On The Performance Of Focused Error Control Codes

by

Fady I. Alajaji

Thesis submitted to the Faculty of The Graduate School
of The University of Maryland in partial fulfillment
of the requirements for the degree of
Master of Science
1990

Advisory Committee:

Dr. Thomas Fuja Advisor
Dr. Prakash Narayan
Dr. Stephen Tretter

ABSTRACT

Title of Thesis:	On The Performance Of Focused Error Control Codes
Name of Degree Candidate:	Fady I. Alajaji
Degree and Year:	Master of Science, 1990
Thesis directed by:	Dr. Thomas Fuja Assistant Professor Department of Electrical Engineering

Consider an additive noise channel with inputs and outputs in the field GF(q), where $q > 2$; every time a symbol is transmitted (or stored) over such a channel, there are q-1 different errors that can occur, corresponding to the q-1 non-zero elements that the channel can add to the transmitted symbol. Among these errors, there are some errors that occur much more frequently than others; we call them "common" errors. However, "traditional" error correcting codes - designed with respect to the Hamming metric - treat each of these q-1 errors the same. Fuja and Heegard have designed a class of codes, called focused error control codes, that can offer different levels of protection against "common" and "uncommon" errors. (The motivating example: for many codes over GF(2^b) where $b > 1$, almost all errors involve exactly one bit per symbol.) If we let $\mathbf{B}$ denote a set of common errors, then a code is (t_1, t_2)-*focused* on $\mathbf{B}$ if it can correct up to $t_1 + t_2$ errors provided at most t_1 of these errors lie outside $\mathbf{B}$.

In this thesis, we study the performance of these codes in terms of rate and performance tradeoffs with respect to "idealized" skewed channels as well as realistic non-binary modulation schemes.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction And Motivation

When a symbol from a codeword over GF(q) is sent over a channel with additive noise, there are q-1 different non-zero noise symbols that can corrupt the sent field element."Traditional" error control codes, designed with respect to the Hamming metric, treat each of these q-1 possibilities the same, as simply representing a generic "error".

In many non-binary transmission channels or data communication and storage systems, there are some errors that occur much more frequently than others. These errors are called "common" errors. We say that a channel is skewed on a set $\mathbf{B} \subset GF(q)^*$ (where $q > 2$ and field $GF(q)^* = GF(q) - \{0\}$) if the "common" errors generated by the channel lie in $\mathbf{B}$.

As an example, consider a modulation scheme mapping data onto non-binary signals using a Gray code so that the most likely detection errors cause exactly one bit error per symbol. So for a given transmitted signal s_1, the most likely symbol detection error will be in detecting one of the signals that are directly adjacent to s_1 in the signal constellation. Thus, the most

likely errors will result in a received symbol that differs from the transmitted symbol in one bit of their binary representation; therefore $\mathbf{B}$={All field elements with a single "1" in their binary representation} forms the "common" error set.

As another example, consider random access memory systems employing "byte wide" RAM chips; that is each chip delivers $b \geq 2$ bits at a time. Codes over GF(2^b) can be used for such systems, with each chip's output constituting one element of GF(2^b). It has been observed that the vast majority of chip failures are single-cell failures and so would affect only one bit per byte; thus, most of the symbol errors that would confront such codes would be one of the b field elements with exactly one non-zero bit in its binary representation. Once again, these elements constitute the set of common errors $\mathbf{B}$. Of course, there exist some failure mechanisms that may cause multiple bit failures - catastrophic whole-chip failures (burst errors), for instance - so *some* protection against arbitrary errors would be needed; however providing the same degree of protection against "common" and "uncommon" errors is not efficient.

The above examples illustrate one of the most common applications where it becomes desirable to make the *distinction* between two different kinds of errors - when we wish to correct single-bit errors using a non-binary code. Moreover, there are some applications for which the set of common errors $\mathbf{B}$ is not simply the set of single-bit errors. For example, in memory systems organized into "byte-wide" chips, the typical failures that usually affect these chips are single-bit errors *and* two-bit-adjacent errors per byte.

In these and in many other applications it is desirable to "focus" the capabilities of a code on the class of common errors $\mathbf{B}$. Since "traditional" error control codes do not differentiate between "common" and "uncommon" errors, Fuja and Heegard [1] developed a new family of error control codes, the "focused" control codes for channels with skewed errors. These codes were designed to give a high level of protection against the set of common errors while keeping a certain level of protection against the uncommon errors.

Our *concern* is to study the performance of these codes in terms of rate/performance tradeoffs with respect to "idealized" skewed channels as

well as realistic non-binary modulation schemes.

1.2 Thesis Description

In chapter 2, we briefly introduce the concept of focused error control codes on channels with skewed errors and review the results of Fuja and Heegard [1] concerning their construction.

In chapter 3, we derive the general expression for the decoder error probability of focused codes and study their performance over "idealized" skewed symmetric channels. We then determine suitable conditions under which the performance of the focused codes is identical to that of the traditional error correcting codes.

In chapters 4 and 5, we derive respectively the analytical expressions for the parameters of additive white Gaussian noise channels associated with M-ary PSK modulation and square constellations. We then analyze the performance of focused control codes used in conjunction with M-ary PSK modulation and square constellations respectively and compare it to the performance of traditional codes. Some simulations on the focused codes performances are also done in order to check the accuracy of the results derived analytically.

In chapter 6, we provide some numerical results by constructing (t_1, t_2)-focused codes and computing the code rate improvements they achieve over $t_1 + t_2$-error correcting codes while having an identical performance, for different values of the code blocklength. We also calculate the coding gains (in energy to noise ratio) (t_1, t_2)-focused codes achieve over $t_1 + t_2$-error correcting codes and uncoded block messages under PSK and square constellation modulations.

Finally, the concept of adaptive decoding of focused codes is introduced in chapter 7. We study the case where we can attain performance matching between focused codes and traditional error control codes using an adaptive decoding algorithm.

Chapter 2

Past Results On Focused Control Codes

In this chapter, we review the results in [1] that are pertinent to the development of this thesis.

2.1 Focused Error Control Codes

2.1.1 The Skewed Symmetric Channel (SSC)

Consider the following channel model for storage or transmission. A character $X \in GF(q)$ is to be transmitted and the character $Y = X + Z \in GF(q)$ is received. It is assumed that the random error, $Z \in GF(q)$, is independent of the input, X. This transmission channel with inputs and outputs over GF(q), $(q > 2)$, is said to be *skewed* on a subset $\mathbf{B} \subset GF(q)^*$ if:

$$P(Z \in \mathbf{B} \mid Z \neq 0) > \frac{|\mathbf{B}|}{(q-1)}.$$

More specifically, a *skewed symmetric channel* is a channel in which the iid error Z is distributed according to:

$$Pr(Z=z) = \begin{cases} 1-\epsilon & \text{if z=0} \\ \dfrac{\epsilon(1-\gamma)}{|\mathbf{B}|} & \text{if } z \in \mathbf{B} \\ \dfrac{\epsilon\gamma}{(q-1-|\mathbf{B}|)} & \text{if } z \in \mathbf{B^c} \end{cases}$$

where:

- ϵ=P($Z \neq 0$)=probability of symbol channel error.
- γ=Pr(error is uncommon | an error has occurred).

Note that in the cases of interest, $\epsilon \ll 1$ and $\gamma \ll 1$. Moreover, $\mathbf{B}$ is the set of common errors and $\mathbf{B^c} = GF(q)^* - \mathbf{B}$ is the set of uncommon errors. Within each class of errors, we assume a *uniform* distribution.

2.1.2 Definition Of Focused Codes

For any $\mathbf{x} \in GF(q)^n$, we denote the *Hamming weight* of $\mathbf{x}$ by $\|\mathbf{x}\|$; that is, if $\mathbf{x} = [x_0, x_1, ..., x_n]$, then

$$\|\mathbf{x}\| \stackrel{\text{def}}{=} \sum_{i=0}^{n-1} \mathbf{1}_{GF(q)^*}(x_i)$$

where $\mathbf{1}(.)$ is the indicator function (i.e., $\mathbf{1_A}(x)$ equals one if $x \in \mathbf{A}$ and equals zero otherwise).

More generally, for any set $\mathbf{A} \subseteq GF(q)^*$, we define the $\mathbf{A}$-*weight* of $\mathbf{x}$ (where $\mathbf{x} \in GF(q)^n$) as the number of components of $\mathbf{x}$ that lie in $\mathbf{A}$; if we

denote the **A**-*weight* of $\mathbf{x}$ by $\|\mathbf{x_A}\|$, then

$$\|\mathbf{x}\|_{\mathbf{A}} \stackrel{\text{def}}{=} \sum_{i=0}^{n-1} \mathbf{1_A}(x_i)$$

Obviously, we have that $\|\mathbf{x}\|_{\mathbf{A}} \leq \|\mathbf{x}\|$.

Definition:

Let $\mathbf{B} \subset GF(q)^*$ be a set of common errors (non-zero elements of GF(q)). A code is (t_1, t_2)-focused on $\mathbf{B}$ if it can correct up to $t_1 + t_2$ errors provided at most t_1 of these errors lie outside $\mathbf{B}$ (i.e. are uncommon). More precisely, such a code is a set $\mathbf{C}$, of n-tuples over GF(q) with the following property. There exists a decoding function $f : GF(q)^n \to \mathbf{C}$ such that $f(\mathbf{c} + \mathbf{e}) = \mathbf{c}$ for any $\mathbf{c} \in \mathbf{C}$ and any $\mathbf{e} \in GF(q)^n$ satisfying the following two conditions:

1. $\|\mathbf{e}\| \leq t_1 + t_2$;
2. $\|\mathbf{e}\|_{\mathbf{B^c}} \leq t_1$.

Note that:

- A (t,0)-focused code is a "traditional" t-error correcting code.
- A (0,t)-focused code is a code that is completely focused on $\mathbf{B}$; i.e. it can correct only up to t common errors.

Lemma:

Let $\mathbf{C}$ be a set of q-ary n-tuples with the following property. For any c_1 and $c_2 \in \mathbf{C}$, *at least one* of the following conditions holds:

- (i) $\|\mathbf{c_1} - \mathbf{c_2}\| > 2t_1 + 2t_2$;

- (ii) $\|\mathbf{c_1} - \mathbf{c_2}\| + \|\mathbf{c_1} - \mathbf{c_2}\|_{\mathbf{B^c}} > 4t_1 + 2t_2$.

Then $\mathbf{C}$ is (t_1, t_2)-focused on $\mathbf{B}$.

The implications of the above statement are presented graphically in Figure (2.1). We can plot every q-ary n-tuple in two dimensions by its Hamming weight and its $\mathbf{B^c}$-weight. As long as no codeword difference lies in the shaded region, the code will be (t_1, t_2)-focused on $\mathbf{B}$. By comparison, to insure correction of *all* error patterns of Hamming weight $t_1 + t_2$ or less, we would require that all codeword differences have Hamming weight greater than $2t_1 + 2t_2$; by lowering our requirements we have cut a "notch" in the "forbidden zone". This suggests that *rate improvements* are likely.

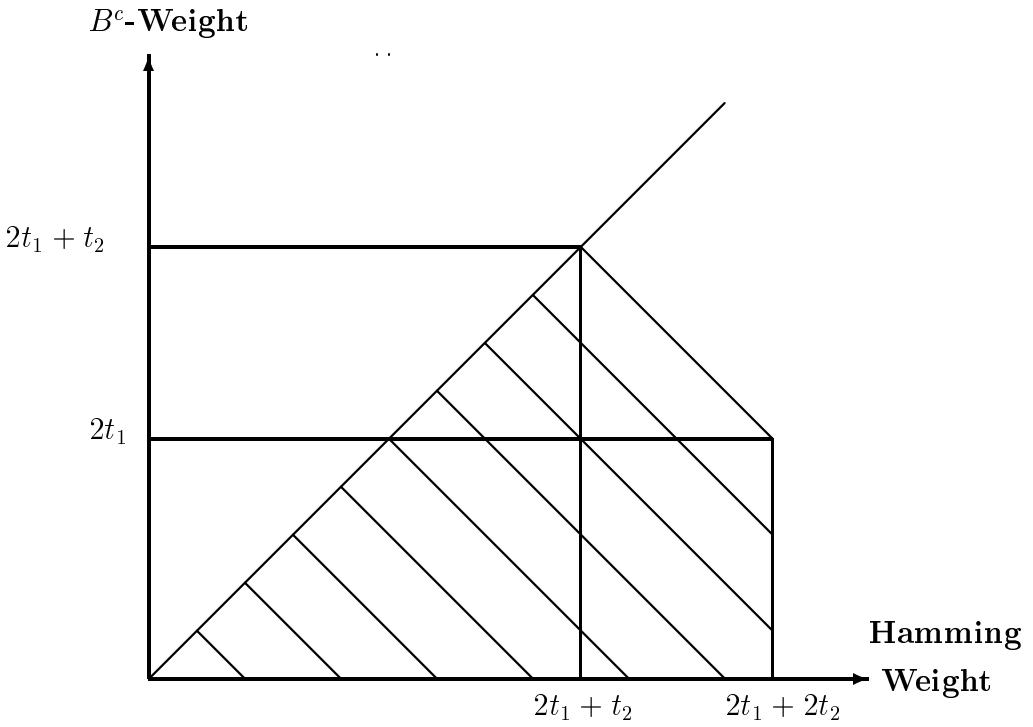

Figure 2.1: Graphic interpretation of the Lemma presenting the sufficient conditions for the construction of (t_1, t_2)-focused codes

2.2 Construction Of Combined Linear Focused Codes

Suppose we intend to construct a code with blocklength n over $GF(2^b)$ that is (t_1, t_2)-focused on the set of odd-weight symbols; that is, the common error set $\mathbf{B}$ consists of all the elements of $GF(2^b)$ with a binary representation containing an odd number of 1's. (Note that this would include the set of single-bit errors.)

A simple way to construct such a code would be to use a concatenated coding scheme as follows. Start with a code over $GF(2^{b-1})$ with minimum distance $2t_1 + t_2 + 1$; then, add an extra bit to each code symbol in every codeword so that every symbol has even parity. To see that the resulting set of n-tuples over $GF(2^b)$ is (t_1, t_2)-focused on the set of odd-weight errors, consider the following decoding algorithm. When an n-tuple over $GF(2^b)$ is received at the decoder, the parity of each symbol is checked; where a parity violation occurs, that symbol is marked as an erasure for decoding by the "outer" code with $d_{min} = 2t_1 + t_2 + 1$. This scheme will correct any combination of $t_1 + t_2$ errors as long as no more than t_1 of those errors have even parity - and thus must be corrected by the outer code without benefit of erasure. This technique constructs codes that have rate $\frac{b-1}{b} R_2$, where R_2 is the rate of the outer code.

The construction described above can be improved as follows. For any n-tuple $\mathbf{x}$ over $GF(2^b)$, let $\mathbf{b(x)}$ be the binary n-tuple obtained by taking the mod-two sum of each component of $\mathbf{x}$.
For example, if $\mathbf{x}$=[0011,0100,1101,1010,1111], then $\mathbf{b(x)}$=[01100]. Then the technique described above can be described as follows. Take a codeword from a code over $GF(2^{b-1})$ and add one bit to each code symbol so that the resulting n-tuple $\mathbf{c}$ satisfies $\mathbf{b(c)=0}$. In this way, we form a (b,b-1) binary parity check code, C_0, that is capable of detecting all the errors lying in $\mathbf{B}$. This technique is something of an "overkill", since it permits us to to flag *every* occurrence of an odd-weight error, while in order to fulfill the code's "mission" we need only to flag up to $t_1 + t_2$ odd-weight errors.

Consider, then, the following construction. Let C_1 be an (n, nR_1) binary *inner* code with minimum distance $d_1 = 2t_1 + 2t_2 + 1$; let C_2 be an (n, nR_2) *outer* code over $GF(2^{b-1})$ with minimum distance $d_2 = 2t_1 + t_2 + 1$. To construct a codeword from our focused code we first take a codeword from C_2 and add one bit to each code symbol such that $\mathbf{c}$, the resulting n-tuple over $GF(2^b)$, satisfies $\mathbf{b}(\mathbf{c}) \in C_1$.

We can check that the code thus constructed is (t_1, t_2)-focused on $\mathbf{B}$ by considering the following decoding algorithm. Given a received 2^b-ary n-tuple $\mathbf{r}$, compute $\mathbf{b(r)}$; find the codeword $\mathbf{x} \in C_1$ that is closest to $\mathbf{b(r)}$. As long as at most $t_1 + t_2$ odd-weight errors have occurred, $\mathbf{x}$ will be equal to $\mathbf{b(c)}$, where $\mathbf{c}$ is the codeword that was actually transmitted. Mark the locations where $\mathbf{x}$ differs from $\mathbf{b(r)}$ as erasures; strip off the last bit in each code symbol and pass the resulting 2^{b-1}-ary n-tuple plus erasure locations to a decoder for C_2. Such a decoder will correct all combinations of $t_1 + t_2$ errors provided at most t_1 of the errors have an even-weight binary representation (i.e., are uncommon).

Note that this improved construction technique adds nR_1 information bits to each codeword, when compared with the simpler construction of the previous section. The overall rate of this code is $\frac{1}{b} R_1 + \frac{b-1}{b} R_2$. More generally, if R_0, R_1 and R_2 are the code rates of C_0, C_1 and C_2 respectively, then the overall rate the focused code is given by:

$$R = (1 - R_0)R_1 + R_0 R_2 \tag{2.1}$$

This technique can be generalized to cover a variety of common error sets; for details, refer to [1].

Chapter 3

Decoder Error Probability

In this chapter, we derive the general expression for the decoder error probability of focused codes and study their performance over "idealized" skewed symmetric channels. We then determine suitable conditions under which the performance of the focused codes is identical to that of the traditional error correcting codes.

3.1 Decoder Block Error Probability Of Focused codes On SSC

We know that a (t_1, t_2)-focused code can correct up to $t_1 + t_2$ errors given that at *most* t_1 errors are uncommon. The decoder block error probability of a (t_1, t_2)-focused code on a skewed symmetric channel (SSC) can thus be written as:

$$P_d = 1 - \sum_{i=0}^{t_1+t_2} \sum_{j=0}^{\min(i,t_1)} \binom{n}{i} \binom{i}{j} \epsilon^i (1-\epsilon)^{n-i} \gamma^j (1-\gamma)^{i-j} \tag{3.1}$$

where:

- ϵ=probability of symbol channel error.
- γ=Pr(error is uncommon | an error has occurred).
- i=number of errors.
- j=number of uncommon errors.
- n=blocklength of a codeword.

P_d can be rewritten as:

$$\begin{aligned} P_d &= \sum_{i=t_1+1}^{t_1+t_2} \sum_{j=t_1+1}^{i} \left(\begin{array}{c} n \\ i \end{array} \right) \left(\begin{array}{c} i \\ j \end{array} \right) \epsilon^i (1-\epsilon)^{n-i} \gamma^j (1-\gamma)^{i-j} \\ &\quad + \sum_{i=t_1+t_2+1}^{n} \left(\begin{array}{c} n \\ i \end{array} \right) \epsilon^i (1-\epsilon)^{n-i} \end{aligned} \tag{3.2}$$

Note that the second term in the above equation is the probability of decoder error of a traditional $t_1 + t_2$-error correcting code.

If ϵ is "reasonably" small, we can approximate P_d by taking only the first terms in the summations of equation (2.2). We get the following:

$$\begin{aligned} P_d &\approx \left(\begin{array}{c} n \\ t_1+1 \end{array} \right) \epsilon^{t_1+1} (1-\epsilon)^{n-t_1-1} \gamma^{t_1+1} \\ &\quad + \left(\begin{array}{c} n \\ t_1+t_2+1 \end{array} \right) \epsilon^{t_1+t_2+1} (1-\epsilon)^{n-t_1-t_2-1} \end{aligned} \tag{3.3}$$

Our *goal* is to make a (t_1, t_2)-focused code perform identically as a traditional $t_1 + t_2$-error correcting code; i.e.,

$$P_d(focused) \approx P_d(traditional)$$

From the above expression, we can make the following observations:

- If γ is small ($\gamma \ll 1$), the second term in the P_d expression is dominant making:
$$P_d(focused) \approx P_d(traditional).$$
- If γ is not small ($\gamma \not\ll 1$), the first term in the P_d expression is dominant making:
$$P_d(focused) \gg P_d(traditional).$$

3.2 Case When ϵ And γ Are Considered Independently

We begin by making the simplifying assumption that ϵ and γ are independent of one another. Specifically, we consider the following question. When one of these two parameters is held constant, what value of the other parameter guarantees that a (t_1, t_2)-focused code performs identically to a $t_1 + t_2$-error correcting code ?

