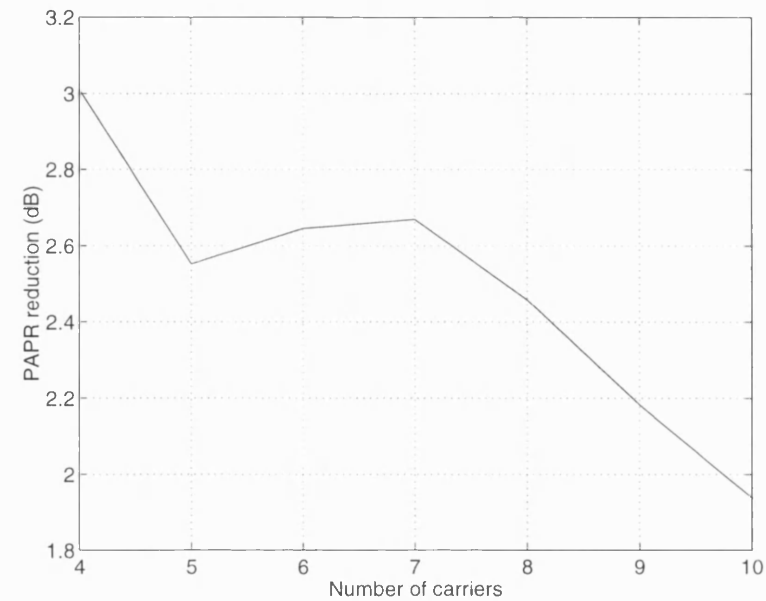

Figure 6.11: Possible PAPR gain for one redundant carrier (QPSK)

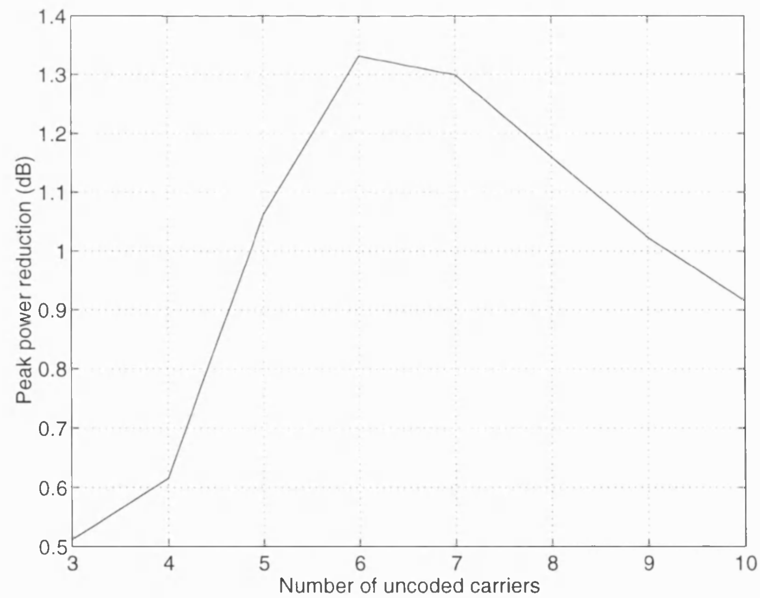

Figure 6.12: Peak power reduction with added redundant carrier (QPSK)

Figure 6.11 displays the possible PAPR improvement when the last carrier is used to add the appropriate redundancy. Figure 6.12 shows the peak power reduction, compared to the power of a system with one less carrier. No attempt has been made at this stage to investigate the complexity of a code that achieves this improvement.

6.7 Consideration of systems with a large number of carriers

Up to this point the number of carriers on the examined systems has been fairly small. Direct extension to a larger number of carriers is possible but rapidly becomes computationally cumbersome. However, it is possible that the coding of systems with a large number of carriers can be addressed by segmentation into blocks of fewer carriers, where the techniques presented here can form a basis for more advanced coding schemes.

As a simple example, each segment can be coded with one of the codes presented earlier, giving some PAPR reduction. Coding schemes that take into account the values of the other segments can give better results. The identification of such schemes is the subject of ongoing research.

6.8 Aspects of error control

When both error control and PAPR reduction are required there are several possible approaches that can be used. The straightforward method is to use a cascaded coding scheme, for example a Reed–Solomon code encapsulating the line code that limits the PAPR. To combat the disadvantages of this method that were discussed earlier, a concatenated coding scheme can be used. Some

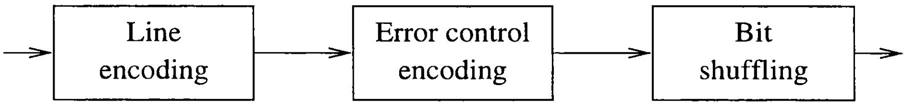

Figure 6.13: Coding structure
proposed in the previous chapter

existing work in this area proposed the use of Golay sequences and Reed–Muller codes[71] or m–sequences[72, 73].

Alternatively, the coding structure proposed in the previous chapter can be employed (Figure 6.13). As seen earlier, it is feasible to design line codes that control the PAPR, while their encoding and decoding is independent of the value of some bit positions. When such a code is utilised, the bit shuffling stage can distribute the parity bits of the systematic error control code into those bit positions. The resulting sequence can then be error corrected before line decoding at the receiver, allowing the use of high rate line and error control codes without sacrificing any of the error control code capability.