3.2.1 Case When ϵ Is Fixed

For fixed ϵ, we can plot $\log_{10} P_d$ versus $\log_{10} \gamma$. From equation (3.3), we have:

$$\begin{aligned} \log_{10} P_d \;\approx\; & \log_{10} \left[\binom{n}{t_1+1} \epsilon^{t_1+1} (1-\epsilon)^{n-t_1-1} \gamma^{t_1+1} \right. \\ & \left. + \binom{n}{t_1+t_2+1} \epsilon^{t_1+t_2+1} (1-\epsilon)^{n-t_1-t_2-1} \right]. \end{aligned} \tag{3.4}$$

Figures 3.1-3.3 show $\log_{10} P_d$ versus $\log_{10} \gamma$ for a probability of channel symbol error $\epsilon = 10^{-3}$, a codeword block length $n = 50$ and different values of t_1 and t_2.

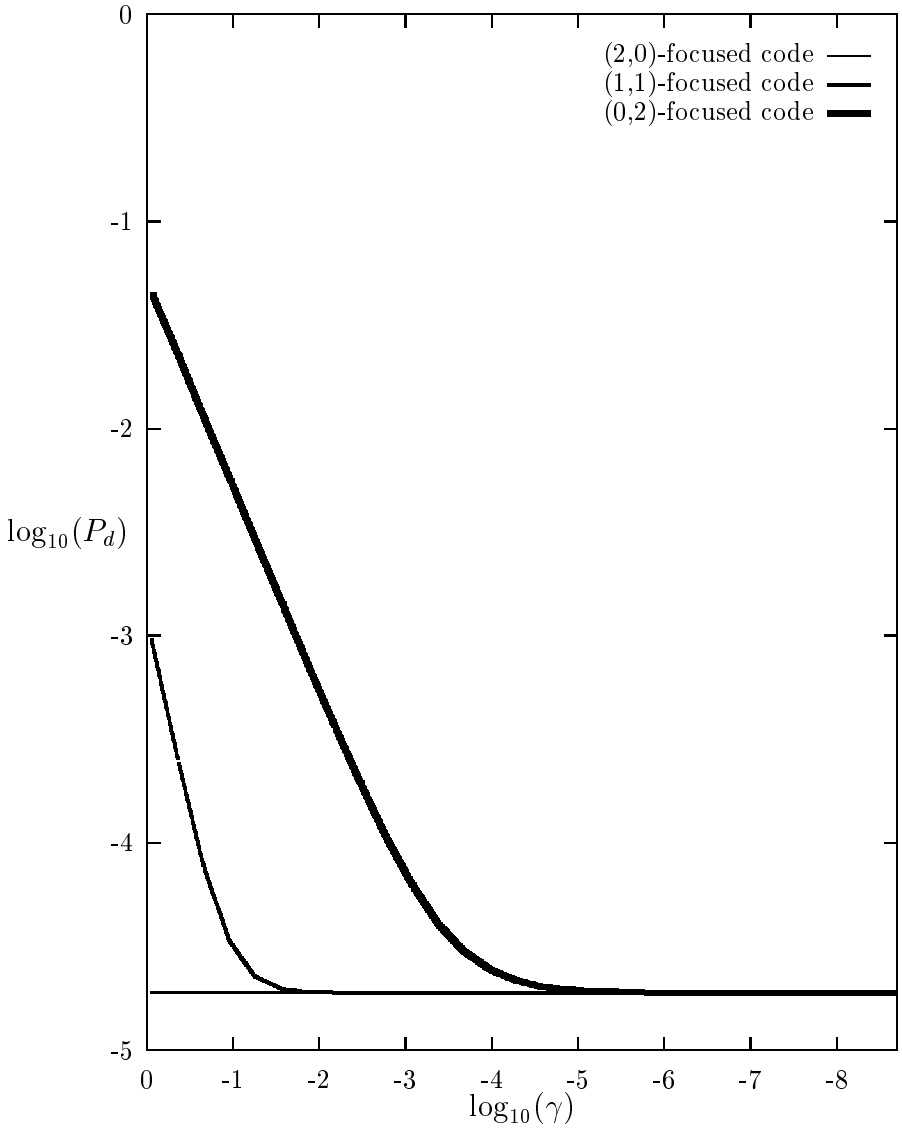

Figure 3.1: P_d vs γ for $t_1 + t_2 = 2$

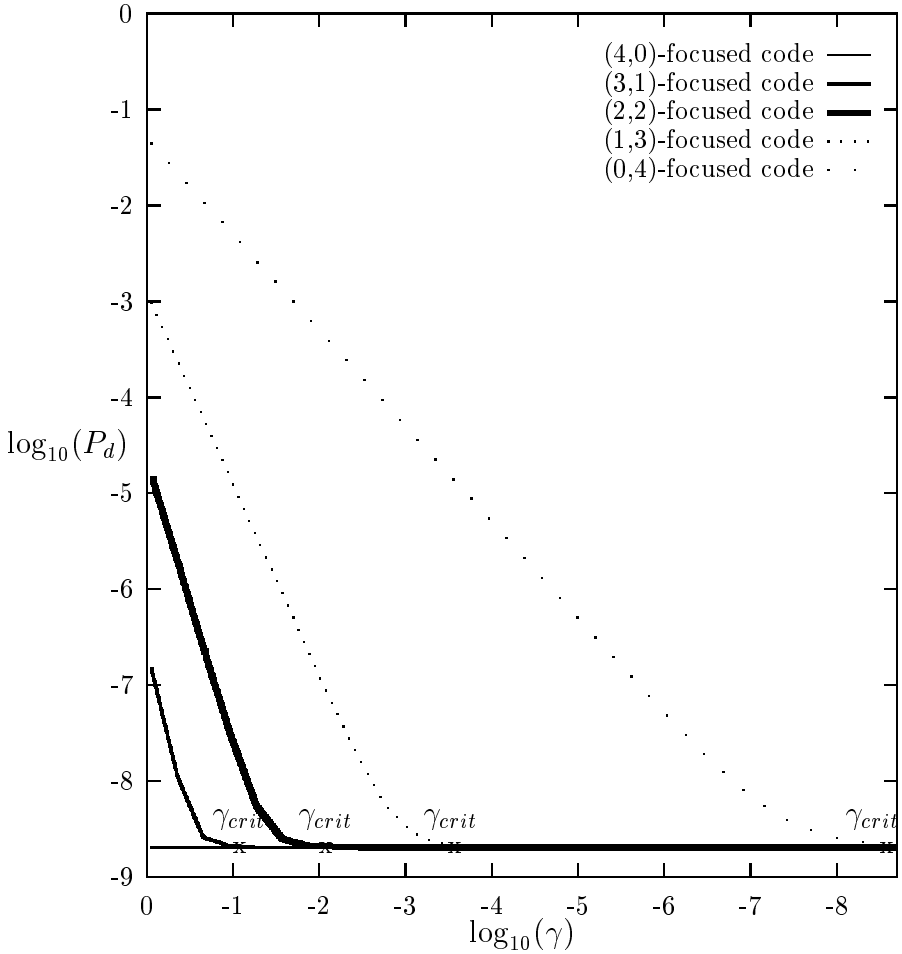

Figure 3.2: P_d vs γ for $t_1 + t_2 = 4$

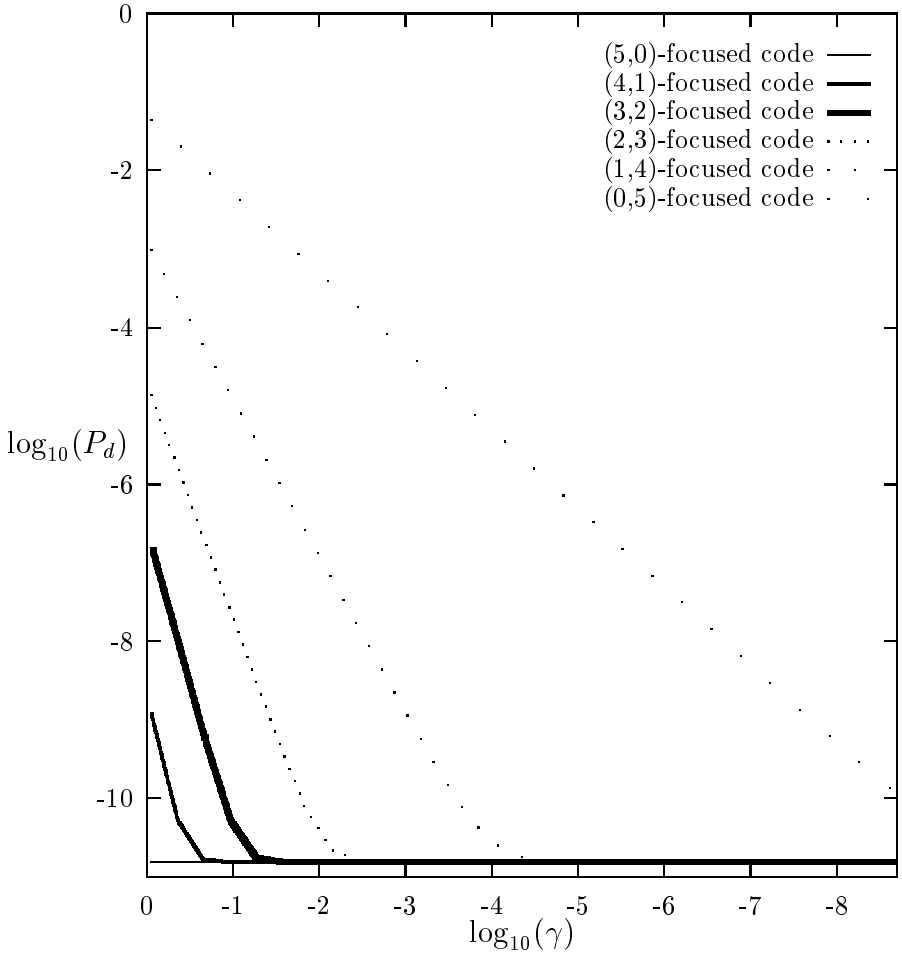

Figure 3.3: P_d vs γ for $t_1 + t_2 = 5$

From the above plots, we can notice that the curve of $\log_{10}(P_d)$ vs $\log_{10}(\gamma)$ is nearly a straight line of slope equal to $t_1 + 1$ for high values of γ, then it converges to a constant as γ becomes smaller and smaller. This can be explained by the fact that as $\gamma \to 1$, equation (3.4) can be approximated by the equation of a straight line of slope $(t_1 + 1)$:

For γ close to 1,

$$\log_{10} P_d \approx (t_1 + 1) \log_{10} \gamma + \log_{10} \left[\binom{n}{t_1 + 1} \epsilon^{t_1+1} (1-\epsilon)^{n-t_1-1} \right] \qquad (3.5)$$

While for $\gamma \to 0$, the dominant term in equation (2.4) is a constant equal to the decoder error probability of a traditional $t_1 + t_2$-error correcting code:

For γ close to 0;

$$\log_{10} P_d \approx \log_{10} \left[\binom{n}{t_1 + t_2 + 1} \epsilon^{t_1+t_2+1} (1-\epsilon)^{n-t_1-t_2-1} \right] \qquad (3.6)$$

This shows us that as γ decreases, the performance of a (t_1, t_2)-focused code converges to that of a traditional $t_1 + t_2$-error correcting code (which is expected since as $\gamma \to 0$, the channel becomes more skewed).

We are interested in determining the critical value γ_{crit} beneath which,

$$P_d(\text{focused code}) \approx P_d(\text{traditional code})$$

We have:

$$\binom{n}{t_1 + 1} \epsilon^{t_1+1} (1-\epsilon)^{n-t_1-1} \gamma^{t_1+1} \approx \binom{n}{t_1 + t_2 + 1} \epsilon^{t_1+t_2+1} (1-\epsilon)^{n-t_1-t_2-1}$$

Yielding:

$$\log_{10} \gamma_{crit} = \frac{1}{t_1 + 1} \left[t_2 \log_{10} \left(\frac{\epsilon}{1-\epsilon} \right) + \log_{10} \left[\frac{\binom{n}{t_1 + t_2 + 1}}{\binom{n}{t_1 + 1}} \right] \right] \qquad (3.7)$$

Thus, given that ϵ is fixed, the performance of a (t_1, t_2)-focused code is identical to that of a $t_1 + t_2$-error correcting code provided $\gamma < \gamma_{crit}$, where γ_{crit} is given in equation (3.7).

3.2.2 Case When γ is Fixed

Using equation (3.4) for a fixed value of γ, we can plot $\log_{10}(P_d)$ versus $\log_{10}(\epsilon)$. The results are shown in Figures 3.4-3.6 for a codeword block length $n = 50$ and different values of γ, t_1 and t_2.

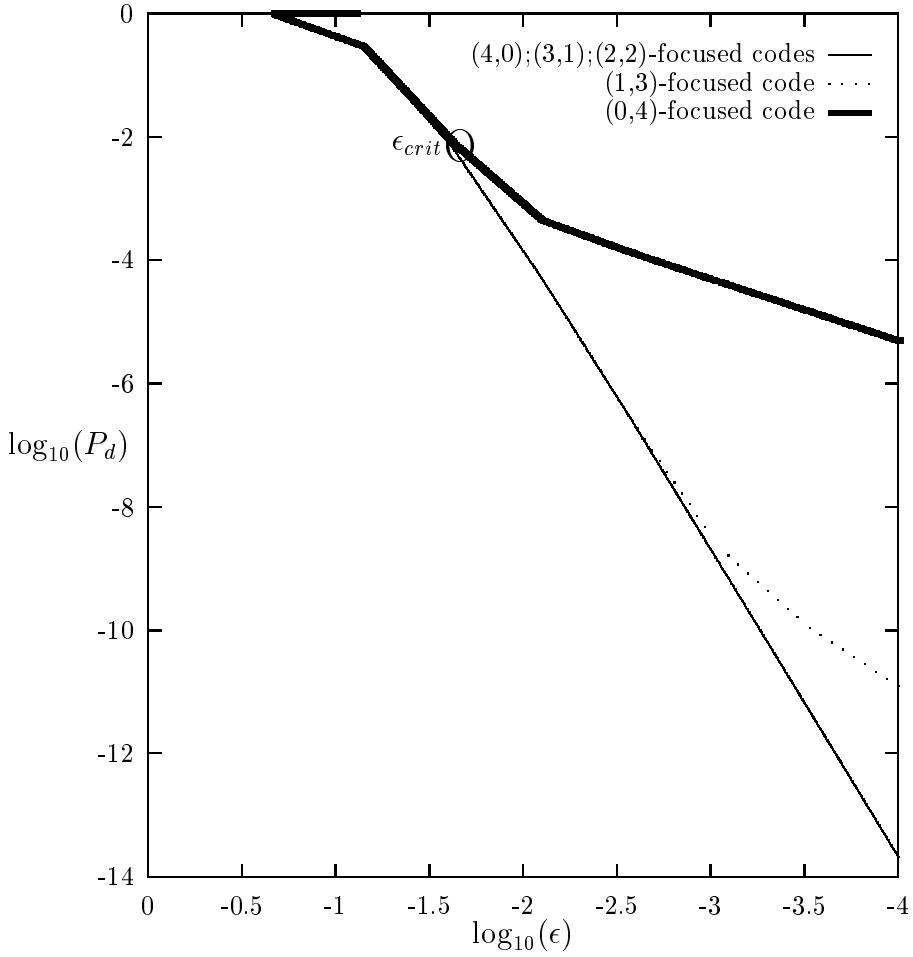

Figure 3.4: P_d vs ϵ for $\gamma = 10^{-3}$ & $t_1 + t_2 = 4$

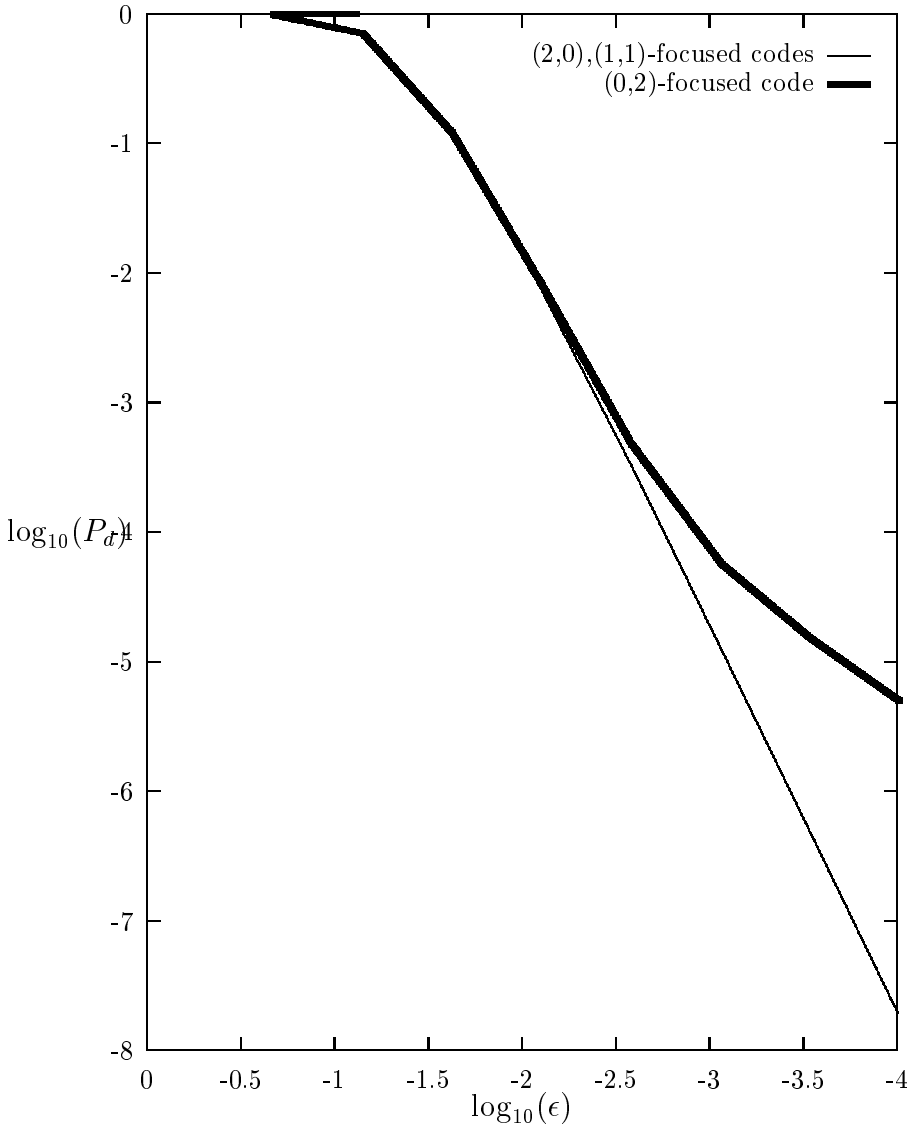

Figure 3.5: P_d vs ϵ for $\gamma = 10^{-3}$ & $t_1 + t_2 = 2$

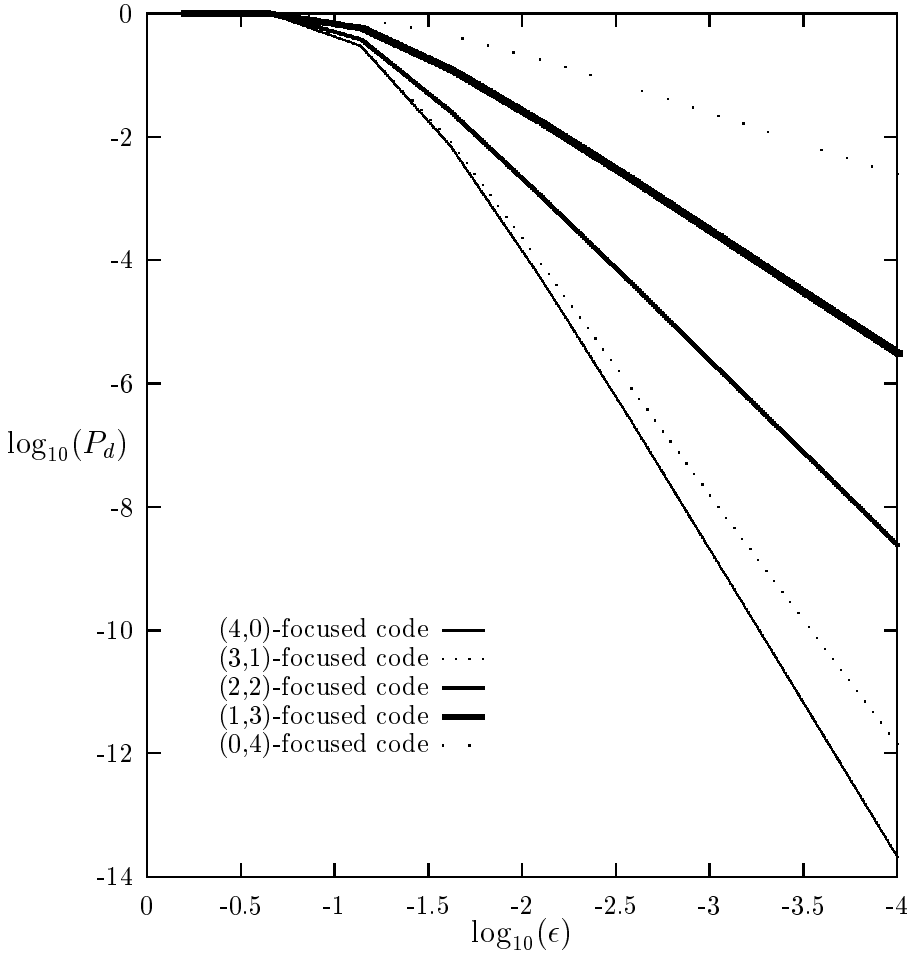

Figure 3.6: P_d vs ϵ for $\gamma = 0.5$ & $t_1 + t_2 = 4$

In a similar way as in the previous case where ϵ was fixed, we can observe from the above plots that the curve of $\log_{10}(P_d)$ versus $\log_{10}(\epsilon)$ is nearly a straight line with slope equal to t_1+1 for small values of ϵ. Analytically, this can be shown from equation (3.3) where its first term is dominant for small values of ϵ while its second term becomes dominant for large values of ϵ.

Thus for small ϵ, we can write:

$$\begin{aligned} P_d &\approx \left(\begin{array}{c} n \\ t_1+1 \end{array} \right) \epsilon^{t_1+1}(1-\epsilon)^{n-t_1-1}\gamma^{t_1+1} \\ &\approx \left(\begin{array}{c} n \\ t_1+1 \end{array} \right) \epsilon^{t_1+1}\gamma^{t_1+1} \end{aligned}$$

Which implies that:

$$\log_{10}(P_d) \approx (t_1+1)\log_{10}\epsilon + \log_{10}\left[\left(\begin{array}{c} n \\ t_1+1 \end{array} \right) \gamma^{t_1+1} \right]$$

For high values of ϵ, the performance of a (t_1, t_2)-focused code matches that of a t_1+t_2-error correcting code since as ϵ increases equation (3.3) reduces to:

$$P_d \approx \left(\begin{array}{c} n \\ t_1+t_2+1 \end{array} \right) \epsilon^{t_1+t_2+1}(1-\epsilon)^{n-t_1-t_2-1}$$

To find the critical value ϵ_{crit} beyond which:

$$P_d(\text{focused code}) \approx P_d(\text{traditional code})$$

We write:

$$\left(\begin{array}{c} n \\ t_1+1 \end{array} \right) \epsilon_{crit}^{t_1+1}(1-\epsilon_{crit})^{n-t_1-1} \approx \left(\begin{array}{c} n \\ t_1+t_2+1 \end{array} \right) \epsilon_{crit}^{t_1+t_2+1}(1-\epsilon_{crit})^{n-t_1-t_2-1}$$

Assuming we are operating in the proper range of ϵ where ϵ is "small"

enough (so that $\frac{\epsilon}{1-\epsilon} \approx \epsilon$), we get:

$$\log_{10} \epsilon_{crit} = \frac{1}{t_2} \left[(t_1 + 1) \log_{10} \gamma + \log_{10} \left[\frac{\begin{pmatrix} n \\ t_1 + 1 \end{pmatrix}}{\begin{pmatrix} n \\ t_1 + t_2 + 1 \end{pmatrix}} \right] \right] \tag{3.8}$$

Thus, given that γ is fixed, the performance of a (t_1, t_2)-focused code is identical to that of a $t_1 + t_2$-error correcting code if $\epsilon > \epsilon_{crit}$ where ϵ_{crit} is given by equation (3.8).

3.3 Generalized Approach For Performance Matching

In many applications, ϵ and γ do not vary independently from each other. For example, in a communication system with a modulation scheme (like QAM or PSK modulations), both ϵ and γ depend on the signal (or energy) to noise ratio.

We now consider a general approach that we call the "benchmark" approach, which is applicable for both cases of dependency or independency between ϵ and γ, in order to determine the condition for which we have performance matching between the focused codes and the traditional correcting codes.

We know that:

1.

$$P_d(\text{traditional code}) \approx \begin{pmatrix} n \\ t_1 + t_2 + 1 \end{pmatrix} \epsilon^{t_1+t_2+1} (1-\epsilon)^{n-t_1-t_2-1}$$

2.

$$P_d(\text{focused code}) \approx \begin{pmatrix} n \\ t_1+1 \end{pmatrix} \epsilon^{t_1+1}(1-\epsilon)^{n-t_1-1}\gamma^{t_1+1} + \begin{pmatrix} n \\ t_1+t_2+1 \end{pmatrix} \epsilon^{t_1+t_2+1}(1-\epsilon)^{n-t_1-t_2-1}$$

So $P_d(focused) \to P_d(traditional)$ if and only if:

$$\begin{pmatrix} n \\ t_1+t_2+1 \end{pmatrix} \epsilon^{t_1+t_2+1}(1-\epsilon)^{n-t_1-t_2-1} \gg \begin{pmatrix} n \\ t_1+1 \end{pmatrix} \epsilon^{t_1+1}(1-\epsilon)^{n-t_1-1}\gamma^{t_1+1}$$

Or,

$$\frac{\begin{pmatrix} n \\ t_1+t_2+1 \end{pmatrix}}{\begin{pmatrix} n \\ t_1+1 \end{pmatrix}} \gg \left(\frac{\epsilon}{1-\epsilon}\right)^{t_2} \gamma^{t_1+1}$$

If we let $\theta = \frac{\epsilon}{1-\epsilon}$, we get:

$$\left[\frac{\gamma^{t_1+1}}{\theta^{t_2}}\right] \left[\frac{\begin{pmatrix} n \\ t_1+1 \end{pmatrix}}{\begin{pmatrix} n \\ t_1+t_2+1 \end{pmatrix}}\right] \ll 1$$

Now, defining our benchmark as β:

$$\beta \stackrel{\text{def}}{=} \left[\frac{\gamma^{t_1+1}}{\theta^{t_2}}\right] \left[\frac{\begin{pmatrix} n \\ t_1+1 \end{pmatrix}}{\begin{pmatrix} n \\ t_1+t_2+1 \end{pmatrix}}\right] \tag{3.9}$$

We therefore obtain performance matching iff:

$$\beta \ll 1$$

Usually, it is good enough to have:

$$\beta \ll 10^{-2}$$

Or

$$\log_{10}(\beta) \ll -2 \tag{3.10}$$

Chapter 4

Focused Codes Used In Conjunction With PSK Modulation

In this chapter, we demonstrate how the results of chapter 3 can be applied to a communication system employing non-binary phase shift keying (PSK).

4.1 PSK Modulation

If we use M-ary PSK modulation with a Gray code so that the difference between the binary representation of any two adjacent signals is one bit, then an additive Gaussian noise channel can be approximated by an M-ary SSC for the focus set $\mathbf{B}$ consisting of all elements with a binary representation containing exactly one "1".

In order to compute the block decoder error probability P_d for a focused code used in conjunction with PSK modulation, we need to determine the parameters of the associated SSC which are:

- ϵ, the probability of channel symbol error.
- γ, the probability of an uncommon error given that an error occurred.

The general analytic expression of an M-ary phase shift keying (PSK) signal is [2]:

$$s_i(t) = \begin{cases} \sqrt{2E_s/T}\,\cos(w_0 t + \phi_i(t)) & \text{if } 0 \le t \le T \\ 0 & \text{otherwise} \end{cases}$$

where,

- w_0 is a given radian frequency of the carrier signal.
- $i = 1, 2, ..., M$
- The phase term $\phi_i(t) = \frac{2\pi i}{M}$; $i = 1, ..., M$.
- T is the symbol duration.
- E_s is the symbol energy.

An M-ary PSK signal needs a set of two orthonormal basis functions for its two-dimension vectorial representation:

$$\psi_1(t) = \begin{cases} \sqrt{2/T}\,\cos w_0 t & \text{if } 0 \le t \le T \\ 0 & \text{otherwise} \end{cases}$$

and

$$\psi_2(t) = \begin{cases} \sqrt{2/T}\,\sin w_0 t & \text{if } 0 \le t \le T \\ 0 & \text{otherwise} \end{cases}$$

The computation of the symbol transmission error probability can be simplified by the complete symmetry the M-ary PSK signal sets demonstrate.

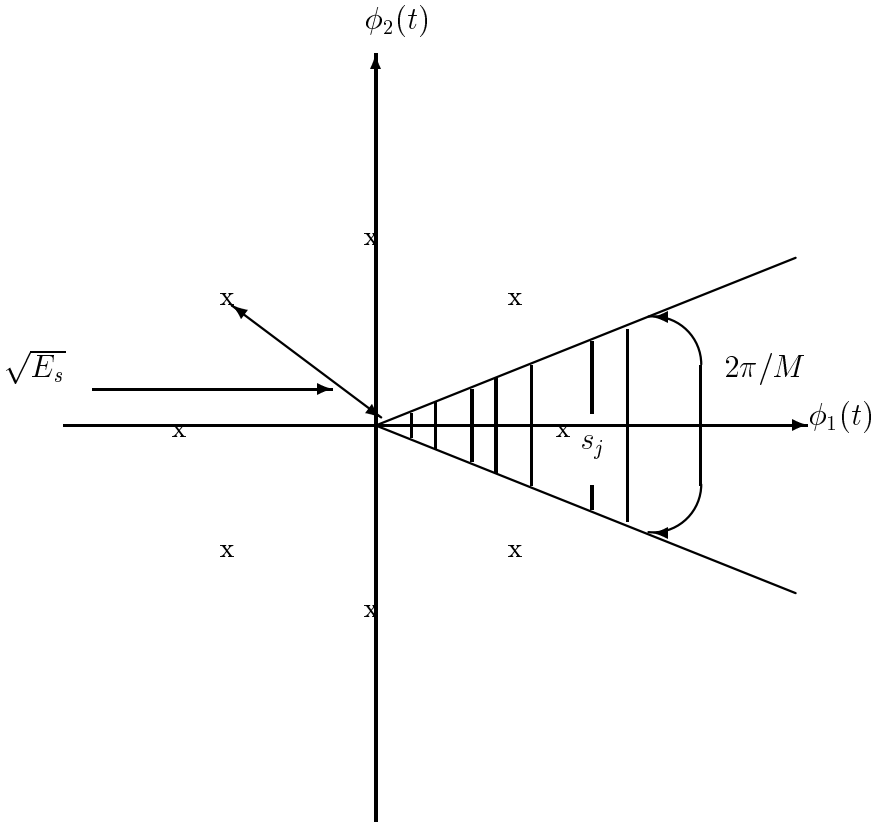

Figure 4.1: Decision region corresponding to an M-ary PSK signal set

We can therefore write [3], for any j,

$$\epsilon = 1 - \int_{R_j} f_{\theta_j} dx \tag{4.1}$$

where:

- R_j is the decision region of the signal vector s_j shown in Figure 4.1.
- θ_j represents the phase displacement of the received signal from the transmitted one.
- f_{θ_j} is the probability density function of θ_j.

For $M \geq 4$ and the energy to noise ratio $E_s/N_0 \gg 1$, an approximation of the probability density function (pdf) $f_\theta(x)$ can be obtained [3]:

$$f_\theta(x) \approx \sqrt{E_s/(\pi N_0)}\, \cos x\, \exp[-(E_s/N_0)\, \sin^2 x] \tag{4.2}$$

where $x \in [-\pi, \pi]$.

An error is made if, for any j, the noise causes a phase displacement greater than π/M in absolute value, corresponding to a received phase lying outside the j'th decision region. Therefore,

$$\begin{aligned} \epsilon &\approx 1 - \int_{-\pi/M}^{\pi/M} \sqrt{E_s/(\pi M)}\, \cos x\, \exp[-(E_s/N_0)\, \sin^2 x]\, dx \\ &\approx 2\, Q\left(\sqrt{2\, E_s/N_0}\, \sin \frac{\pi}{M}\right) \end{aligned} \tag{4.3}$$

where,

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty \exp[-t^2/2]\, dt \tag{4.4}$$

and E_s = energy per symbol.

We now turn our attention to computing γ. We begin by considering M=4 - i.e., QPSK modulation.

From equation(4.3), we have that $M = 4$ yielding:

$$\epsilon = 2\, Q\left(\sqrt{\frac{E_s}{N_0}}\right)$$

We want to compute:

$$\gamma = P(\text{uncommon error/an error occurred}) = \frac{P(\text{uncommon error})}{\epsilon}$$

Since we are using a Gray code, we remark that if a signal is sent, the common error that may be produced by the detection of the corresponding received signal, will be the detection of one of the two signals directly adjacent to the

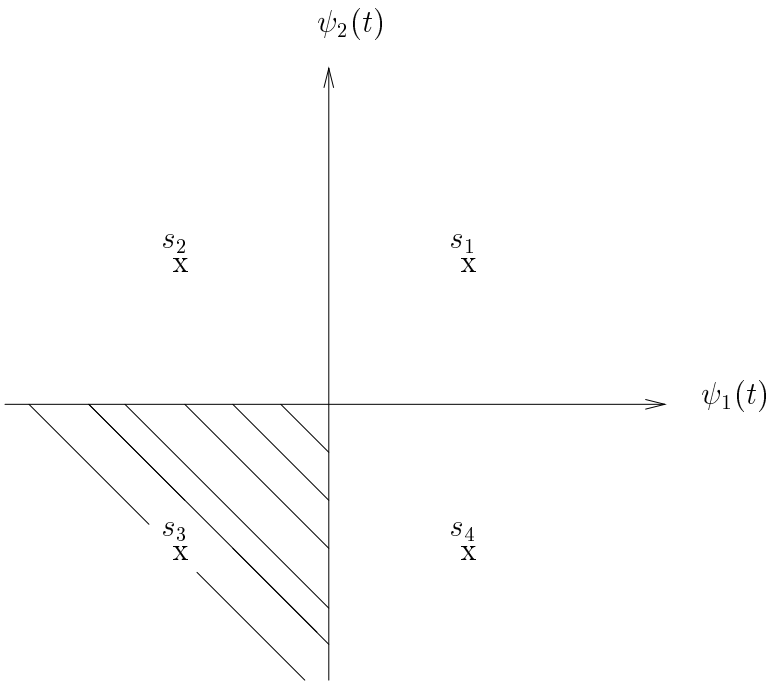

Figure 4.2: QPSK signal set representation

transmitted one in the signal constellation. So for our QPSK case, if s_1 is sent, an uncommon detection error is to detect s_3 (Figure 4.2). By symmetry we can write:

$$P(\text{uncommon error}) = P(s_3 \text{ detected } / \ s_1 \text{ sent}) \tag{4.5}$$

The coordinates in terms of the basis functions $\psi_1(t)$ and $\psi_2(t)$, of the QPSK signals s_j $(j = 1, ..., 4)$ are the following:

- s_1 $\left(\sqrt{\frac{E_s}{2}}, \sqrt{\frac{E_s}{2}}\right)$
- s_2 $\left(-\sqrt{\frac{E_s}{2}}, \sqrt{\frac{E_s}{2}}\right)$
- s_3 $\left(-\sqrt{\frac{E_s}{2}}, -\sqrt{\frac{E_s}{2}}\right)$

- $s_4 \left(\sqrt{\frac{E_s}{2}}, -\sqrt{\frac{E_s}{2}}\right)$

Let x_1 be the received signal corresponding to the transmitted signal s_1. We have $x_1(t) = s_1(t) + n(t)$,

$$\Rightarrow x_1 = \left(\sqrt{\frac{E_s}{2}} + n_1, \sqrt{\frac{E_s}{2}} + n_2\right)$$

where n_1 and n_2 are independent and identically distributed (iid) additive white Gaussian noise (AWGN) random variables with variance equal to $N_0/2$ and zero mean.

$$\text{Detecting } s_3 \Rightarrow \begin{cases} \sqrt{\frac{E_s}{2}} + n_1 \leq 0 \\ \sqrt{\frac{E_s}{2}} + n_2 \leq 0 \end{cases}$$

$$\begin{aligned} P(\text{uncommon error}) &= P\left(n_1 \leq -\sqrt{\frac{E_s}{2}}, n_2 \leq -\sqrt{\frac{E_s}{2}}\right) \\ &= P\left(n_1 \leq -\sqrt{\frac{E_s}{2}}\right) P\left(n_2 \leq -\sqrt{\frac{E_s}{2}}\right) \\ &= \left[P\left(n_1 \leq -\sqrt{\frac{E_s}{2}}\right)\right]^2 \end{aligned}$$

but

$$P\left(n_1 \leq -\sqrt{\frac{E_s}{2}}\right) = P\left(n_1 \geq \sqrt{\frac{E_s}{2}}\right) = Q\left(\sqrt{\frac{E_s}{N_0}}\right)$$

where $Q(x) = \frac{1}{\sqrt{2\,\pi}} \int_x^\infty \exp[-u^2/2]\, du$

$$\Rightarrow P(\text{uncommon error}) = Q^2\left(\sqrt{\frac{E_s}{N_0}}\right)$$

Thus,

$$\gamma = \frac{P(\text{uncommon error})}{\epsilon} = \frac{Q^2\left(\sqrt{\frac{E_s}{N_0}}\right)}{2\,Q\left(\sqrt{\frac{E_s}{N_0}}\right)}$$

$$\Rightarrow \gamma = \frac{1}{2} Q\left(\sqrt{\frac{E_s}{N_0}}\right) \tag{4.6}$$

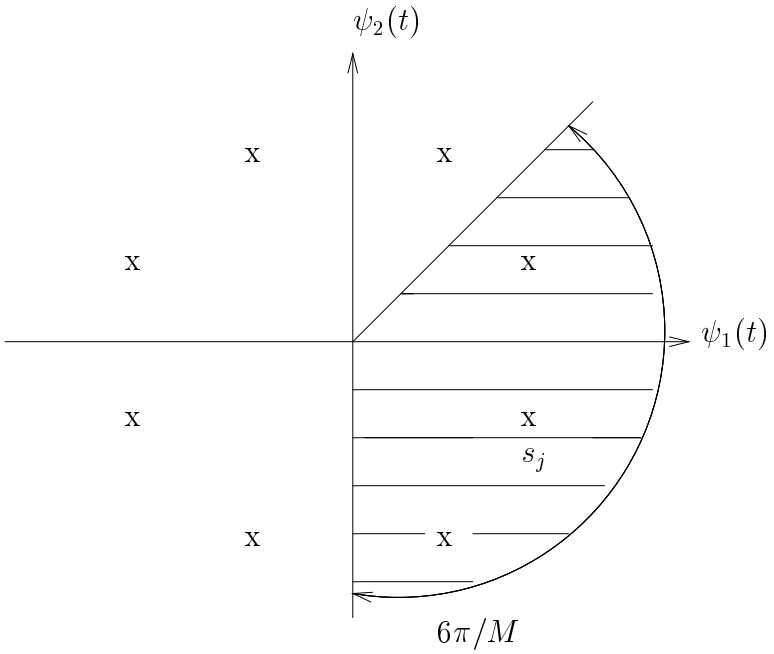

Figure 4.3: Decision region corresponding to a correct or a common error detection for an M-ary PSK signal set

For $M \geq 8$ and $E_s/N_0 \gg 1$, we can use the same expression of the pdf of the received M-ary PSK signal as at the beginning of this section given by equation (4.2).