6.9 Summary

New codes that can be used to reduce the peak to average power ratio of multi–carrier modulation systems have been devised and their complexity investigated. In particular, two very simple codes were presented that achieve some improvement in the PAPR for both BPSK and QPSK systems. These work by adding one single bit of redundancy, its value being the inverse of a suitably chosen bit of the code–word.

Moreover, the encoder and decoder overhead for several other codes of the same rate was studied. An estimate of their complexity was investigated as a function of the required PAPR reduction. Furthermore, the use of two bits of redundancy with QPSK modulation was explored: one of the carriers is dedicated to reducing the PAPR. This is of interest for systems with individually modulated carriers.

It has been noted that this study has been constrained to small numbers of carriers, but that the techniques presented here may be extended to apply to larger systems, with more carriers, or may form a basis for the design of more complicated coding schemes for such systems.

Finally, the use of such codes in conjunction with error control coding was also discussed, and the coding structure proposed in the previous chapter was found appropriate for this case as well.

The following chapter brings the thesis to an end by reviewing the main contributions and identifying areas for further work.

7 Concluding remarks

7.1 Introduction

In this thesis, several constrained channels have been studied and appropriate coding techniques were developed to exploit the individual characteristics of the channel. The aim has been to offer better performance compared to conventional generic solutions and to do so with an emphasis on relative simplicity for implementation of the proposed coding structures. This concluding chapter will review the main contributions of this study and identify some areas where further work is likely to prove fruitful.

7.2 Contributions of this thesis

The main contributions of this thesis are as follows:

In the third chapter:

• Previously unpublished properties of asymmetric Z–channel error correcting codes were identified. These are considered to be important contributions from a theoretical and a practical point of view since they aid the search for both tighter code bounds and new, higher rate, codes.

• New tighter upper bounds for the size of asymmetric Z–channel error correcting codes are presented. Those are based on the above mentioned new properties as well as recently published values for related bounds.

• Asymmetric Z–channel error correcting codes with more code–words than those available in the literature were designed.

In the fourth chapter:

• A new disparity constrained error correcting line coding structure was proposed and suitable codes were identified. This is an extension of an existing technique, that allows the generation of higher rate codes and can be very useful when combined codes are required.

• The possibility of combining asymmetric Z–channel error control with disparity control to realise combined asymmetric Z–channel error correcting line codes was explored and examples of such codes were devised, establishing for the first time the feasibility of this strategy.

In the fifth chapter:

• A coding structure that can achieve overall high coding rate in combined run length limited codes together with conventional systematic error correcting codes, without a loss of error correcting performance was identified. This coding structure is potentially very useful since it can be extended for use in other coding problems as well.

• To aid in the design of such codes, optimisation techniques were developed that enabled reduced hardware complexity of line codes. To the best of my knowledge, no other such techniques have been previously presented and the proposed ones give very good results. Therefore, the resulting code implementations can have a significantly simpler implementation compared with a more conventional table lookup implementation.

In the sixth chapter:

• A study of the trade–off between implementation complexity and performance in high rate peak–to–average power ratio reducing codes for multi–carrier systems was presented, concentrating on systems with a small number of carriers. The techniques presented can form a basis for addressing the problem in systems with a large number of carriers

• Specific new high rate codes were developed based on this study, that give a considerable reduction of the peak–to–average power ratio on systems with a small number of carriers.

• The generality of the coding structure proposed in the fifth chapter is demonstrated by the design of a combined PAPR reducing error correcting coding scheme.

A more detailed overview of the contributions made in each chapter can be found in their respective concluding sections.

7.3 Suggestions for further work

Promising areas for further study include the following:

• Devise means of cascading disparity constrained line codes with systematic error correcting codes, similar to the run length limiting technique of chapter four. A simple approach to this is to code all but one of the information bits of a linear transparent systematic code with a disparity control code, then error control encode, and if the overall disparity has the same sign as the running disparity invert the whole codeword and use the previously uncoded bit to keep track of the inversion. This approach will not provide very tight disparity bounds but further improvements may be possible.

• Identify run length limited asymmetric Z–channel error correcting codes. Every asymmetric Z–channel error correcting code can be converted to such a code by removing the all one and all zero word. Then it is trivial to show that the maximum run of the code will be less than or equal to $2(n - \Delta)$, where n is the length of the code–word and Δ is the minimum asymmetric distance of the code. However, codes with smaller maximum runs providing for efficient encoding and decoding should be possible.

• Investigate ways to construct long codewords for multi–carrier systems with limited peak to average power ratio. This might be achieved for example, by concatenating shorter codewords. The resulting peak to average power ratio reduction is slightly higher than can be achieved with each of the shorter codewords. However, the possible gain for the same rate in longer codewords is many times more, so it is appropriate for more efficient schemes to be expected.

Appendix A: Limits of constant weight codes

A.1 Introduction

In this appendix a few quantities that are used in the elsewhere to calculate bounds on the size of asymmetric error correcting codes are discussed and bounds on their values are given.

One useful quantity is $A(n, d, w)$, the maximum number of code–words in any binary code of length n that have constant weight w and minimum distance d. Since not all of these numbers are known, upper and lower bounds to their value are of interest and several methods have been proposed for their calculation. Here, we concentrate on the calculation of the upper bound. Furthermore, lower bounds from several sources are collected and presented together with the respective upper bounds.