An uncommon error is made if, for any j, the noise causes a phase displacement greater than $3\pi/M$ in absolute value, corresponding to a received phase lying outside the j'th decision region (which is the dashed area shown in Figure (4.3)).

Due to symmetry,

$$\begin{aligned} P(\text{uncommon error}) &= 1 - P(\text{uncommon error / } s_j \text{ sent}) \\ &= 1 - P(s_j \in \text{shaded area in Fig. 4.3}) \\ &= 1 - \int_{-3\pi/M}^{3\pi/M} f_\theta(x)\, dx \\ &= 1 - 2\int_{0}^{3\pi/M} f_\theta(x)\, dx \end{aligned}$$

where,

$$f_\theta(x) = \sqrt{E_s/(\pi N_0)}\, \cos x\, \exp[-(E_s/N_0)\sin^2 x]$$

We get:

$$P(\text{uncommon error}) = 2\,Q\left(\sqrt{\frac{2E_s}{N_0}}\, \sin\frac{3\pi}{M}\right) \tag{4.7}$$

Therefore,

$$\begin{aligned} \gamma &= \frac{P(\text{uncommon error})}{\epsilon} \\ &= \frac{Q\left(\sqrt{\frac{2E_s}{N_0}}\, \sin\frac{3\pi}{M}\right)}{Q\left(\sqrt{\frac{2E_s}{N_0}}\, \sin\frac{\pi}{M}\right)} \end{aligned} \tag{4.8}$$

Regrouping all the results we previously obtained, we have:

$$\epsilon = 2\,Q\left(\sqrt{\frac{2E_s}{N_0}}\, \sin\frac{\pi}{M}\right) \tag{4.9}$$

$$\gamma = \begin{cases} \frac{1}{2} Q\left(\sqrt{\frac{E_s}{N_0}}\right) & \text{if } M = 4 \\ \dfrac{Q\left(\sqrt{\frac{2E_s}{N_0}}\, \sin\frac{3\pi}{M}\right)}{Q\left(\sqrt{\frac{2E_s}{N_0}}\, \sin\frac{\pi}{M}\right)} & \text{if } M \geq 8 \end{cases} \tag{4.10}$$

where,

- $Q(y) = \frac{1}{\sqrt{2\pi}} \int_y^\infty \exp[-u^2/2]\, du$
- E_s = symbol energy.

In order to study the performance of the focused codes used in association with PSK modulation, we need to determine a range of the symbol energy to noise ratio (E_s/N_0) under which we can "suitably" operate.

A suitable range for E_s/N_0 would be the one corresponding to a channel symbol error probability (ϵ) less than $\frac{1}{2}$ but exceeding at least 10^{-5}:

$$10^{-5} \leq \epsilon < 0.5$$

Using equation (4.9), we get:

$$\frac{0.22445}{\sin^2(\pi/M)} < \frac{E_s}{N_0} \leq \frac{9.82}{\sin^2(\pi/M)} \tag{4.11}$$

4.2 Performance Of Focused Codes Using PSK Modulation

4.2.1 Benchmark And P_d vs E_s/N_0 Plots

Applying equations (3.2), (3.9), (4.9), (4.10) and selecting a suitable range for E_s/N_0 with the help of equation (4.11) while keeping a reasonable value of P_d (not exceeding 10^{-11}), we plot the curves of P_d versus E_s/N_0 and benchmark β versus E_s/N_0 for different values of M, n, t_1 and t_2.

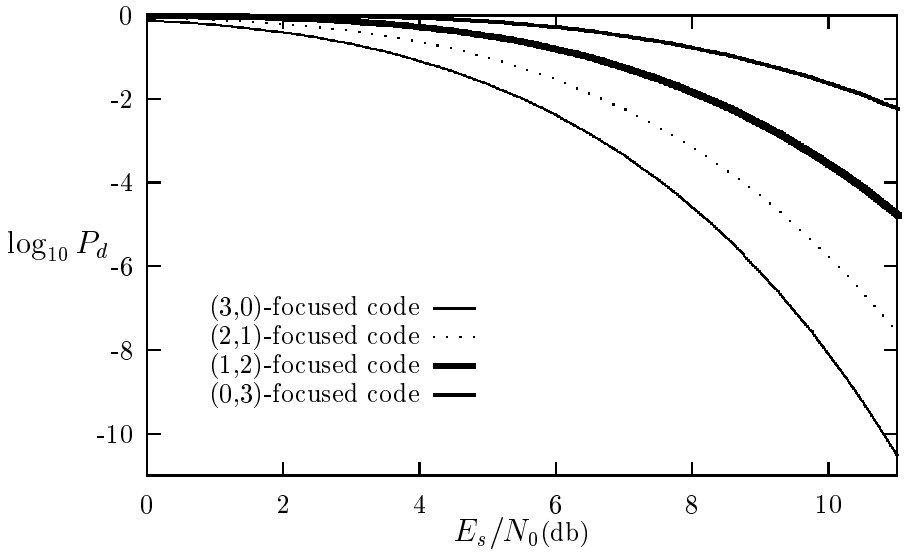

Figure 4.4: P_d vs E_s/N_0 for $M = 4$ PSK, $n = 15$, $t_1 + t_2 = 3$

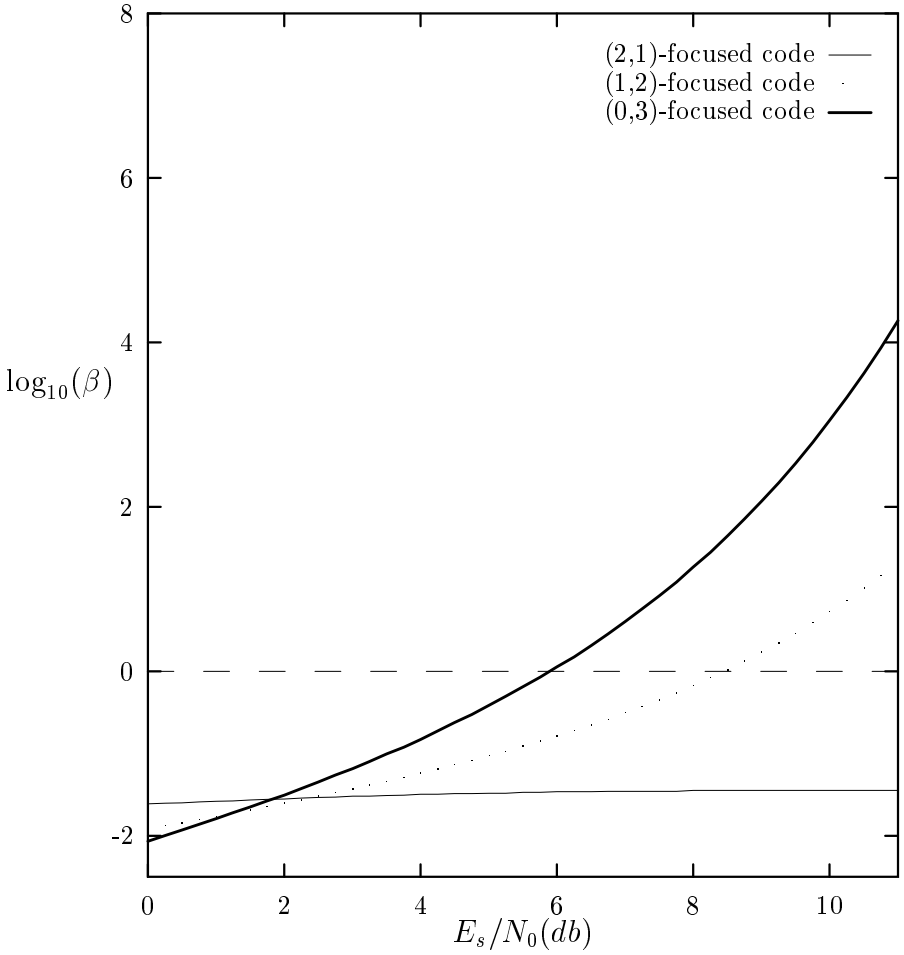

Figure 4.5: Benchmark plot for $M = 4$, $n = 15$ PSK, $t_1 + t_2 = 3$

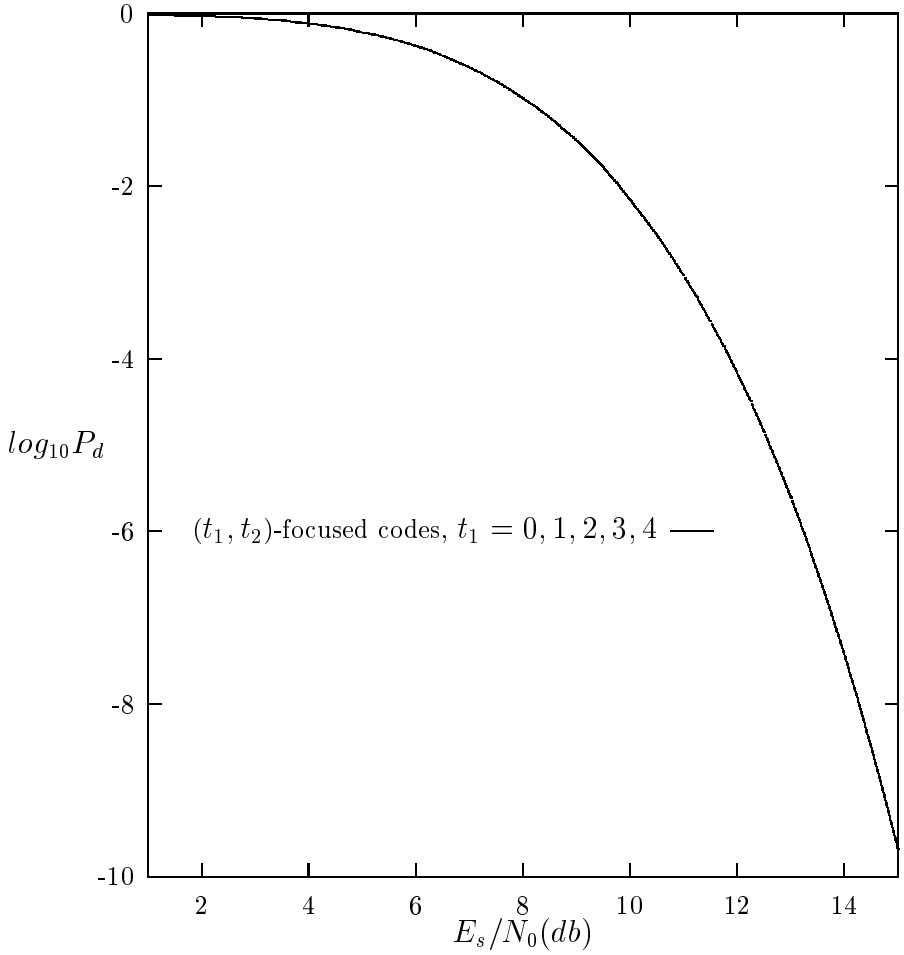

Figure 4.6: P_d vs E_s/N_0 for $M = 8$ PSK, $n = 15$, $t_1 + t_2 = 4$

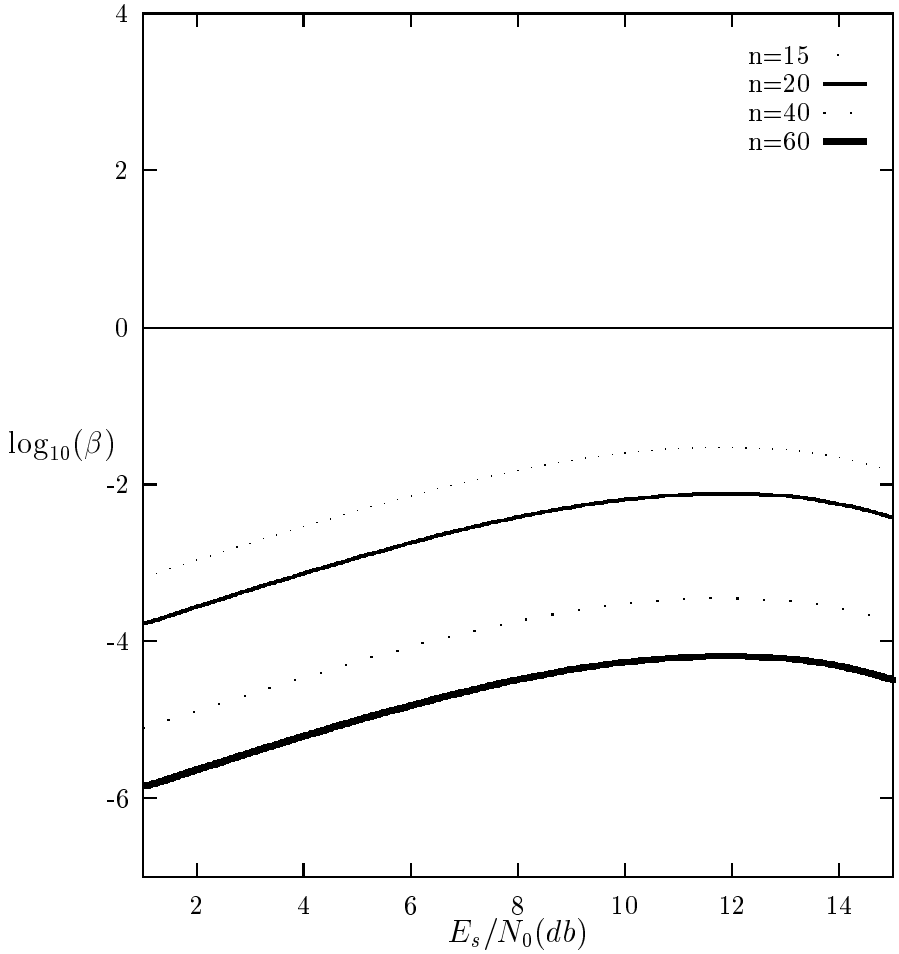

Figure 4.7: Benchmark plot for $M = 8$ PSK, (0,4)-focused code

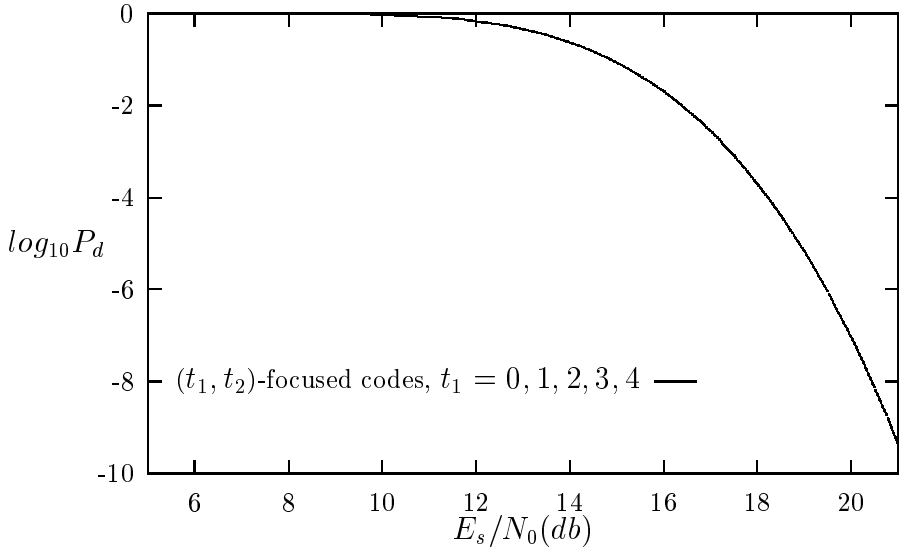

Figure 4.8: P_d vs E_s/N_0 for $M = 16$ PSK, $n = 20$, $t_1 + t_2 = 4$

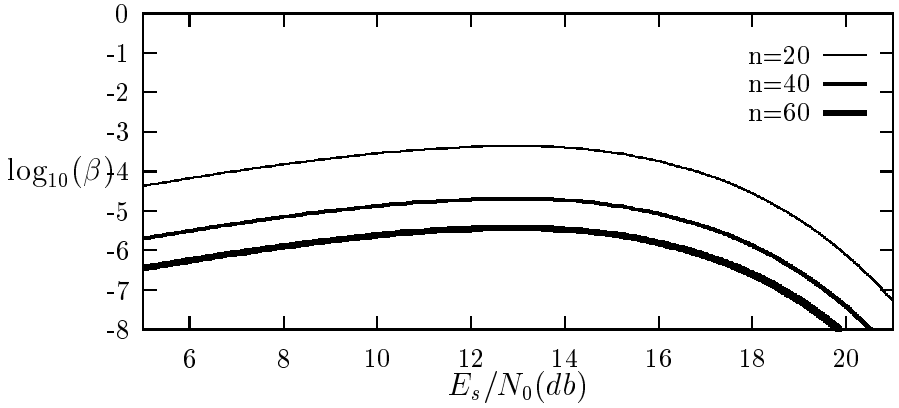

Figure 4.9: Benchmark plot for $M = 16$ PSK, (0,4)-focused code

4.2.2 Evaluation Of The Plots

The above plots illustrate for us the condition given by equation (3.10) for performance matching between focused codes and traditional codes. In fact we can remark from figures (4.6), (4.7), (4.8) and (4.9) that as long as the benchmark of a (t_1, t_2)-focused code is less than 10^{-2}, its performance matches that of $t_1 + t_2$-error correcting code (which is a $(t_1 + t_2, 0)$-focused code) otherwise there is no performance matching (figure (4.4)).

We note also that if a completely focused code (i.e., a (0,t)-focused code) performs identically to a t-error correcting code, then so do any (t_1, t_2)-focused code (where $t_1 + t_2 = t$) because a completely focused code obviously performs worse than a partially focused code (figure (4.5)).

We can furthermore notice from figures (4.7) and (4.9), that as the block length n of the codeword increases, the benchmark decreases; yielding a stronger matching between the focused codes and the traditional codes and a better performance. This can be verified by examining equation (3.9) where the denominator becomes much bigger than the numerator as n increases substantially; making the benchmark β to decrease.

Applying the same equations we used for the plots, we obtain that *any* (t_1, t_2)-focused code performs identically to a t-error correcting code such that $t_1 + t_2 = t$, for the following values of M and t:

- $M = 8$; $t = 1, 2, 3$ for all E_s/N_0 and all $n \geq 7$; $t = 4$ for all E_s/N_0 and all $n \geq 15$.
- $M = 16$; $t = 1, 2, 3, 4, 5, 6$ for all E_s/N_0 and all $n \geq 10$; $t = 7$ for all E_s/N_0 and all $n \geq 25$.

In conclusion, we can state that we will always obtain a performance matching between focused codes and traditional codes using PSK modulation as long as condition (3.10) holds.

4.3 Simulation Of The Focused Codes Performance Using PSK Modulation

4.3.1 Simulation Description

In order to check the truthfulness of equations (4.3), (4.6) and (4.8) that we derived analytically in the first section of this chapter, we use a C simulation to evaluate the performance (P_d vs E_s/N_0) of a (0,3)-focused code with a block length n equal to 15 associated with a 16-ary PSK modulation, in the presence of an additive white Gaussian noise (AWGN).

We present a brief description of the simulation algorithm:

1. For a specific value of E_s/N_0 and given that signal $s_1(\sqrt{E_s}, 0)$ is sent, determine the ranges of the phase angles covering the correct detection region R_c, the common error detection region R_{ce} and the uncommon error detection region R_{ue} respectively. Set to zero the counter c_w of

the number of codeword errors, the counter c_e of the number of signal errors and the counter c_{ue} of the number of uncommon signal errors.

2. Generate successively two AWGN random variables with zero mean and variance equal to $N_0/2$ (where N_0 is given) and add them respectively to the x-y coordinates of s_1 to obtain the coordinates of the received noisy signal. Measure the phase angle of the received signal ($\phi = \arctan(y/x)$);

   - If $\phi \in R_{ue}$, increment c_e and c_{ue} by 1.
   - If $\phi \in R_{ce}$,increment c_e by 1.
   - If $\phi \in R_c$, proceed to the next step.

3. Repeat step 2 n times to generate a received vector of length n.

4. - If $c_e \geq t_1 + t_2 + 1$, increment c_w by 1 and go to step 5.
   - If $c_e \leq t_1 + t_2 + 1$ and $c_{ue} \geq t_1 + 1$, increment c_w by 1 and go to step 5.

5. Repeat steps 1 to 4 a large number L of times (on the order of 10^8) and calculate $P_d = c_w/L$.

6. Repeat steps 1 to 5 for several values of E_s/N_0 and get the simulated plot.

4.3.2 Simulation Results

We use the above simulation program (complete C program is available from the author) for 14 different values of E_s/N_0 on a (t_1, t_2)-focused code (of block length n) associated with an M-ary PSK modulation, where:

- $M = 16$, $n = 15$.
- $t_1 = 0$, $t_2 = 3$.
- $N_0 = (2)(10^{-6})$.

- Number of generated codewords: L of the order of 10^8.

We get the following simulated results:

E_s/N_0	P_d	L
9.00	-0.0195	100000
10.00	-0.0469	100000
11.00	-0.1043	100000
12.00	-0.2061	100000
13.00	-0.3742	100000
14.00	-0.6292	100000
15.00	-1.0101	100000
16.00	-1.5321	100000
17.00	-2.2298	100000
18.00	-3.1478	1000000
19.00	-4.3392	1000000
20.00	-5.8298	10000000
21.00	-7.6899	100000000

Table 4.1: Simulation results of the performance of a 16-ary PSK modulated (0,3)-focused code with $n = 15$

From the results shown in table (4.1) and figure (4.10), we can clearly remark that the simulation results match perfectly the performance plots that we obtained analytically.

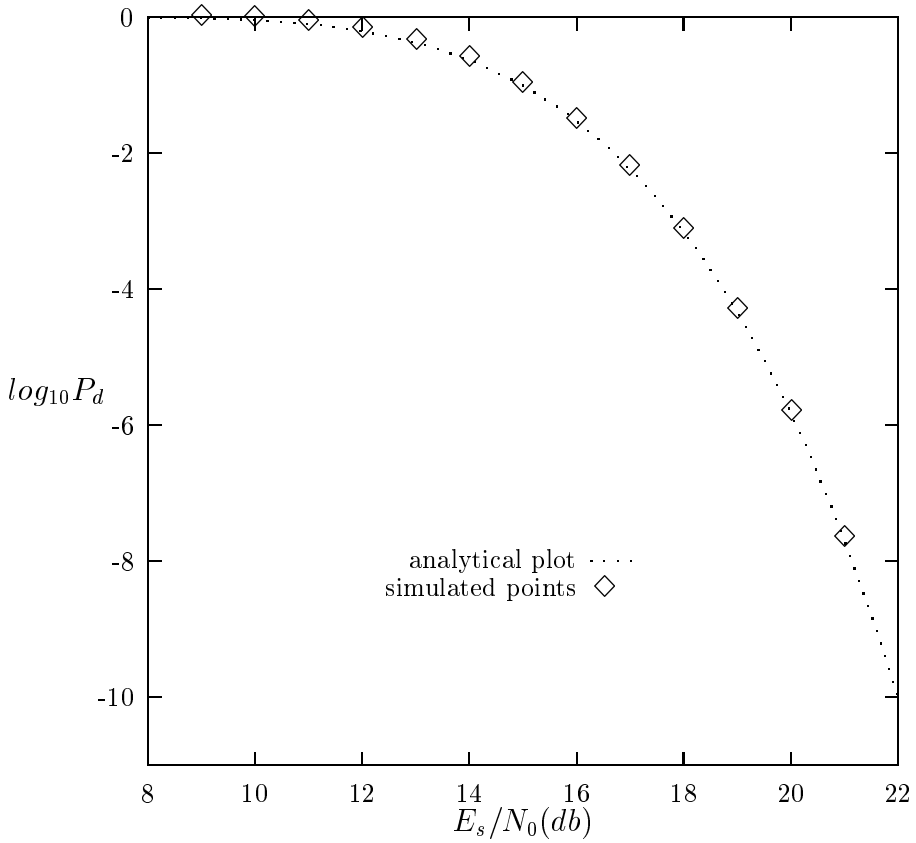

Figure 4.10: Simulation vs analytical performance plots for $M = 16$ PSK, $n = 15$, $(0, 3)$-focused code

Chapter 5

Focused Codes Used With Square Constellations

In this chapter, we demonstrate how the results of chapter 3 can be applied to communication systems using square constellations.

5.1 Square Constellations(QAM)

An additive white Gaussian noise (AWGN) channel and an M-ary square signal constellation can be approximated by an M-ary SSC channel with parameters ϵ and γ to be determined. For the square constellation, we assume that we can use Gray coding so that a common error caused in detecting a transmitted signal s_1 is due to the detection of one of the *vertical* or *horizontal* signals directly adjacent to s_1 in the signal constellation.

For example , for the case where $M = 16$ shown in figure (5.1), if we suppose that signal s_1 is sent, then a common error committed in detecting s_1 will be to detect either s_2, s_3, s_4 or s_5.

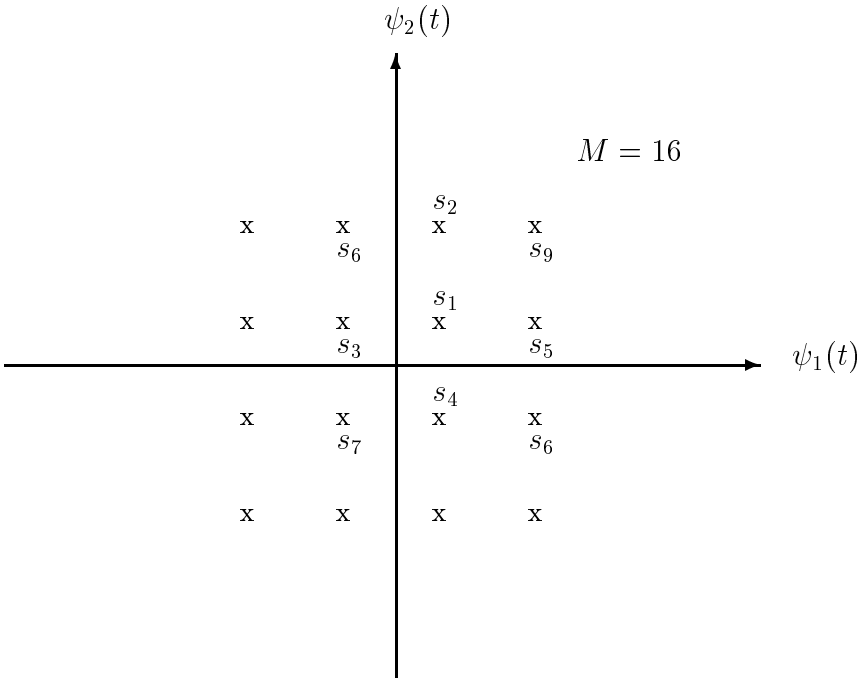

Figure 5.1: 16-ary square constellation

The general analytic expression of an M-ary quadrature amplitude modulation (QAM) signal is [2]:

$$s_i(t) = \begin{cases} \sqrt{2E_i/T}\,\cos(w_0 t + \phi_i(t)) & \text{if } 0 \le t \le T \\ 0 & \text{otherwise} \end{cases}$$

where:

- E_i is an indexed energy per symbol; $i = 1, 2, ..., M$.
- The phase term $\phi_i(t) = 2\pi i/M$; $i = 1, 2, ..., M$.
- T is the symbol duration.

Using the same two orthonormal signals $\psi_1(t)$ and $\psi_2(t)$ we used in section (4.1) as basis functions, we arrange the set of M symbols in the two-dimensional signal space in a rectangular constellation.

For a square ($\sqrt{M} \times \sqrt{M}$ where $\log_2 M$ is even) constellation of a QAM modulated pattern, the x-y coordinates of the signals with respect to the basis functions $\psi_1(t)$ and $\psi_2(t)$ are:

$$x_i = (2i - 1 - \sqrt{M})\frac{d}{2}$$

and

$$y_j = (2j - 1 - \sqrt{M})\frac{d}{2}$$

where,

- d is a constant distance between any two signals.
- $i, j = 1, 2, ..., \sqrt{M}$.

In order to compute the probability of symbol channel error ϵ, we first need to determine the *average* energy per symbol E_s as a function of d. Due to complete symmetry, we can simplify our work by considering only one quadrant ($M/4$ signals). We have:

$$E_s = \frac{1}{M/4} \sum_{i=1}^{\sqrt{M/2}} \frac{\sqrt{M}}{2} (x_i^2 + y_i^2)$$

but $x_i = y_i$,

$$\begin{aligned} \Rightarrow E_s &= \frac{4}{M} \sum_{i=1}^{\sqrt{M/2}} \sqrt{M}\, x_i^2 \\ &= \frac{d^2}{\sqrt{M}} \sum_{i=1}^{\sqrt{M/2}} (2i - 1 - \sqrt{M})^2 \end{aligned}$$

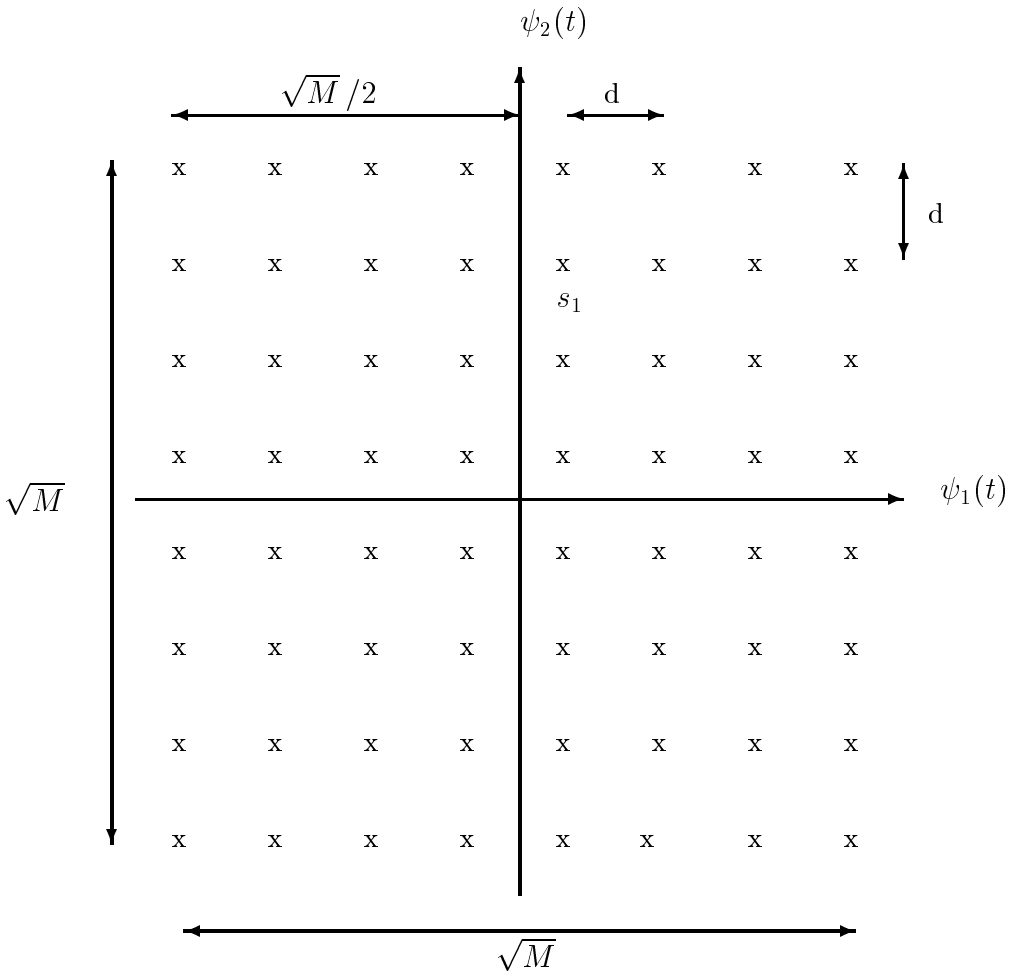

Figure 5.2: M-ary square constellation

Using the following identities:

- $\sum_{i=1}^{n} i = n(n+1)/2$.
- $\sum_{i=1}^{n} i^2 = n(n+1)(2n+1)/6$.

we get:

$$E_s = \frac{M-1}{6}\, d^2 \tag{5.1}$$

or

$$d = \sqrt{\frac{6\,E_s}{M-1}} \tag{5.2}$$

Now, we can start the derivation of ϵ. We will use a *pessimistic* approach, that is we will compute the worst symbol error probability. This can be done by calculating ϵ assuming that an "inner" signal in the square constellation is transmitted; i.e., a signal that has surrounding neighbor signals in all directions (like signal s_1 in figure (5.2)).

Given that z_1 and z_2 are the respective x and y components of the additive noise corrupting the sent signal s_1, such that z_1 and z_2 are iid AWGN random variables with zero mean and variance equal to $N_0/2$, we have:

$$\begin{aligned}
\epsilon &= P(error/s_1 \text{ sent}) \\
&= P(|z_1| > d/2) + P(|z_2| > d/2) - P(|z_1| > d/2)P(|z_2| > d/2) \\
&= 2P(|z_1| > d/2) - (P(|z_1| > d/2))^2
\end{aligned}$$

but $P(|z_1| > d/2) = P(z_1 > d/2) + P(z_1 < -d/2)$ and

$$P(z_1 > d/2) = P(z_1 < -d/2) = Q\left(d/\sqrt{2N_0}\right)$$

where Q(.) is defined by equation (4.4).

Using equation (5.2), we finally obtain:

$$\epsilon = 2q - q^2 \tag{5.3}$$

where

$$q = 2\,Q\left(\sqrt{\frac{3}{M-1}\,\frac{E_s}{N_0}}\right) \tag{5.4}$$

We now consider the derivation of γ. We know that:

$$\gamma = \frac{P(\text{uncommon error})}{\epsilon}$$

Using the same pessimistic approach we used in deriving ϵ, we write:

$$\begin{aligned} P(\text{uncommon error}) &= P(\text{uncommon error}/s_1\text{is sent}) \\ &= 2\,P\left[(|z_1| < d/2)\,\text{and}\,(|z_2| > 3d/2)\right] \\ &\quad + P\left[(|z_1| > d/2\text{and}|z_2| > d/2)\right] \\ &= (P(|z_1| > d/2))^2 + 2\,P(|z_1| < d/2)P(|z_2| > 3d/2) \end{aligned} \tag{5.5}$$

We define:

$$p_1 = Q\left(d/\sqrt{2N_0}\right) \tag{5.6}$$

$$p_2 = Q\left(3d/\sqrt{2N_0}\right) \tag{5.7}$$

We have:

- $P(|z_1| > d/2) = 2p_1$.
- $P(|z_1| > 3d/2) = 2p_2$.

Substituting equations (5.6), (5.7) and (5.2) in equation (5.5), we get:

$$P(\text{uncommon error}) = 4p_1^2 + 4p_2(1 - 2p_1) \tag{5.8}$$

Noting that $\epsilon = 4p_1 - 4p_1^2$, we finally obtain:

$$\gamma = \frac{4p_1^2 + 4p_2(1 - 2p_1)}{4p_1 - 4p_1^2} \tag{5.9}$$

$$\Rightarrow \gamma = \frac{p_1^2 + p_2(1 - 2p_1)}{p_1(1 - p_1)} \tag{5.10}$$

where,

$$p_1 = Q\left(\sqrt{\frac{3}{M-1}\,\frac{E_s}{N_0}}\right) \tag{5.11}$$

$$p_2 = Q\left(3\sqrt{\frac{3}{M-1}\,\frac{E_s}{N_0}}\right) \tag{5.12}$$

In order to study the performance of the focused codes used in association with square constellations, we need to determine a range of the symbol energy to noise ratio (E_s/N_0) under which we can "suitably" operate.

A suitable range for E_s/N_0 would be the one corresponding to a symbol error probability less than $\frac{1}{2}$ but exceeding at least 10^{-5}:

$$10^{-5} \leq \epsilon < 0.5$$

In general, we can approximate equation (5.3) in the following way:

$$\epsilon = 2q - q^2 \approx 2q \tag{5.13}$$

where,

$$q = 2\,Q\left(\sqrt{\frac{3}{M-1}\,\frac{E_s}{N_0}}\right)$$

Using the range for the values of ϵ along with equation (5.12), we get:

$$0.441(M-1) < \frac{E_s}{N_0} \leq 6.946(M-1) \tag{5.14}$$

5.2 Performance Of Focused Codes Using Square Constellations

5.2.1 Benchmark And P_d vs E_s/N_0 Plots

Applying equations (3.2), (3.9), (5.3) and (5.10) and selecting a suitable range for E_s/N_0 with the help of equation (5.13) while keeping a reasonable value of P_d (not exceeding 10^{-11}), we plot the curves of P_d versus E_s/N_0 and β versus E_s/N_0 for different values of M, n, t_1, and t_2.