Another quantity that is useful is $T(w_1, n_1, w_2, n_2, d)$, which is the number of code–words of length $n_1 + n_2$, with w_1 one bits in the first n_1 bits and w_2 ones in the remaining n_2 bits of the code–word. Some simple bounds for T are presented as well.

A.2 Trivial values

Some values of $A(n, d, w)$ are easy to calculate.

$$A(n, d, w) = A(m, d + 1, w) \qquad \text{if } d \text{ is odd.}$$

$$A(n, d, w) = A(n, d, n - w).$$

$$A(n, d, w) = 1 \qquad \text{if } 2w < d.$$

$$A(n, d, w) = \lfloor \tfrac{n}{w} \rfloor \qquad \text{if } 2w = d.$$

$$A(n, 2, w) = \tbinom{n}{w}.$$

A.3 Johnson's bounds

Johnson[75] presented several methods that can be used to calculate upper bounds to $A(n,d,w)$.

$$A(n,d,w) \leq \lfloor \frac{n}{w} A(n-1,d,w-1) \rfloor, \qquad (n \geq w \geq 1),$$
$$A(n,d,w) \leq \lfloor \frac{n}{n-w} A(n-1,d,w) \rfloor, \qquad (n > w \geq 0).$$

Furthermore, he described the following theorem:

Suppose $A(n,d,w) = M$, and q, r are the integers satisfying

$$wM = nq + r, \qquad 0 \leq r < n.$$

Then

$$nq(q-1) + 2qr \leq (w - d/2)M(M-1).$$

Using this theorem some values of M can be rejected, thereby leading to tighter upper bounds.

Another bound he proposed is the following:

$$A(n,2u,w) \leq M_j(n,2u,w) = \binom{n}{w-j} / F_j(n,2u,w),$$

with $u = j + 2g, \mid j \mid \leq d/2$ and

$$F_j(n,2u,w) = \sum_{i=0}^{g-1} \binom{w}{j+i}\binom{n-w}{i} + K_u^j.$$
$$K_u^j \geq \max\{\frac{C_u^j}{t_j}, \frac{2m_u^j C_u^j - D_u^j}{m_u^j(1+m_u^j)}\},$$

where $m_u^j = \lfloor 1 + D_u^j / C_u^j \rfloor$, $t_j = \min\{\lfloor \frac{w-j}{g} \rfloor, \lfloor \frac{n-w+j}{u-g} \rfloor\}$, $D_u^j \leq \binom{u}{g}^2 T(w-u, w, u, n-w, 2u)$ and $C_u^j = \binom{w}{j+g}\binom{n-w}{g}$.

Calculating the values of M_j for all possible values of j we get an improved upper bound on several occasions.

A.4 Bounds using linear programming

Another method that can be used to calculate bounds on codes is linear programming.

Let C be a code of length n, weight $w \leq \frac{n}{2}$ and weight distribution $A_0, \ldots, A_{2w}$. Then, to obtain a bound on $A(n,d,w)$ we maximise the sum $A_0 + A_1 + \ldots + A_{2w}$ subject to the constraints

$$A_{2i} \geq 0, \qquad i = d/2, \ldots, w,$$

$$A_0 = 1, A_2 = \ldots = A_{d-2} = 0.$$

Furthermore, we have upper limits on the number of code–words A_{2i} at distance $2i$ from a given code–word:

$$A_{2i} \leq T(i, w, i, n-w, d), \qquad i = d/2, \ldots, w.$$

The following two linear programmes have been described in the literature, giving additional constraints on the numbers A_i.

A.4.1 Delsarte's bound

Best et. al.[76] present a linear programming arrangement that gives an upper bound of $A(n,d,w)$ based on the following theorem given by Delsarte[77].

$$\sum_{i=0}^{w} A_{2i} Q_k(i,n,w) \geq 0, \qquad k = 0, ..., w$$

The coefficients $Q_k(i,n,w)$ are given by

$$Q_k(i,n,w) = \frac{n-2k+1}{n-k+1} E_i(k) \binom{n}{k} / \binom{w}{i} \binom{n-w}{i}$$

and $E_i(x)$ is an Eberlein polynomial defined by

$$E_i(x) = \sum_{j=0}^{i} (-1)^{i-j} \binom{w-j}{i-j} \binom{w-x}{j} \binom{n-w+j-x}{j}$$

A.4.2 Van Pul's bound

C. L. M. van Pul[78] presented a different linear programming bound in his Master's thesis:

$$B_k = \sum_{i=0}^{w} ((\binom{n}{k} - w^{(k)})w^{(k)} - \frac{1}{2}\binom{n}{k}\sum_{j=0}^{i-1}\binom{2i}{2j+1}\binom{n-2i}{k-2j-1}),$$

$$B_k \geq 0, \quad k = 1, ..., n$$

where

$$w^{(k)} = \sum_{i=0}^{\lfloor \frac{w}{2} \rfloor} \binom{w}{2i+1}\binom{n-w}{k-2i-1}.$$