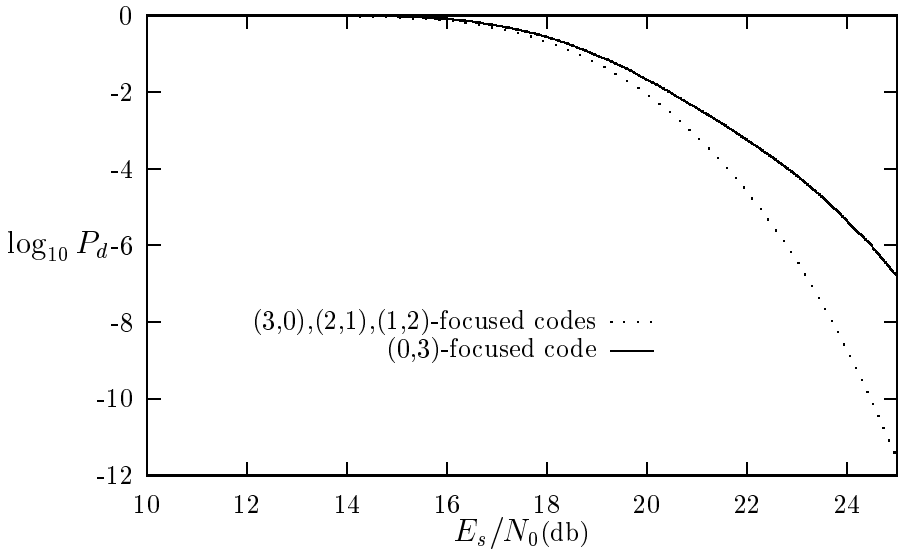

Figure 5.3: P_d vs E_s/N_0 for $M = 64$ QAM, $n = 15$, $t_1 + t_2 = 3$

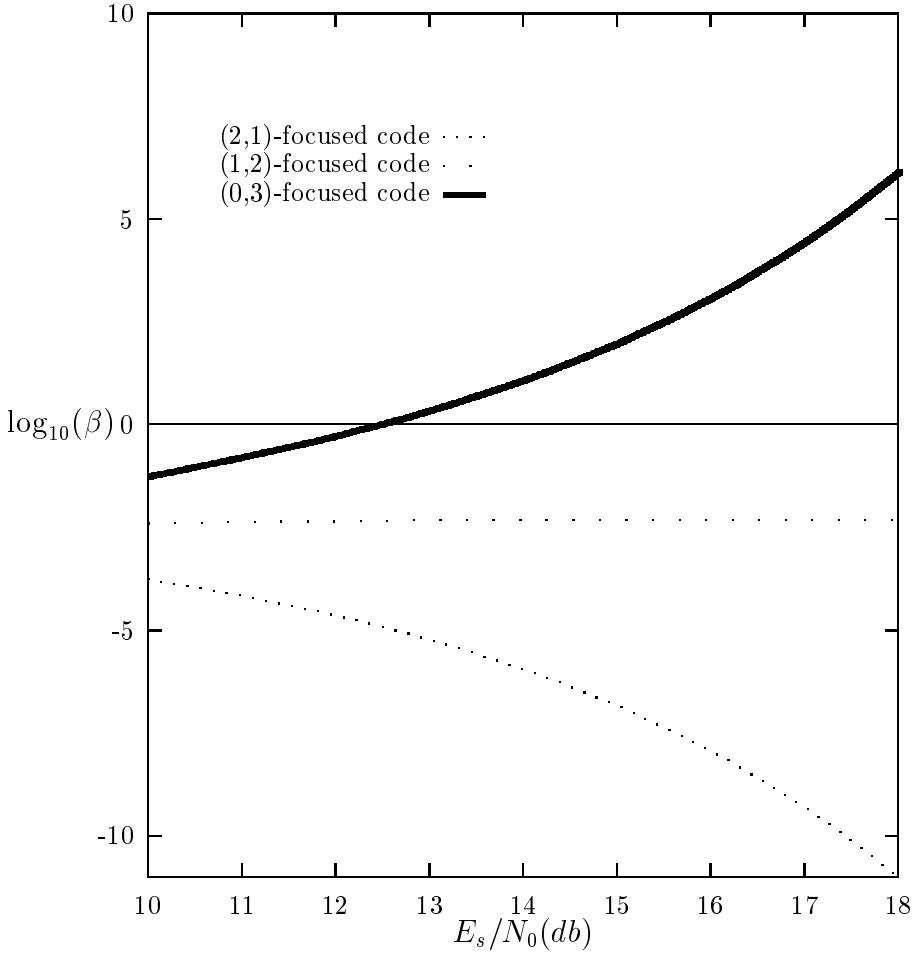

Figure 5.4: Benchmark plot for $M = 64$ QAM, $n = 15$, $t_1 + t_2 = 3$

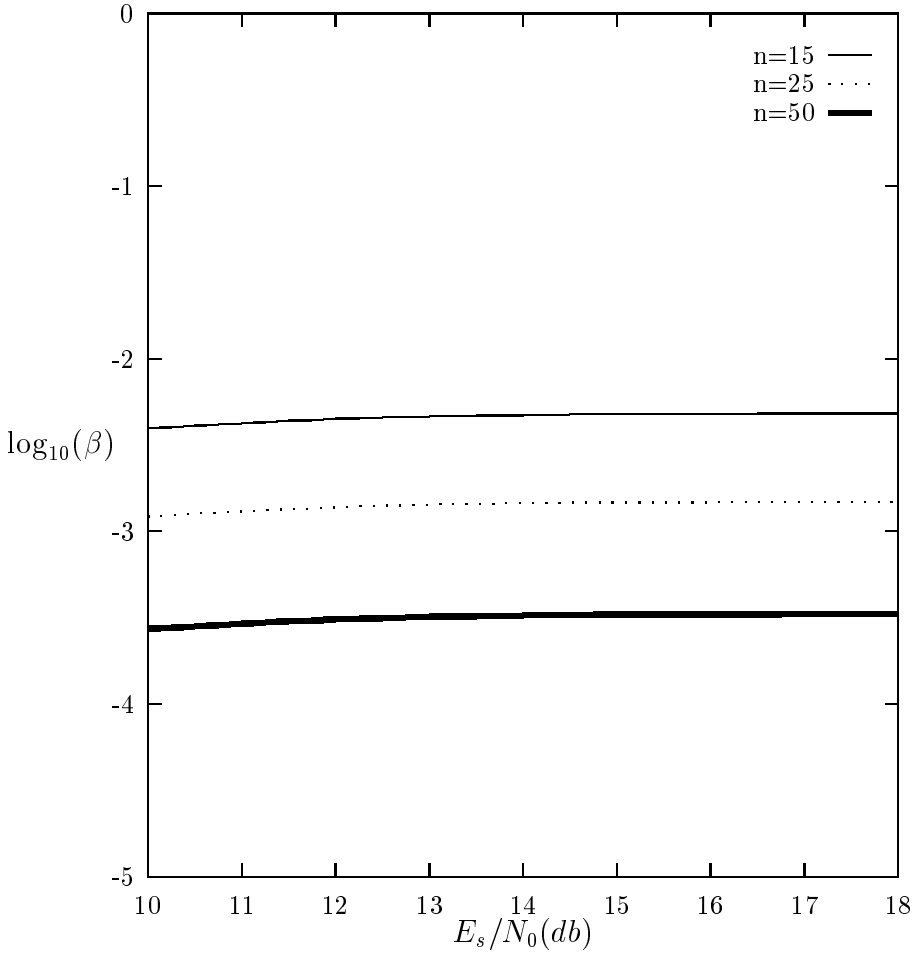

Figure 5.5: Benchmark plot for $M = 64$ QAM, (1,2)-focused code with various block lengths n

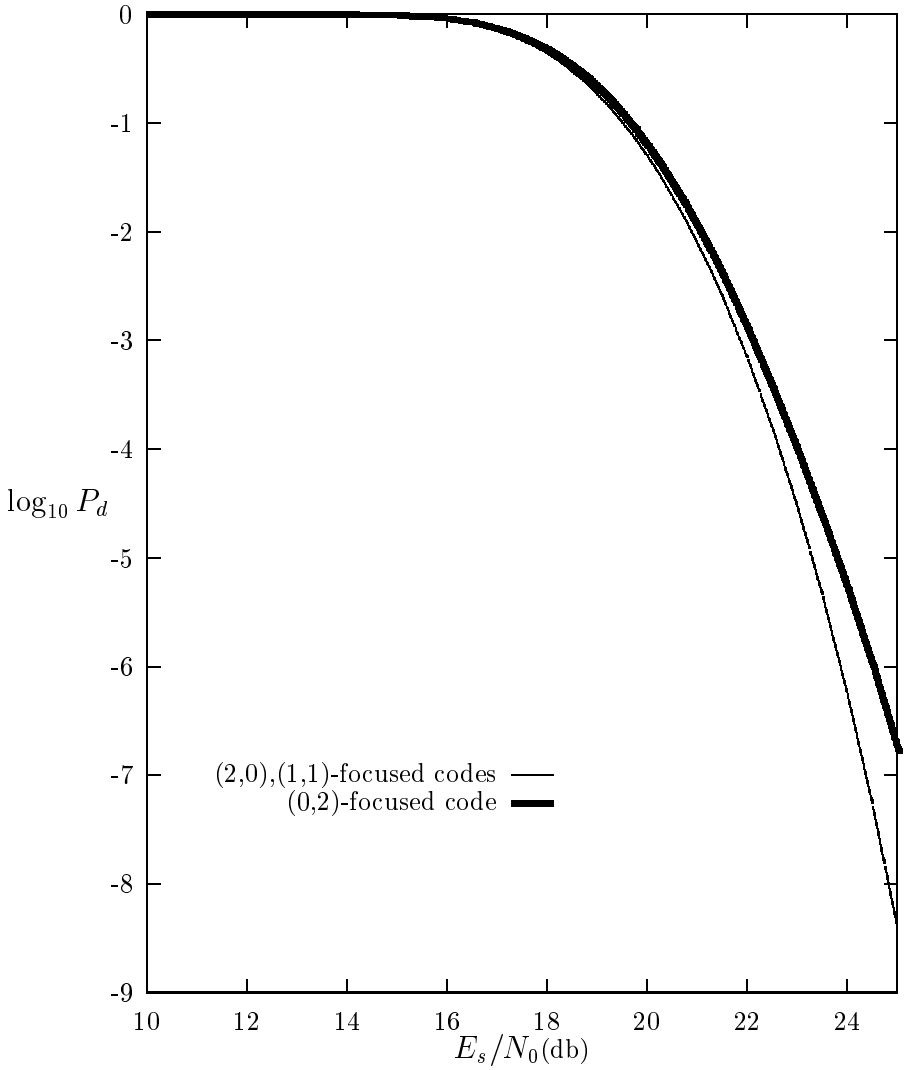

Figure 5.6: P_d vs E_s/N_0 for $M = 64$ QAM, $n = 15$, $t_1 + t_2 = 2$

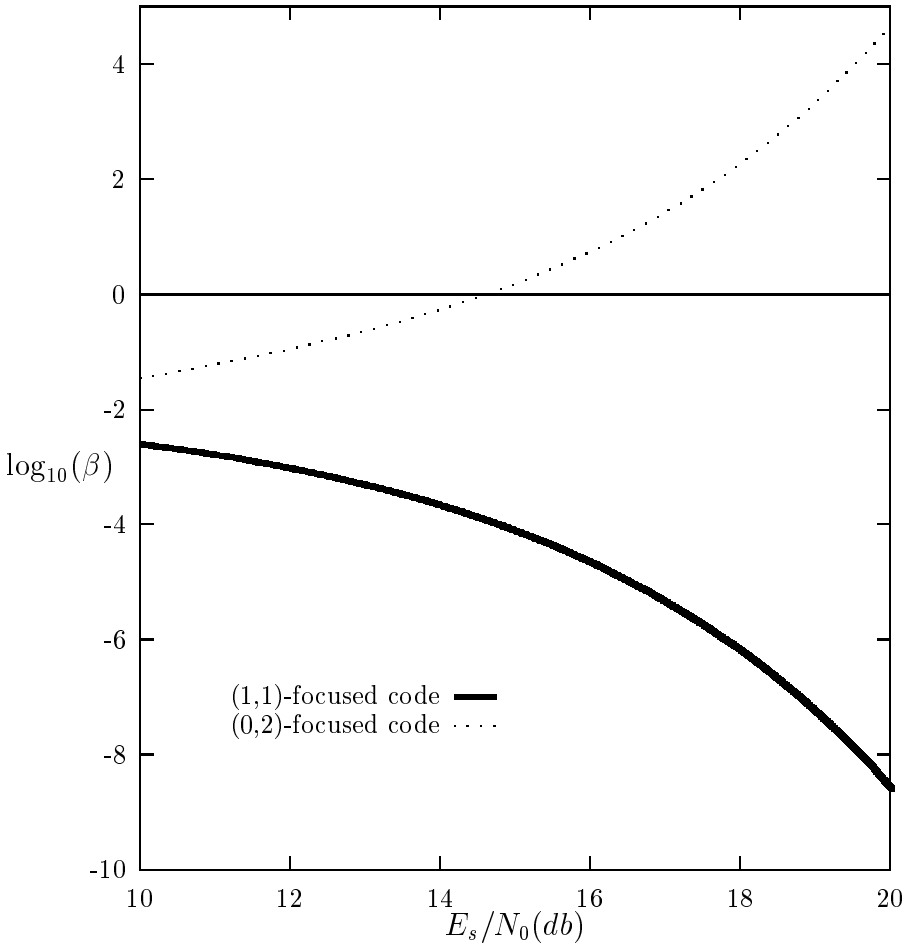

Figure 5.7: Benchmark plot for $M = 64$ QAM, $n = 15$, $t_1 + t_2 = 2$

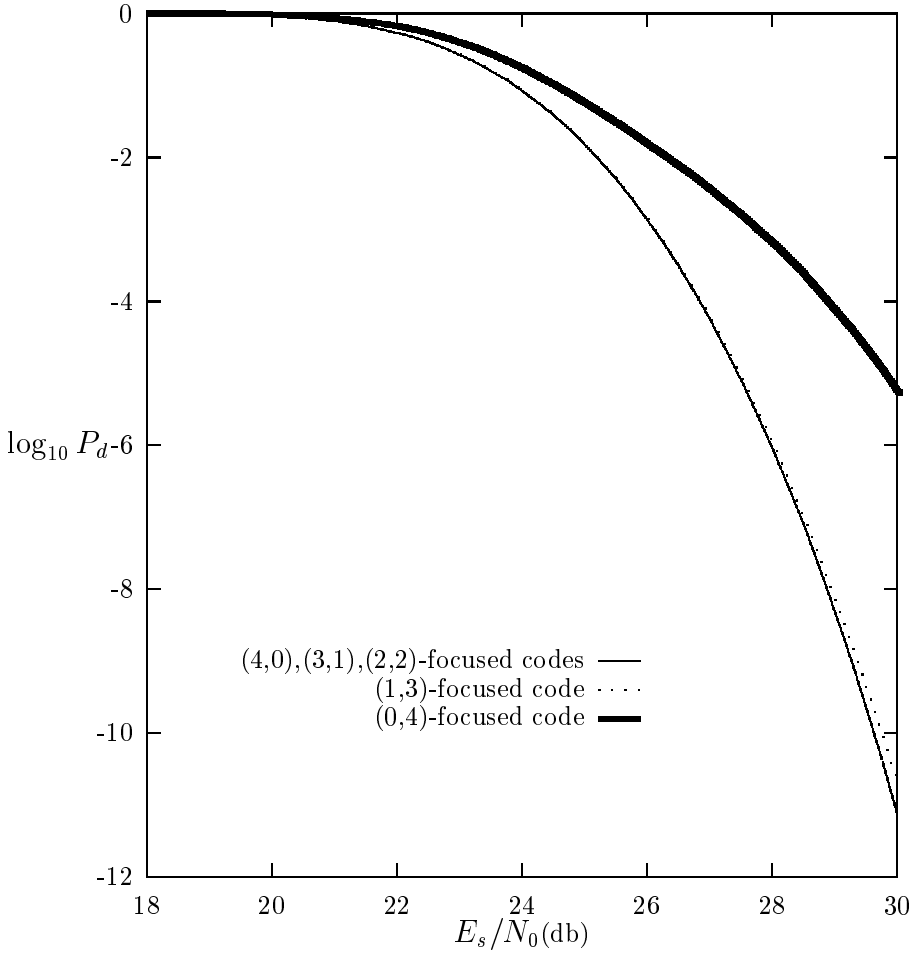

Figure 5.8: P_d vs E_s/N_0 for $M = 256$ QAM, $n = 15$, $t_1 + t_2 = 4$

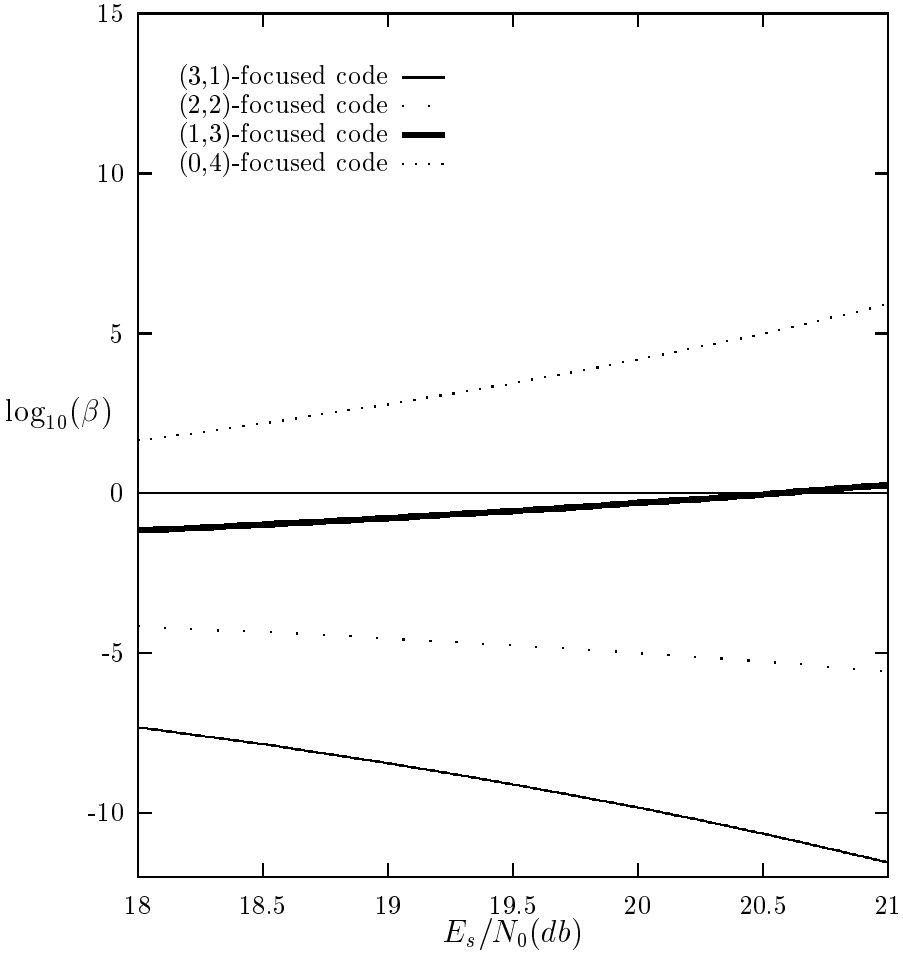

Figure 5.9: Benchmark plot for $M = 256$ QAM, $n = 15$, $t_1 + t_2 = 4$

5.2.2 Evaluation Of The Plots

In a similar way to the case when we used PSK modulation in the previous chapter, the above plots illustrate for us the condition given by equation (3.10). In fact as long as the benchmark of a (t_1, t_2)-focused code is less than 10^{-2}, its performance matches that of $t_1 + t_2$-error correcting code (which is a $(t_1 + t_2, 0)$-focused code) otherwise there is no performance matching (figures (5.3), (5.4), (5.6),(5.7),(5.8) and (5.9)). We remark also (figure (5.5)) that as the block length of the codeword increases, its benchmark decreases; yielding a stronger matching between the focused codes and the traditional codes and a better performance.

However unlike the case with PSK modulation where we could achieve a performance matching between *completely focused* codes ($(0, t)$-focused codes) and traditional t-error correcting codes, we remark that this can be achieved less frequently with square constellations. This may be explained by the fact that in square constellations each signals have 8 surrounding neighbors, out of which 4 of them are uncommon errors; while in PSK modulation, each signal has 2 direct common error neighbors only, and its next uncommon error signals are further away. This makes it highly unprobable to achieve a completely focused code (i.e., can correct only common errors) associated with square constellations that can match the performance of a traditional code.

Applying the same equations we used for the plots, we deduce the following results for an 8×8 constellation ($M = 64$) and all E_s/N_0:

- A 1-error correcting code performs identically to a (0,1)-focused code.
- A 2-error correcting code performs identically to a (1,1)-focused code.
- A 3-error correcting code performs identically to a (1,2)-focused code.
- A 4-error correcting code performs identically to a (2,2)-focused code.

5.3 Simulation Of The Focused Codes Performance Using Square Constellations

5.3.1 Simulation Description

As we did in chapter 4 for PSK modulation, we desire to check the accuracy of equations (5.3) and (5.10) that we derived analytically in section (5.1) by using a C simulation program to evaluate the performance (P_d vs E_s/N_0) of a (0,4)-focused code and of a (1,3)-focused code with a block length n equal to 15 associated with an 8×8 constellation ($M = 64$), in the presence of an additive white Gaussian noise (AWGN).

We present a brief description of the simulation algorithm:

1. For a specific value of E_s/N_0 and given that signal $s_1(d/2, d/2)$ is sent, determine the ranges of the x-y coordinates covering the correct detection region R_c, the common error detection region R_{ce} and the uncommon error detection region R_{ue} respectively. Set to zero the counter c_w of the number of codeword errors, the counter c_e of the number of signal errors and the counter c_{ue} of the number of uncommon signal errors.

2. Generate successively two AWGN random variables with zero mean and variance equal to $N_0/2$ (where N_0 is given) and add them respectively to the x-y coordinates of s_1 to obtain the coordinates of the received noisy signal $r_1(n_1 + d/2, n_2 + d/2)$.

   - If $r_1 \in R_{ue}$, increment c_e and c_{ue} by 1.
   - If $r_1 \in R_{ce}$, increment c_e by 1.
   - If $r_1 \in R_c$, proceed to the next step.