Furthermore, Van Pull gives a lower bound on the quantities B_k that can be used to tighten the constraints of the linear programme. Specifically, if $A(n, d, w) = M$ and

$$Mw^{(k)} = q_k \binom{n}{k} + r_k, \qquad 0 \leq r_k < \binom{n}{k}$$

then

$$B_k \geq r_k(\binom{n}{k} - r_k)/M.$$

This lower limit can be utilised in the following way: after we have established an upper value for $A(n, d, w)$, we set M equal to that value. Then we try to solve the linear programme using the lower limits for B_k that we can calculate using this value of M. If the programme is infeasible, then we repeat the procedure with the next lower M, until we get a feasible solution. Then M is our new upper limit.

n \ w	3	4	5	6	7	8	9	10	11	12	13	14
6	4											
7	7											
8	8	14										
9	12	18										
10	13	30	36									
11	17	35	66									
12	20	51	80-84	132								
13	26	65	123-132	166-182								
14	28	91	169-182	278-308	325-364							
15	35	105	237-271	389-455	585-660							
16	37	140	315-336	615-722	836-1040	1170-1320						
17	44	156-157	441-476	854-952	1416-1753	1770-2210						
18	48	198	518-565	1260-1428	2041-2448	3186-3944	3540-4420					
19	57	228	692-752	1620-1789	3172-3876	4667-5814	6726-8326					
20	60	285	874-912	2304-2506	4213-5111	7730-9690	10039-12920	13452-16652				
21	70	315	1071-1197	2856-3192	6156-7518	10753-13416	16897-22609_2	20188-27131_2				
22	73	385	1386	3927-4389	8252-10032	16430-20674	25570-32794	36381-49739	39688-54262			
23	83	418-419	1771	5313	11638-14421	23276-28842	40786-52833	57436-75426	73794-103999			
24	88	498	1895-2011	7084	15656-18216	34914-43262_2	59387-76911_3	96496-126799	116937-164565	146552-207998		
25	100	550	2334-2490	7772-8379	21106-25299_3	46872-56925	88748-120172	140605-192277	196449-288179	228901-342843		
26	104	650	2670-2860	10010-10790	26920-31122	65364-82221	128050-164449_3	218905-312447	315700-454472	398381-624387	425950-685686	
27	117	702	3276-3510	12012-12870	35510-41618	87709-105036	186058-246663	330347-444012	510571-766915	675262-1022562	778872-1296803	
28	121	819	3718-3931	15288-16380	44747-51480	121403-145663	260224-326778	502068-690656	806303-1130212	1154541-1789468	1400118-2202441	1520224-2593606

Table A.1: $A(n, 4, w)$

n \ w	4	5	6	7	8	9	10	11	12	13	14
8	2										
9	3										
10	5	6									
11	6	11									
12	9	12	22								
13	13	18	26								
14	14	28	42	42							
15	15	42	70	69-78							
16	20	48	112	109-138	120-150_1						
17	20	68	112-136	166-234	184-282_2						
18	22	68-72	132-202	243-349	260-427_1	304-424_3					
19	25	76-83	172-228	338-520_1	408-734	504-789_1					
20	30	84-100	232-276	462-651	588-1107	832-1363_1	944-1420_3				
21	31	108-126	269-350	570-828	774-1695	1184-2364_1	1454-2701_3				
22	37	132-136	319-462	759-1100	1139-2277	1792-3774_3	2182-4310_1	2636-5064_1			
23	40	147-170	399-521	969-1518	1436-3162	2271-5819	2970-7521_1	3585-7953_1			
24	42	168-192	532-680	1368-1786	1882-4554	3041-8432	4200-12186	5267-14682	5616-15906		
25	50	210	700-800	1900-2428	2590-5581	4127-12620	6036-19037	7960-24630_1	9031-30587		
26	52	260	910	2600-2971	3532-7891	5703-16122	8695-28893	12037-42081_1	14836-49233_1	15977-61174	
27	54	260-280	1170	3510	4786-10027	7727-23673	12368-43529	18096-66078_3	23879-84574_1	27553-91079_3	
28	63	280-302	1170-1306	4680	6315-12285	10313-31195	17447-63756	29484-104231_1	40188-142117	49462-164219_3	52995-169739_3

Table A.2: $A(n,6,w)$

n \ w	5	6	7	8	9	10	11	12	13	14
10	2									
11	2									
12	3	4								
13	3	4								
14	4	7	8							
15	6	10	15							
16	6	16	16	30						
17	7	17	24	34						
18	9	21	33-39	46-54	48-68					
19	12	28	52-57	78-92	88-114					
20	16	40	80	130-142	160-204	176-228				
21	21	56	120	210	280-318_1	336-423_2				
22	21	77	176	330	280-493	616-639_1	672-766_2			
23	23	77-80	253	506	400-795_2	616-1110_3	1288-1328_1			
24	24	78-92	253-274	759	640-1143	960-1638_3	1288-2188_1	2576		
25	30	100	254-328	759-856	829-1610	1248-2448_1	1662-3516_1	2576-4168_2		
26	30	130	257-371	760-1066	883-2160	1519-3719_1	1988-5314_3	3070-6790_1	3588-7566_3	
27	30-32	130-135	278-500	766-1252	970-2914	1597-5260_1	2295-7837_1	3335-10547_1	4094-12148_1	
28	33	130-149	296-540	833-1750	1107-3895	1820-7368_1	2756-11939	4916-17299_1	4805-21739_1	6090-23268_1