3. Repeat step 2 n times to generate a received vector of length n.

4. - If $c_e \geq t_1 + t_2 + 1$, increment c_w by 1 and go to step 5.

- If $c_e \leq t_1 + t_2 + 1$ and $c_{ue} \geq t_1 + 1$, increment c_w by 1 and go to step 5.

5. Repeat steps 1 to 4 a large number L of times (in the order of 10^9) and calculate $P_d = c_w/L$.

6. Repeat steps 1 to 5 for several values of E_s/N_0 and get the simulated plot.

5.3.2 Simulation Results

We use the above simulation program (complete C program is available from the author) for 14 different values of E_s/N_0 on a (0,4)-focused code and a (1,3)-focused code (of block length n) associated with an M-ary square constellation, where:

- $M = 64$, $n = 15$.
- $N_0 = (2)(10^{-6})$.
- Number of generated codewords: L of the order of 10^9.

We get the following simulated results:

E_s/N_0	P_d	L
10.00	-0.000009	100000
11.00	-0.000089	100000
12.00	-0.000786	100000
13.00	-0.004657	100000
14.00	-0.021013	100000
15.00	-0.069841	100000
16.00	-0.191482	100000
17.00	-0.430104	100000
18.00	-0.800101	100000
19.00	-1.288459	100000
20.00	-1.857924	100000
21.00	-2.498771	100000
22.00	-3.255963	1000000
23.00	-4.201031	1000000

Table 5.1: Simulation results of the performance of a (0,4)-focused code associated with a 64-ary square constellation and $n = 15$

E_s/N_0	P_d	L
10.00	-0.000068	100000
11.00	-0.000373	100000
12.00	-0.002501	100000
13.00	-0.012010	100000
14.00	-0.041113	100000
15.00	-0.117163	100000
16.00	-0.289631	100000
17.00	-0.603293	100000
18.00	-1.116957	100000
19.00	-1.880012	100000
20.00	-2.938876	100000
21.00	-4.346671	1000000
22.00	-6.168182	100000000
23.00	-8.458912	1000000000

Table 5.2: Simulation results of the performance of a (1,3)-focused code associated with a 64-ary square constellation and $n = 15$

From the results shown in tables (5.1) and (5.2) and figure (5.10), we can clearly remark that the simulation results match perfectly the performance plots that we obtained analytically.

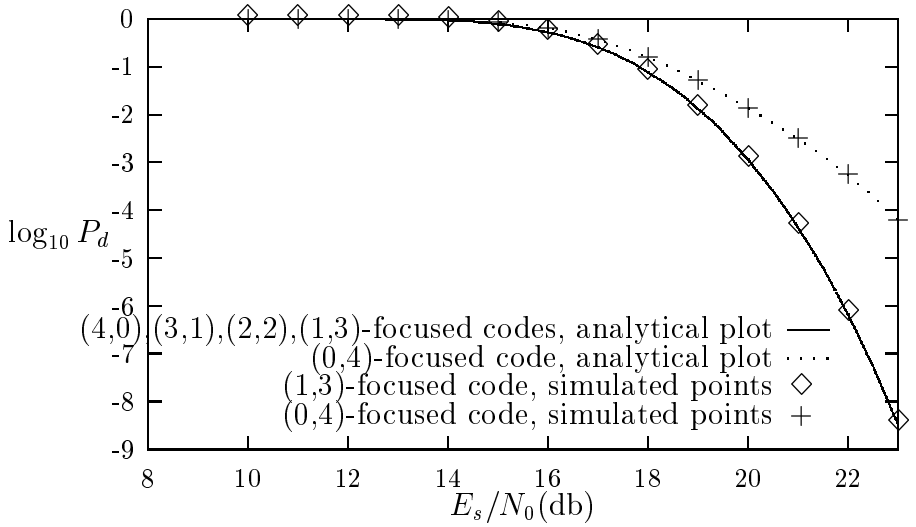

Figure 5.10: Simulation vs analytical performance plots for $M = 64$ QAM, $n = 15$, $(0, 4)$ and $(1, 3)$-focused codes

Chapter 6

Coding Gain Of Focused Codes

In this chapter, we evaluate the coding gains obtained using known techniques for constructing focused codes.

6.1 Construction Of Some Focused Codes

Referring to the construction technique described in chapter 2 and to code tables in [2] - [6], we construct (t_1, t_2)-focused codes over GF(2^b) and we focus them on the set $\mathbf{B} = \{$1-bit per-symbol errors$\}$. We compute also their code rate and the rate improvement they achieve over $t_1 + t_2$-error correcting codes for different values of the code block length n.

To construct a (t_1, t_2)-focused code C_f over GF(2^b) that is focused on the set $\mathbf{B} = \{$1-bit per-symbol errors$\}$we need:

- C_0, a (b,b-1) parity check code over GF(2) with rate: $R_0 = (b-1)/b$.
- C_1, an (n,k_1) code over GF(2) with minimum distance $d_1 = 2(t_1+t_2)+1$ and rate $R_1 = k_1/n$.

- C_2, an (n,k_2) code over GF(2^{b-1}) with minimum distance $d_2 = 2t_1 + t_2 + 1$ and rate $R_2 = k_2/n$.

Using equation (2.1), the overall rate of the focused code C_f becomes:

$$R_f = \frac{1}{b} R_1 + \frac{b-1}{b} R_2 \tag{6.1}$$

6.1.1 (t_1, t_2)-Focused Codes Over GF(16)

6.1.1.1 (0,1)-focused code vs 1-error correcting code

1. We get for C_f:
   - C_1: Hamming binary code with $d_1 = 3$.
   - C_2: parity check code over GF(8) with $d_2 = 2$.
2. 1-error correcting code: Reed Solomon code over GF(16) with $d = 3$ and rate $R = (n - d + 1)/n = (n - 2)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
7	4/7	6/7	0.7857	0.7142	10%

Table 6.1: (0,1)-focused code vs 1-error correcting code over GF(16)

We note that for larger block lengths n ($8 \leq n \leq 15$), (0,1)-focused code does not achieve any rate improvement over a 1-error correcting code.

6.1.1.2 (0,2)-focused code vs 2-error correcting code

1. We get for C_f:
   - C_1:binary code with $d_1 = 5$.
   - C_2:$d_2 = 3$, shortened generalized Hamming code over GF(8).

2. 2-error correcting code: $d = 5$, shortened Reed Solomon code over GF(16) with rate $R = (n - d + 1)/n = (n - 4)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
8	2/8	6/8	0.625	0.5	25%

Table 6.2: (0,2)-focused code vs 2-error correcting code over GF(16)

Here again, a (0,2)-focused code does not achieve any rate improvement over a 2-error correcting code for larger block lengths n.

6.1.2 (t_1, t_2)-Focused Codes Over GF(32)

6.1.2.1 (0,1)-focused code vs 1-error correcting code

1. We get for C_f:
   - C_1: binary code with $d_1 = 3$.
   - C_2: parity check code over GF(16) with $d_2 = 2$.
2. 1-error correcting code: Reed Solomon code over GF(32) with $d = 3$ and rate $R = (n - d + 1)/n = (n - 2)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
7	4/7	6/7	0.8	0.7142	12 %
9	5/9	8/9	0.8222	0.7778	5.7 %

Table 6.3: (0,1)-focused code vs 1-error correcting code over GF(32)

6.1.2.2 (0,2)-focused code vs 2-error correcting code

1. We get for C_f:
   - C_1: binary code with $d_1 = 5$.

- C_2: $d_2 = 3$, shortened generalized Hamming code over GF(16).

2. 2-error correcting code: $d = 5$, shortened Reed Solomon code over GF(32) with rate $R = (n - d + 1)/n = (n - 4)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
8	2/8	6/8	0.65	0.5	30 %
10	3/10	8/10	0.7	0.6	16.67 %
11	4/11	9/11	0.7272	0.6363	14.28 %
13	5/13	11/13	0.7538	0.6923	8.88 %
14	6/14	12/14	0.7714	0.7142	8 %
15	7/15	13/15	0.7867	0.7333	7.28 %
16	8/16	14/16	0.8	0.75	6.67 %
17	9/17	15/17	0.8118	0.7647	6.16 %

Table 6.4: (0,2)-focused code vs 2-error correcting code over GF(32)

6.1.2.3 (0,3)-focused code vs 3-error correcting code

1. We get for C_f:

   - C_1: binary code with $d_1 = 7$.
   - C_2: $d_2 = 4$, shortened Reed Solomon code over GF(16).

2. 3-error correcting code: $d = 7$, shortened Reed Solomon code over GF(32) with rate $R = (n - d + 1)/n = (n - 6)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
11	2/11	8/11	0.6181	0.4545	36 %
13	3/13	10/13	0.6615	0.5385	22.8 %
14	4/14	11/14	0.6857	0.5714	20 %
15	5/15	12/15	0.7067	0.6	17.78 %

Table 6.5: (0,3)-focused code vs 3-error correcting code over GF(32)

6.1.2.4 (0,4)-focused code vs 4-error correcting code

1. We get for C_f:
   - C_1: binary code with $d_1 = 9$.
   - C_2: $d_2 = 5$, shortened Reed Solomon code over GF(16).
2. 4-error correcting code: $d = 9$, shortened Reed Solomon code over GF(32) with rate $R = (n - d + 1)/n = (n - 8)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
14	2/14	10/14	0.5833	0.4286	36.1 %

Table 6.6: (0,4)-focused code vs 4-error correcting code over GF(32)

6.1.3 (t_1, t_2)-Focused Codes Over GF(64)

6.1.3.1 (0,1)-focused code vs 1-error correcting code

1. We get for C_f:
   - C_1: binary code with $d_1 = 3$.
   - C_2: $d_2 = 2$, parity check code over GF(32).
2. 1-error correcting code: $d = 3$, Reed Solomon code over GF(64) with rate $R = (n - d + 1)/n = (n - 2)/n$.

6.1.3.2 (1,1)-focused code vs 2-error correcting code

1. We get for C_f:
   - C_1: binary code with $d_1 = 5$.
   - C_2: $d_2 = 4$, shortened Reed Solomon code over GF(32).

n	R_1	R_2	R_f	R	*Rate improvement*
7	4/7	6/7	0.8095	0.7142	13.34 %
9	5/9	8/9	0.8333	0.7777	7.1 %
10	6/10	9/10	0.85	0.8	6.25 %
11	7/11	10/11	0.8636	0.8181	5.56 %
12	8/12	11/12	0.875	0.8333	5 %
13	9/13	12/13	0.8846	0.8461	4.55 %
14	10/14	13/14	0.8928	0.8571	4.16 %
15	11/15	14/15	0.9	0.8667	3.84 %

Table 6.7: (0,1)-focused code vs 1-error correcting code over GF(64)

2. 2-error correcting code: $d = 5$, Reed Solomon code over GF(64) with rate $R = (n - d + 1)/n = (n - 4)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
8	2/8	5/8	0.5625	0.5	12.5 %
10	3/10	7/10	0.6333	0.6	5 55 %
11	4/11	8/11	0.6667	0.6363	4.77 %

Table 6.8: (1,1)-focused code vs 2-error correcting code over GF(64)

6.1.3.3 (1,2)-focused code vs 3-error correcting code

1. We get for C_f:
   - C_1: binary code with $d_1 = 7$.
   - C_2: $d_2 = 5$, shortened Reed Solomon code over GF(32).
2. 3-error correcting code: $d = 7$, Reed Solomon code over GF(64) with rate $R = (n - d + 1)/n = (n - 6)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
11	2/11	7/11	0.5606	0.4545	23.34 %
13	3/13	9/13	0.6154	0.5385	14.28 %
14	4/14	10/14	0.6429	0.5714	12.51 %
15	5/15	11/15	0.6667	0.6	11.12 %
17	6/17	13/17	0.6961	0.6471	7.57 %
18	7/18	14/18	0.7129	0.6667	6.93 %
19	8/19	15/19	0.7281	0.6842	6.42 %
21	10 /21	17/21	0.7540	0.7143	5.56 %

Table 6.9: (1,2)-focused code vs 3-error correcting code over GF(64)

6.1.3.4 (2,2)-focused code vs 4-error correcting code

1. We get for C_f:
   - C_1: binary code with $d_1 = 9$.
   - C_2: $d_2 = 7$, shortened Reed Solomon code over GF(32).
2. 4-error correcting code: $d = 9$, Reed Solomon code over GF(64) with rate $R = (n - d + 1)/n = (n - 8)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
14	2/14	8/14	0.5	0.4286	16.66 %
17	3/17	11/17	0.5686	0.5294	4.54 %
20	5/20	14/20	0.625	0.6	4.17 %

Table 6.10: (2,2)-focused code vs 4-error correcting code over GF(64)

6.1.4 (t_1, t_2)-Focused Codes Over GF(256)

6.1.4.1 (0,1)-focused code vs 1-error correcting code

1. We get for C_f:

- C_1: binary code with $d_1 = 3$.
- C_2: $d_2 = 2$, parity check code over GF(128).

2. 1-error correcting code: $d = 3$, Reed Solomon code over GF(256) with rate $R = (n - d + 1)/n = (n - 2)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
5	2/5	4/5	0.75	0.6	25 %
6	3/6	5/6	0.7917	0.6667	18.75 %
7	4/7	6/7	0.8214	0.7143	15 %
9	5/9	8/9	0.8472	0.7778	8.9 %
11	7/11	10/11	0.875	0.8181	6.96 %
13	9/13	12/13	0.8942	0.8461	5.68 %
15	11/15	14/15	0.9083	0.8667	4.8 %
17	12/17	16/17	0.9118	0.8823	3.34 %

Table 6.11: (0,1)-focused code vs 1-error correcting code over GF(256)

6.1.4.2 (1,1)-focused code vs 2-error correcting code

1. We get for C_f:
   - C_1: binary code with $d_1 = 5$.
   - C_2: $d_2 = 4$, shortened Reed Solomon code over GF(128).
2. 2-error correcting code: $d = 5$, Reed Solomon code over GF(256) with rate $R = (n - d + 1)/n = (n - 4)/n$.

6.1.4.3 (1,2)-focused code vs 3-error correcting code

1. We get for C_f:
   - C_1: binary code with $d_1 = 7$.

n	R_1	R_2	R_f	R	*Rate improvement*
8	2/8	5/8	0.5781	0.5	15.62 %
10	3/10	7/10	0.65	0.6	8.33 %
13	5/13	10/13	0.7211	0.6923	4.16 %
15	7/15	12/15	0.7583	0.7333	3.41 %

Table 6.12: (1,1)-focused code vs 2-error correcting code over GF(256)

- C_2: $d_2 = 5$, shortened Reed Solomon code over GF(128).

2. 3-error correcting code: $d = 7$, Reed Solomon code over GF(256) with rate $R = (n - d + 1)/n = (n - 6)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
11	2/11	7/11	0.5795	0.4545	27.5 %
13	3/13	9/13	0.6346	0.5385	17.85 %
15	5/15	11/15	0.6833	0.6	13.88 %
17	6/17	13/17	0.7132	0.6471	10.21 %
19	8/19	15/19	0.7434	0.6842	8.65 %
21	10/21	17/21	0.7678	0.7143	7.49 %
26	13/26	22/26	0.8029	0.7692	4.38 %
30	16/30	26/30	0.825	0.8	3.125 %

Table 6.13: (1,2)-focused code vs 3-error correcting code over GF(256)

6.1.4.4 (2,2)-focused code vs 4-error correcting code

1. We get for C_f:
   - C_1: binary code with $d_1 = 9$.
   - C_2: $d_2 = 7$, shortened Reed Solomon code over GF(128).
2. 4-error correcting code: $d = 9$, Reed Solomon code over GF(256) with rate $R = (n - d + 1)/n = (n - 8)/n$.

n	R_1	R_2	R_f	R	*Rate improvement*
14	2/14	8/14	0.5178	0.4286	20.82 %
17	3/17	11/17	0.5882	0.5294	11.11 %
19	4/19	13/19	0.625	0.5789	7.96 %
20	5/20	14/20	0.6438	0.6	7.29 %
23	7/23	17/23	0.6848	0.6522	5 %
25	8/25	19/25	0.705	0.68	3.68 %
30	12/30	24/30	0.75	0.7333	2.28 %

Table 6.14: (2,2)-focused code vs 4-error correcting code over GF(256)

6.1.5 Observation

From the above results, we can remark a net gain in the rate that focused codes achieve over traditional error correcting codes while having an identical performance. For example, a (0,2)-focused code that performs identically to a 2-error correcting code when used with a 16-ary PSK modulation (section 4.2.2), achieves a rate improvement of 25 % for n=8 (table (6.2)). Note that a (1,1)-focused code over GF(64) has a rate improvement of 4.77 % for n=11, while for the (1,2)-focused code with the same blocklength, the improvement is more impressive (14.28 %) (table (6.9)). This fact reinforces the common sense idea that the more "focused" a code is -that is, the more error patterns we avoid correcting - the bigger the payoff in terms of rate. We furthermore observe that the rate improvement vanishes as the blocklength increases. This is due to the particular construction technique we are using.

6.2 Coding Gain Using PSK and QAM Modulations

We are interested in computing the coding gain in information bit energy to noise ratio E_b/N_0 for (t_1, t_2)-focused codes. We will use the decoded *symbol* error probability P_s instead of the decoder error probability P_d that we used in the previous chapters.

Using the expression of P_d given by equation (3.2), we can directly deduce the expression of P_s:

$$\begin{aligned} P_s &= \sum_{i=t_1+1}^{t_1+t_2} \sum_{j=t_1+1}^{i} \min\left[n, \frac{t_1+t_2+i}{n}\right] \left(\begin{array}{c} n \\ i \end{array}\right) \left(\begin{array}{c} i \\ j \end{array}\right) \epsilon^i (1-\epsilon)^{n-i} \gamma^j (1-\gamma)^{i-j} \\ &\quad + \sum_{i=t_1+t_2+1}^{n} \min\left[n, \frac{t_1+t_2+i}{n}\right] \left(\begin{array}{c} n \\ i \end{array}\right) \epsilon^i (1-\epsilon)^{n-i} \end{aligned} \tag{6.2}$$

We note that if ϵ and γ are "reasonably" small (i.e., for high energy to noise ratio in the case we are using square constellations or PSK modulation), we can write:

$$P_s \approx \left(\frac{d_2}{n}\right) P_d + \frac{t_2}{n} P_t \tag{6.3}$$

where:

- $d_2 = 2t_1 + t_2 + 1$, is the minimum distance of the outer code C_2 used in the construction of the (t_1, t_2)-focused codes.
- $P_t \approx \left(\begin{array}{c} n \\ t_1+t_2+1 \end{array}\right) \epsilon^{t_1+t_2+1} (1-\epsilon)^{n-t_1-t_2-1}$, is the decoder error probability of a traditional $t_1 + t_2$-error correcting code.

An effective computation of the coding gain achieved by a (t_1, t_2)-focused code used in conjunction with M-ary PSK modulation or M-ary square constellation, over a $t_1 + t_2$-error correcting code, would be to compute it in terms of it information bit energy to noise ratio E_b/N_0.

If we let:

- E_b be the energy per information bit,
- E_s be the energy per symbol,
- R_f be the rate of a (t_1, t_2)-focused code.

We have:

$$E_b = \frac{E_s}{(\log_2 M)\, R_f} \tag{6.4}$$

$$\Rightarrow E_s = (\log_2 M)\, R_f\, E_b \tag{6.5}$$

Using equations (6.2) and (6.5) along with the rate results computed in section (6.1) and the equations (4.9), (4.10), (5.3) and (5.10) previously derived, we plot the curves of P_s vs E_b/N_0 for (t_1, t_2)-focused codes, $t_1 + t_2$)-error correcting Reed Solomon (RS) codes and uncoded data strings associated with M-ary PSK modulation and square constellations. (Complete C programs for all the different plots are available from the author.)