Table A.3: $A(n, 8, w)$

n \ w	6	7	8	9	10	11	12	13	14
12	2								
13	2								
14	2	2							
15	3	3							
16	3	4	4						
17	3	5	6						
18	4	6	9	10					
19	4	8	12	19					
20	5	10	17	20-26	38				
21	7	13	21	27-39	38-54				
22	7	16	24-33	35-51	46-81_1	46-86_2			
23	8	20-23	33-46	45-83	54-116_2	65-135_1			
24	9	24-27	38-69	56-118_2	72-170_2	95-222_2	122-246_2		
25	10	28-32	48-84	72-158_1	100-262_1	125-385_2	132-462		
26	13	28-36	54-104	91-213_2	130-410	168-576_2	195-727_1	210-886_2	
27	14	36-48	66-121	118-298_3	162-575	222-972	351-1288_2	405-1460_1	
28	16	37-56	78-168	132-376	210-821_1	286-1435_2	365-1980_2	756-2438_1	790-2629_1

Table A.4: $A(n, 10, w)$

n \ w	7	8	9	10	11	12	13	14
14	2							
15	2							
16	2	2						
17	2	2						
18	3	3	4					
19	3	3	4					
20	3	5	5	6				
21	3	5	7	7				
22	4	6	8	11	12			
23	4	6	10	16	23			
24	4	9	16	24	24-34	46		
25	5	10	25	28-35_2	36-51_1	50-67_2		
26	5	13	26-28	33-56	39-82	54-97_1	58-104_1	
27	6	15	39	39-75	54-110_2	82-139_2	86-156_2	
28	8	19	39-46	49-102_2	65-148_2	84-198_2	99-244_2	172-264_2

Table A.5: $A(n, 12, w)$

n \ w	8	9	10	11	12	13	14
16	2						
17	2						
18	2	2					
19	2	2					
20	2	2	2				
21	3	3	3				
22	3	3	4	4			
23	3	3	4	4			
24	3	4	5	6	6		
25	3	5	6	7	8		
26	4	6	8	10	13	14	
27	4	6	9	13	19-20	27	
28	4	7	11	21	28-30	28-41_2	54

Table A.6: $A(n, 14, w)$

A.4.3 Additional constraints

The results of the linear programme can be improved by using some additional constraints. In particular, for several parameter combinations we find that $A_{2w} \leq 1$. The existence of a code–word at this distance from another code–word constrains the distribution of other code–words, giving extra terms of the form

$$A_{2i} + (T(i, w, i, n-w, d) - t)A_{2w} < T(i, w, i, n-w, d),$$

where t is given by:

$$t = \sum_{j=d/2}^{w} T(j, w, i, w, j-i, n-2w, d).$$

$T(w1, n1, w2, n2, w3, n3, d)$ is the extension of T to three pairs of w, n.

A.5 Tables of upper and lower bounds of $A(n, d, w)$

The tables presented here, give both upper and lower bound to $A(n, d, w)$, or the exact value where it is known. The range selected for the values of n, d and w in these tables is the same as that used in [28], except that $d \leq 14$.

The lower limits were obtained from two sources. The main one is the tables of Brouwer et. al.[28]. Furthermore, some newer values obtained by Nurmela et. al.[79, 80], together with values obtained from the tables electronically published by Rains and Sloane[81], were added.

The upper limits were calculated using all the methods described earlier. The linear programmes were solved using the simplex procedure, as described by Sultan[21]. Where there is no suffix to a value, then the value was calculated using Johnson's formulas. Where linear programming was used there is a suffix specifying which method was used; 1 where the value was obtained using Delsarte's programme, 2 when Van Pul's bound was used, and 3 when a combination of both programmes gave the presented result.

On several occasions, the values obtained for the upper limit were worse than the ones presented in both Best[76] and Van Pul[78]. Specifically, the following tighter upper bounds were given:

$A(21, 8, 10) \leq 399$ and $A(20, 10, 9) \leq 24$ as presented in reference [76] as well as $A(21, 10, 10) \leq 44$, $A(22, 10, 10) \leq 73$, $A(22, 10, 11) \leq 81$ and $A(24, 10, 8) \leq 68$ found in reference [78].

Most of these values were calculated using linear programming, except for $A(20, 10, 9)$ and $A(24, 10, 8)$, which seem to be in error since no references are given. For the rest of the values, it is possible that the upper limits of the value of $T(w1, n1, w2, n2, d)$ used were more exact, possibly calculated through the linear programming bound described in [76]. However, several of the values of $T(w1, n1, w2, n2, 10)$ that were presented in that paper were found to be in error[28].

Furthermore, the additional constrains described earlier that can be used when $A_{2w} \leq 1$ can be generalised for any value $A_{2i} \leq 1$. Such values exist for many of the parameter sets where a discrepancy was found.

A.5 Summary

A detailed study of the upper limit of the number of code–words in a constant weight code was presented, together with a collection of the latest published lower bounds. These assist in the calculation of new upper and lower bounds of the number of code–words in asymmetric error correcting codes, together with the properties of such codes that were discussed in the third chapter.

References

[1] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "A new error correcting line code," in *ITS/IEEE ROC&C International Telecommunications Symposium*, (Acapulco, Mexico), pp. 54–58, Oct. 1996.

[2] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "Exploiting soft decision decoding for error correcting line codes," in *IEEE ICCS/ISPACS Conference Proceedings*, vol. 2, (Singapore), pp. 638–642, Nov. 1996.

[3] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "Soft decision error correcting line code for optical data storage," in *LEOS Conference Proceedings*, vol. 1, (Boston, USA), pp. 201–202, Nov. 1996.

[4] S. Fragiacomo, Y. Bian, A. Popplewell, C. Matrakidis, and J. J. O'Reilly, "An accelerated simulation technique for evaluating communication systems using FEC," in *Proceedings of the European Conference on Networks and Optical Communications*, vol. 2, (Antwerp, Belgium), pp. 145–148, June 1997.

[5] C. Matrakidis and J. J. O'Reilly, "A block decodable line code for high speed optical communication," in *IEEE International Symposium on Information Theory*, (Ulm, Germany), p. 221, June 1997.

[6] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "A class of low complexity line codes," in *IEEE International Symposium on Information Theory*, (Ulm, Germany), p. 219, June 1997.

[7] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "Class of low complexity line codes for optical data storage and communications," in *The Pacific*

Rim Conference on Lasers and Electro–Optics, (Chiba, Japan), pp. 92–93, July 1997.

[8] C. Matrakidis and J. J. O'Reilly, "Limiting the maximum run–length of block turbo codes," in *Proceedings of the International Symposium on Turbo Codes & Related Topics*, (Brest, France), pp. 220–222, Sept. 1997.

[9] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "Performance aspects of a class of low complexity line codes," in *International Conference on Signal Processing Applications & Technology*, vol. 1, (San Diego, USA), pp. 632–636, Sept. 1997.

[10] S. Fragiacomo, C. Matrakidis, and J. J. O'Reilly, "Multicarrier transmission peak–to–average power reduction using simple block code," *Electronics Letters*, vol. 34, pp. 953–954, May 1998.

[11] S. Fragiacomo, C. Matrakidis, A. Popplewell, and J. J. O'Reilly, "Novel accelerated technique for low bit error rate communication systems," *IEE Proceedings on Communications*, vol. 145, pp. 337–341, Oct. 1998.

[12] K. W. Cattermole, "Principles of digital line coding," *International Journal of Electronics*, vol. 55, no. 1, pp. 3–33, 1983.

[13] K. A. Schouhamer Immink, *Properties and Constructions of Binary Channel Codes.* PhD thesis, Technishe Hogeschool Eindhoven, May 1985.

[14] D. T. Tang and L. R. Bahl, "Block codes for a class of constrained noiseless channels," *Information and Control*, vol. 17, pp. 436–461, 1970.

[15] S. Lin and D. J. Costello, Jr., *Error Control Coding: Fundamentals and Applications.* Prentice–Hall, Inc., 1983.

[16] J. R. Pierce, "Optical channels: Practical limits with photon counting," *IEEE Transactions on Communications*, vol. 26, pp. 1819–1821, Dec. 1978.

[17] S. D. Constantin and T. R. N. Rao, "On the theory of binary asymmetric error correcting codes," *Information and Control*, vol. 40, pp. 20–36, 1979.

[18] R. R. Varshamov, "Some features of linear codes that correct asymmetric errors," *Soviet Physics–Doklady*, vol. 9, pp. 538–540, Jan. 1965.

[19] J. H. Weber, C. de Vroedt, and D. E. Boekee, "Necessary and sufficient conditions on block codes correcting/detecting errors of various types," *IEEE Transactions on Computers*, vol. 41, pp. 1189–1193, Sept. 1992.

[20] T. Kløve, "Upper bounds on codes correcting asymmetric errors," *IEEE Transactions on Information Theory*, vol. 27, pp. 128–131, Jan. 1981.

[21] A. Sultan, *Linear Programming. An Introduction with Applications.* Academic Press, Inc., 1993.

[22] S. Walukiewicz, *Integer Programming.* Kluwer Academic Publishers, 1991.

[23] J. H. Weber, C. de Vroedt, and D. E. Boekee, "New upper bounds on the size of codes correcting asymmetric errors," *IEEE Transactions on Information Theory*, vol. 33, pp. 434–437, May 1987.

[24] J. H. Weber, C. de Vroedt, and D. E. Boekee, "Bounds and constructions for binary codes of length less than 24 and asymmetric distance less than 6," *IEEE Transactions on Information Theory*, vol. 34, pp. 1321–1331, Sept. 1988.

[25] Y. Saitoh, K. Yamaguchi, and H. Imai, "Some new binary codes correcting asymmetric/unidirectional errors," *IEEE Transactions on Information Theory*, vol. 36, pp. 645–647, May 1990.

[26] Z. Zhang and Z.-G. Xia, "New lower bounds for binary codes of asymmetric distance two," *IEEE Transactions on Information Theory*, vol. 38, pp. 1592–1597, Sept. 1992.

[27] S. Al–Bassam, R. Venkatesan, and S. Al–Muhammadi, "New single asymmetric error–correcting codes," *IEEE Transactions on Information Theory*, vol. 43, pp. 1619–1623, Sept. 1997.

[28] A. E. Brouwer, J. B. Shearer, N. J. A. Sloane, and W. D. Smith, "A new table of constant weight codes," *IEEE Transactions on Information Theory*, vol. 36, pp. 1334–1380, Nov. 1990.

[29] T. Etzion, "New lower bounds for asymmetric and unidirectional codes," *IEEE Transactions on Information Theory*, vol. 37, pp. 1696–1704, Nov. 1991.

[30] E. Gorog, "Redundant alphabets with desirable frequency spectrum properties," *IBM Journal of Research and Development*, vol. 12, pp. 234–241, May 1968.

[31] G. L. Pierobon, "Codes for zero spectral density at zero frequency," *IEEE Transactions on Information Theory*, vol. 30, pp. 435–439, Mar. 1984.

[32] R. O. Carter, "Low–disparity binary coding system," *Electronics Letters*, vol. 1, pp. 67–68, May 1965.

[33] J. M. Griffiths, "Binary code suitable for line transmission," *Electronics Letters*, vol. 5, pp. 79–81, Feb. 1969.

[34] G. L. Cariolaro and G. P. Tronca, "Spectra of block coded digital signals," *IEEE Transactions on Communications*, vol. 22, pp. 1555–1564, Oct. 1974.

[35] G. L. Cariolaro and G. P. Tronca, "Correlation and spectral properties of multilevel (M,N) coded digital signals with applications to pseudoternary (4,3) codes," *Alta Frequenza*, vol. 43, pp. 2–15, Jan. 1974.

[36] G. L. Cariolaro, G. L. Pierobon, and G. P. Tronca, "Analysis of codes and spectra calculations," *International Journal of Electronics*, vol. 55, no. 1, pp. 35–79, 1983.

[37] G. S. Poo, "Computer aids for code spectra calculations," *IEE proceedings F*, vol. 128, pp. 323–330, Oct. 1981.

[38] R. H. Deng and M. A. Herro, "DC–free coset codes," *IEEE Transactions on Information Theory*, vol. 34, pp. 786–792, July 1988.

[39] J. J. O'Reilly and A. Popplewell, "Class of disparity reducing transmission codes with embedded error protection," *IEE Proceedings I*, vol. 137, pp. 73–77, Apr. 1990.

[40] A. Popplewell, *Combined Line and Error Control Coding.* PhD thesis, University of Wales, Bangor, Mar. 1990.

[41] R. W. Tkach, R. M. Derosier, F. Forghieri, A. H. Gnauck, A. M. Vengsarkar, D. W. Peckham, J. L. Zyskind, J. W. Sulhoff, and A. R. Chraplyvy, "Transmission of eight 20–gb/s channels over 232 km of conventional single–mode fiber," *IEEE Photonics Technology Letters*, vol. 7, pp. 1369–1371, Nov. 1995.

[42] W. A. Krzymien, "Transmission performance analysis of a new class of line codes for optical fiber systems," *IEEE Transactions on Communications*, vol. 37, pp. 402–404, Apr. 1989.

[43] J. P. J. Heemskerk and K. A. S. Immink, "Compact disc: System aspects and modulation," *Philips Technical Review*, vol. 40, no. 6, pp. 157–164, 1982.

[44] W. G. Bliss, "Circuitry for performing error correction calculations on baseband encoded data to eliminate error propagation," *IBM Technical Disclosure Bulletin*, vol. 23, no. 10, pp. 4633–4634, 1981.

[45] K. A. Schouhamer Immink, "Code configuration for avoiding error propagation," *Electronics Letters*, vol. 32, pp. 2191–2192, Nov. 1996.

[46] K. A. S. Immink, "A practical method for approaching the channel capacity of constrained channels," *IEEE Transactions on Information Theory*, vol. 43, pp. 1389–1399, Sept. 1997.

[47] J. L. Fan and A. R. Calderbank, "A modified concatenated coding scheme, with applications to magnetic data storage," *IEEE Transactions on Information Theory*, vol. 44, pp. 1565–1574, July 1998.

[48] A. Popplewell and J. J. O'Reilly, "Runlength limited binary error control codes," *IEE Proceedings I*, vol. 139, pp. 349–355, June 1992.

[49] A. Popplewell and J. J. O'Reilly, "New class of runlength–limited error–control codes with minimum distance 4," *IEE Proceedings I*, vol. 140, pp. 104–108, Apr. 1993.

[50] A. J. van Wijngaarden and K. A. S. Immink, "Combinatorial construction of high rate runlength–limited codes," in *IEEE Globecom Conference Record*, vol. 1, (London, UK), pp. 343–347, Nov. 1996.

[51] K. A. S. Immink and A. J. van Wijngaarden, "Simple high–rate constrained codes," *Electronics Letters*, vol. 32, p. 1877, Sept. 1996.

[52] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *Journal of Chemical Physics*, vol. 21, pp. 1087–1092, June 1953.

[53] A. A. El Gamal, L. A. Hemachandra, I. Shperling, and V. K. Wei, "Using simulated annealing to design good codes," *IEEE Transactions on Information Theory*, vol. 33, pp. 116–123, Jan. 1987.

[54] "Xilinx xc3000 series field programmable gate arrays," Nov. 1997.

[55] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. Sangiovanni–Vincentelli, "Sis: A system for sequential circuit synthesis," tech. rep., Department of Electrical Engineering and Computer Science, University of California, Berkeley, May 1992.

[56] A. E. Jones, T. A. Wilkinson, and S. K. Barton, "Block coding scheme for reduction of peak to mean envelope power ratio of multicarrier transmission schemes," *Electronics Letters*, vol. 30, pp. 2098–2099, Dec. 1994.

[57] L. W. Couch, *Digital and Analog Communication Systems.* Macmillan Publishing Company, 1993.

[58] D. Wulich, "Peak factor in orthogonal multicarrier modulation with variable levels," *Electronics Letters*, vol. 32, pp. 1859–1860, Sept. 1996.

[59] L. J. Greenstein and P. J. Fitzgerald, "Phasing multitone signals to minimize peak factors," *IEEE Transactions on Communications*, vol. 29, pp. 1072–1074, July 1981.

[60] S. Boyd, "Multitone signals with low crest factor," *IEEE Transactions on Circuits and Systems*, vol. 33, pp. 1018–1022, Oct. 1986.

[61] A. van den Bos, "A new method for synthesis of low–peak–factor signals," *IEEE Transactions on Acoustics, Speech, and signal Processing*, vol. 35, pp. 120–122, Jan. 1987.

[62] E. van der Ouderaa, J. Schoukens, and J. Renneboog, "Comments on "multitone signals with low crest factor"," *IEEE Transactions on Circuits and Systems*, vol. 34, pp. 1125–1127, Sept. 1987.

[63] D. W. Bennett, R. J. Wilkinson, and P. B. Kenington, "Determining the power rating of a multichannel linear amplifier," *IEE Proceedings Communications*, vol. 142, pp. 274–280, Aug. 1995.

[64] P. W. J. van Eetvelt, S. J. Shepherd, and S. K. Barton, "The distribution of peak factor in qpsk multi–carrier modulation," *Wireless Personal Communications*, vol. 2, pp. 87–96, 1995.

[65] S. Shepherd, J. Orriss, and S. Barton, "Asymptotic limits in peak envelope power reduction by redundant coding in orthogonal frequency–division multiplex modulation," *IEEE Transactions on Communications*, vol. 46, pp. 5–10, Jan. 1998.

[66] D. Wulich, "Reduction of peak to mean ratio of multicarrier modulation using cyclic coding," *Electronics Letters*, vol. 32, pp. 432–433, Feb. 1996.

[67] S. J. Shepherd, P. W. J. van Eetvelt, C. W. Wyatt-Millington, and S. K. Barton, "Simple coding scheme to reduce peak factor in qpsk multicarrier modulation," *Electronics Letters*, vol. 31, pp. 1131–1132, July 1995.

[68] M. Friese, "Multicarrier modulation with low peak–to–average power ratio," *Electronics Letters*, vol. 32, pp. 713–714, Apr. 1996.

[69] P. van Eetvelt, G. Wade, and M. Tomlinson, "Peak to average power reduction for ofdm schemes by selective scrambling," *Electronics Letters*, vol. 32, pp. 1963–1964, Oct. 1996.

[70] R. W. Bäuml, R. F. H. Fischer, and J. B. Huber, "Reducing the peak-to-average power ratio of multicarrier modulation by selected mapping," *Electronics Letters*, vol. 32, pp. 2056–2057, Oct. 1996.

[71] J. A. Davis and J. Jedwab, "Peak-to-mean power control and error correction for ofdm transmission using Golay sequences and Reed–Muller codes," *Electronics Letters*, vol. 33, pp. 267–268, Feb. 1997.

[72] X. Li and J. A. Ritcey, "M-sequences for ofdm peak-to-average power ratio reduction and error correction," *Electronics Letters*, vol. 33, pp. 554–555, Mar. 1997.

[73] C. Tellambura, "Use of m-sequences for ofdm peak-to-average power ratio reduction," *Electronics Letters*, vol. 33, pp. 1300–1301, July 1997.

[74] C. Tellambura, "Comment: Multicarrier transmission peak-to-average power reduction using simple block code," *Electronics Letters*, vol. 34, p. 1646, Sept. 1998.

[75] S. Johnson, "Upper bounds for constant weight error correcting codes," *Discrete Mathematics*, vol. 3, pp. 109–124, 1972.

[76] M. R. Best, A. E. Brouwer, F. J. MacWilliams, A. M. Odlyzko, and N. J. A. Sloane, "Bounds for binary codes of length less than 25," *IEEE Transactions on Information Theory*, vol. 24, pp. 81–93, Jan. 1978.

[77] P. Delsarte, *An Algebraic Approach to the Association Schemes of Coding Theory*. PhD thesis, Université Catholique de Louvain, June 1973.

[78] C. L. M. van Pul, "On bounds on codes," Master's thesis, Eindhoven University of Technology, Aug. 1982.

[79] K. J. Nurmela, M. K. Kaikkonen, and P. R. J. Östergård, "New constant weight codes from linear permutation groups," in *IEEE International Symposium on Information Theory*, (Ulm, Germany), p. 453, 1997.

[80] K. J. Nurmela, M. K. Kaikkonen, and P. R. J. Östergård, "New constant weight codes from linear permutation groups," *IEEE Transactions on Information Theory*, vol. 43, pp. 1623–1630, Sept. 1997.

[81] E. M. Rains and N. J. A. Sloane, "Table of constant weight binary codes," *http://www.research.att.com/~njas/codes/Andw*, May 1998.