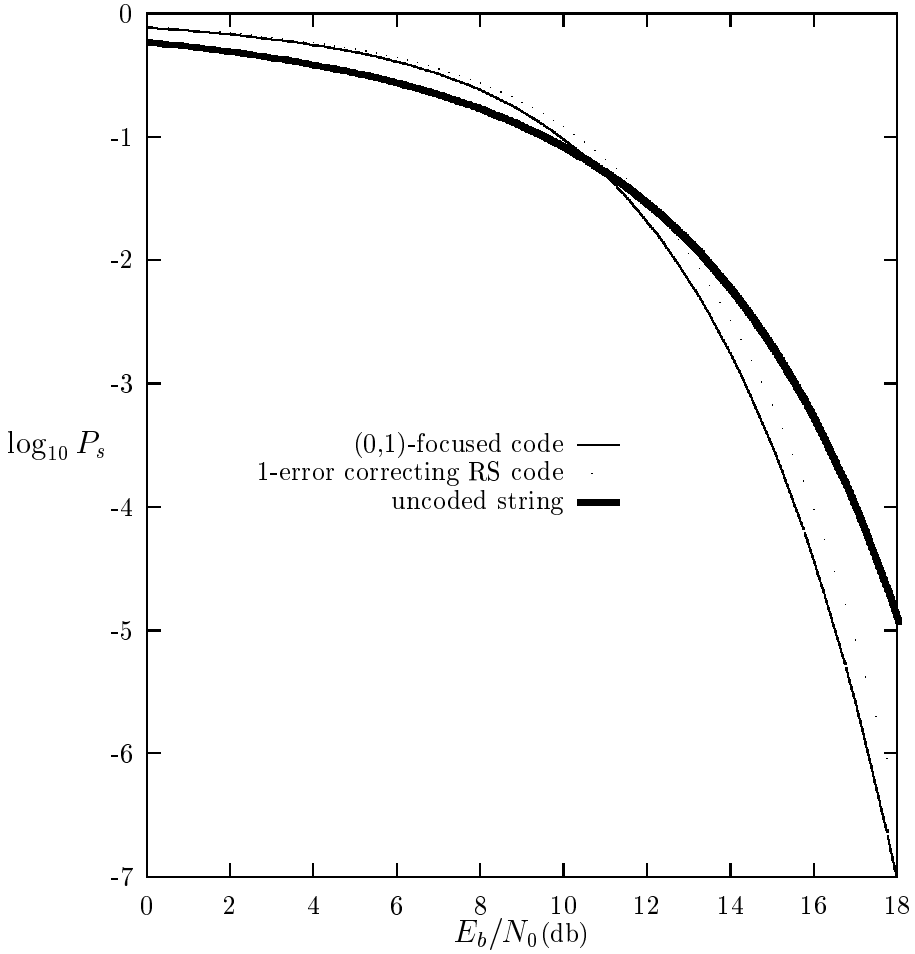

Figure 6.1: 16-ary PSK modulation with n=7; $t_1 + t_2 = 1$

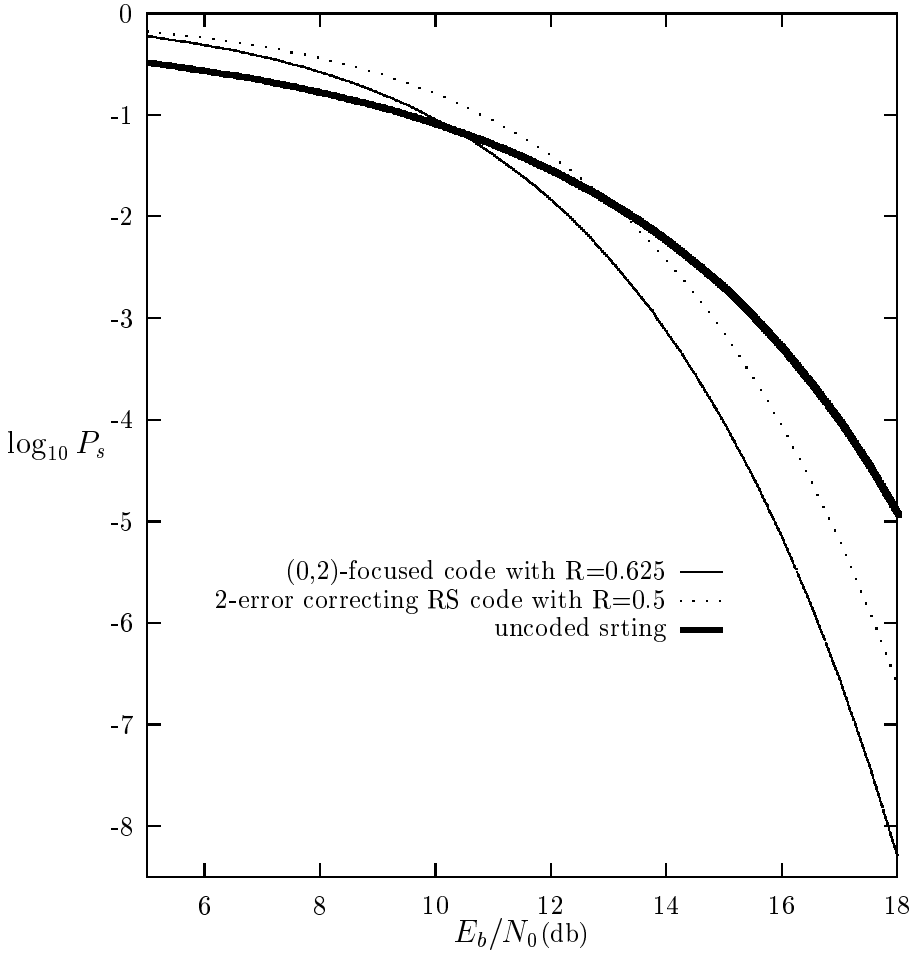

Figure 6.2: 16-ary PSK modulation with n=8; $t_1 + t_2 = 2$

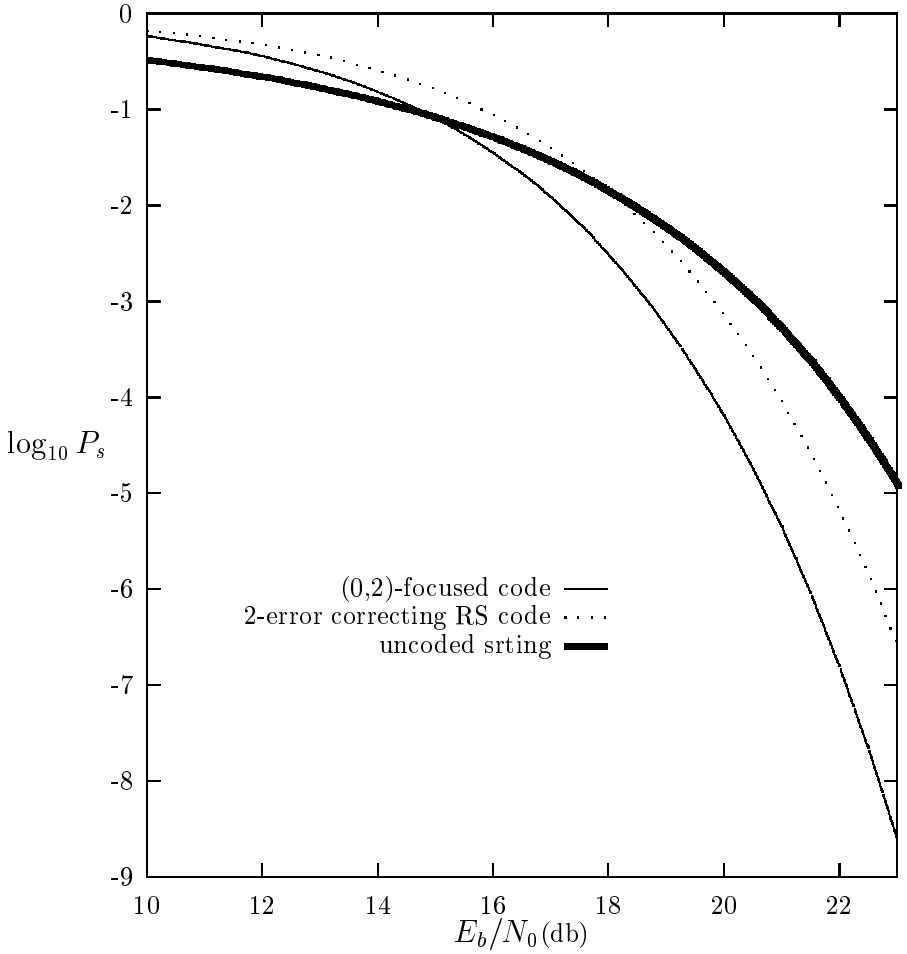

Figure 6.3: 32-ary PSK modulation with n=8; $t_1 + t_2 = 2$

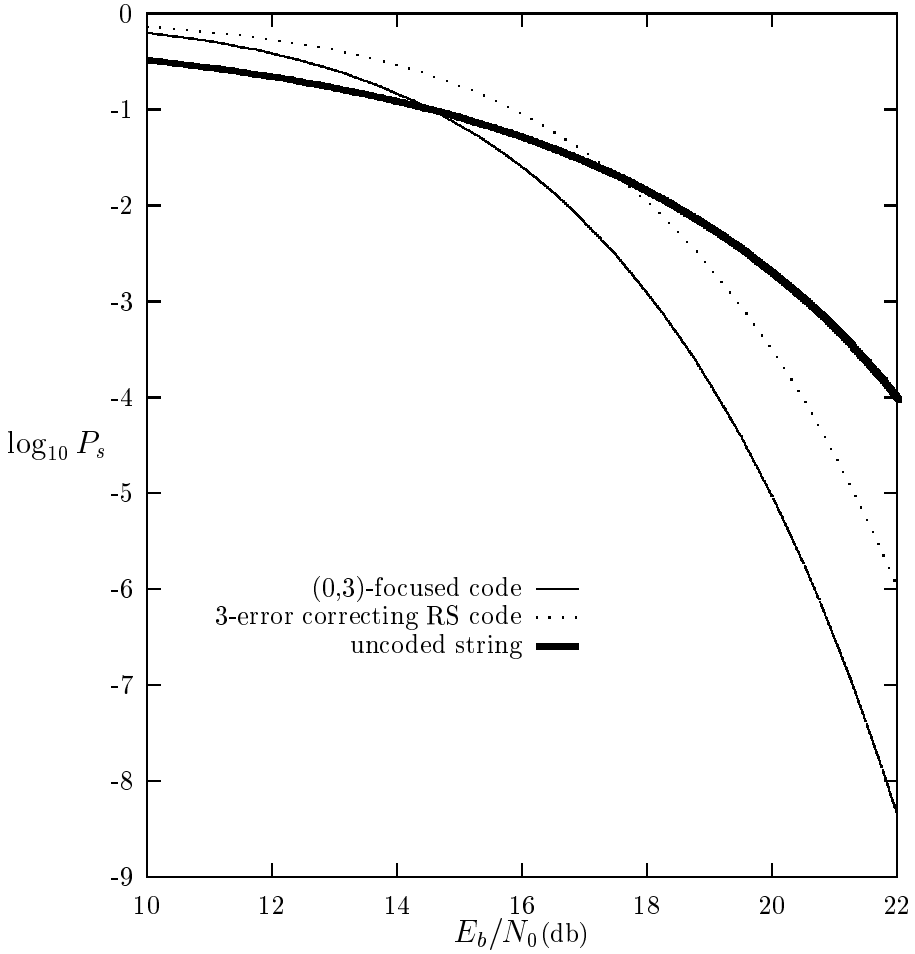

Figure 6.4: 32-ary PSK modulation with n=11; $t_1 + t_2 = 3$

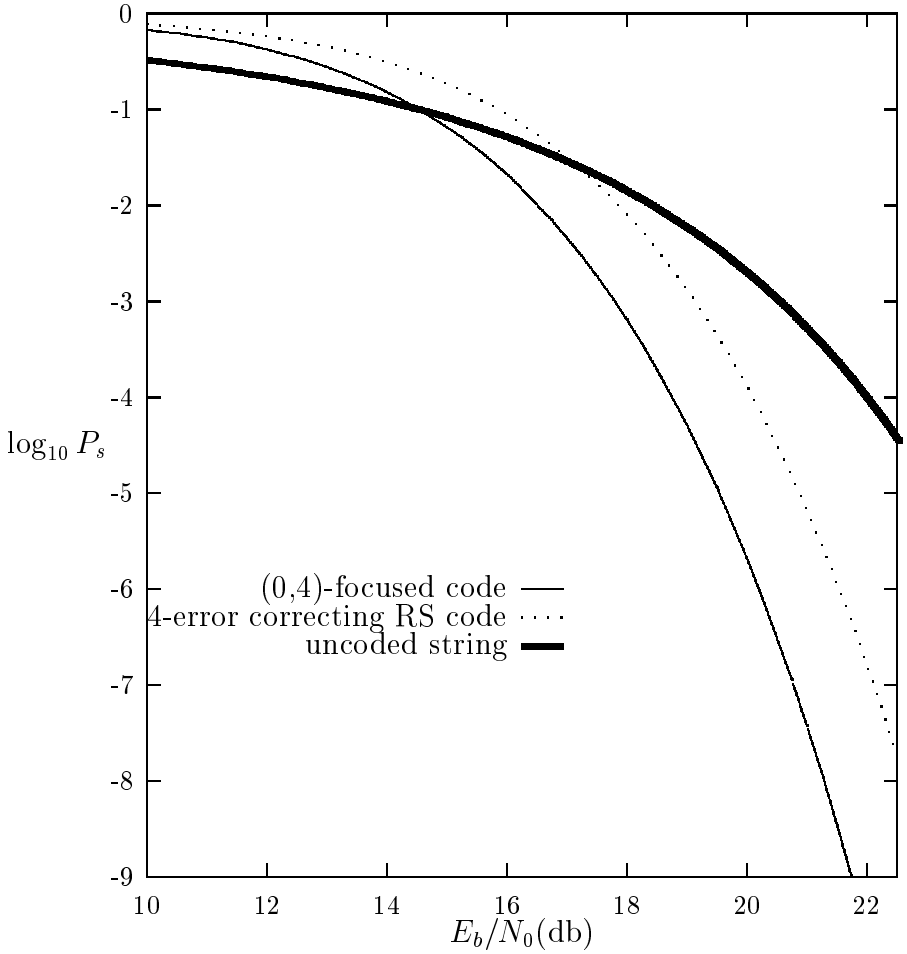

Figure 6.5: 32-ary PSK modulation with n=14; $t_1 + t_2 = 4$

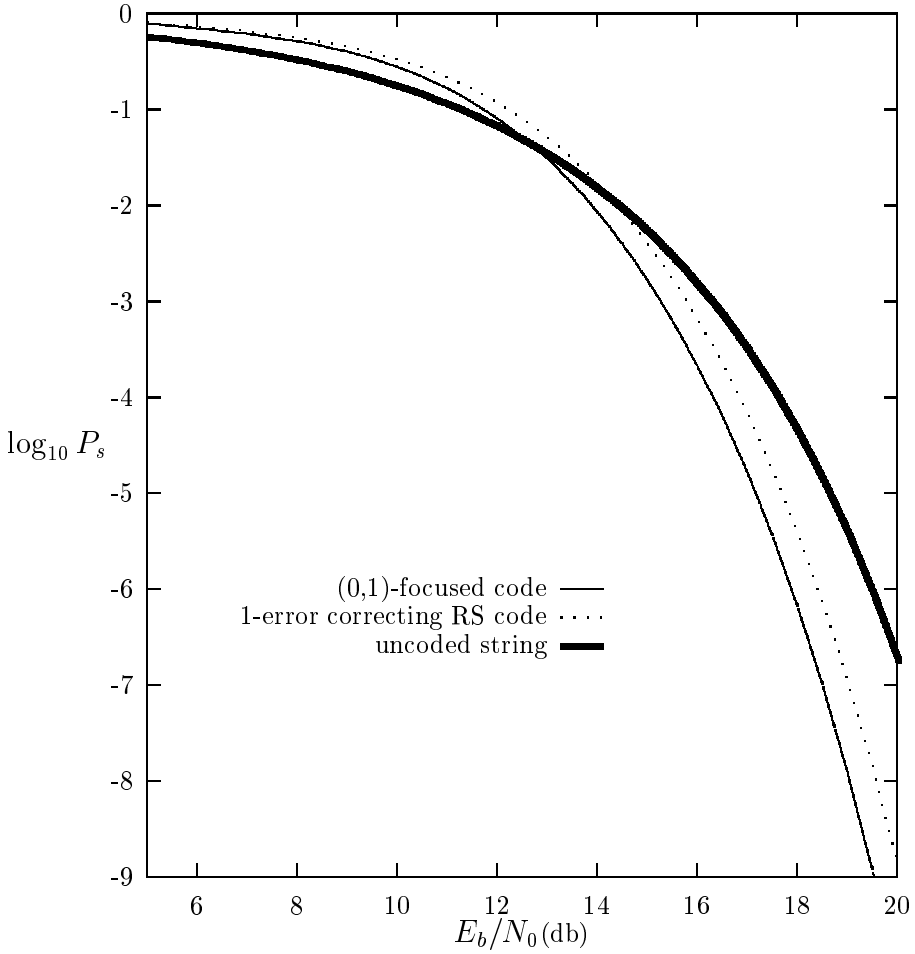

Figure 6.6: 8×8 square constellation with n=7; $t_1 + t_2 = 1$

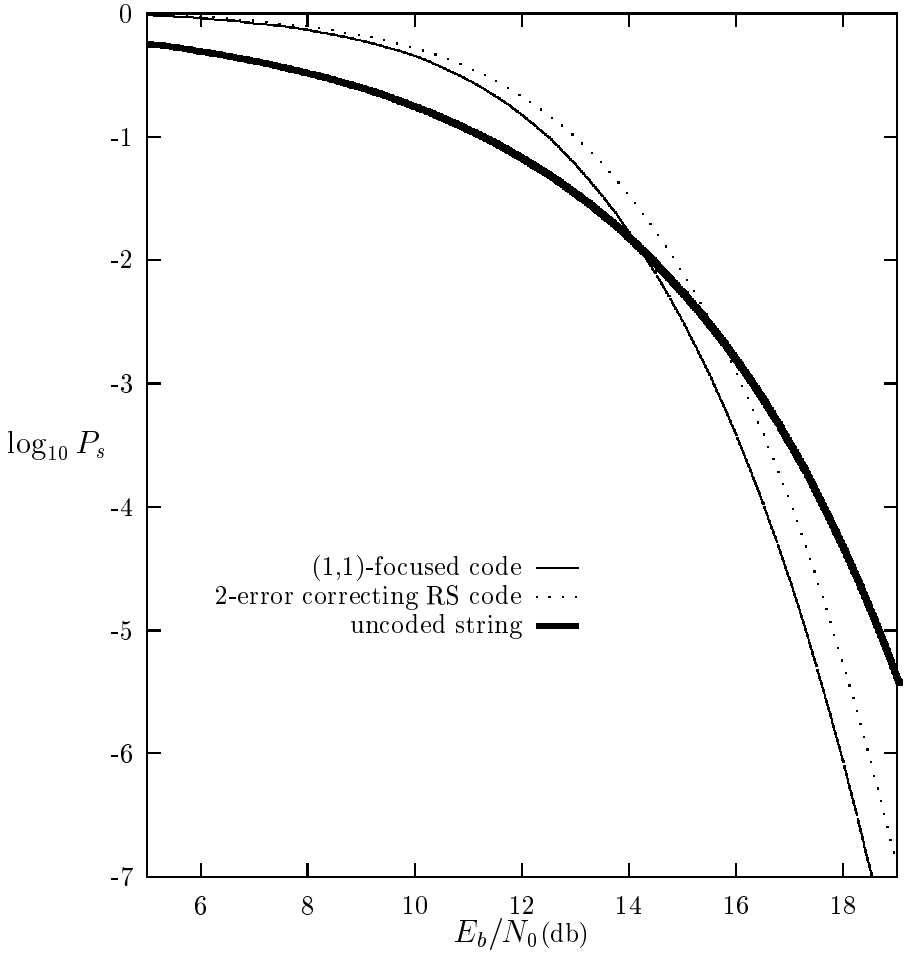

Figure 6.7: 8×8 square constellation with n=8; $t_1 + t_2 = 2$

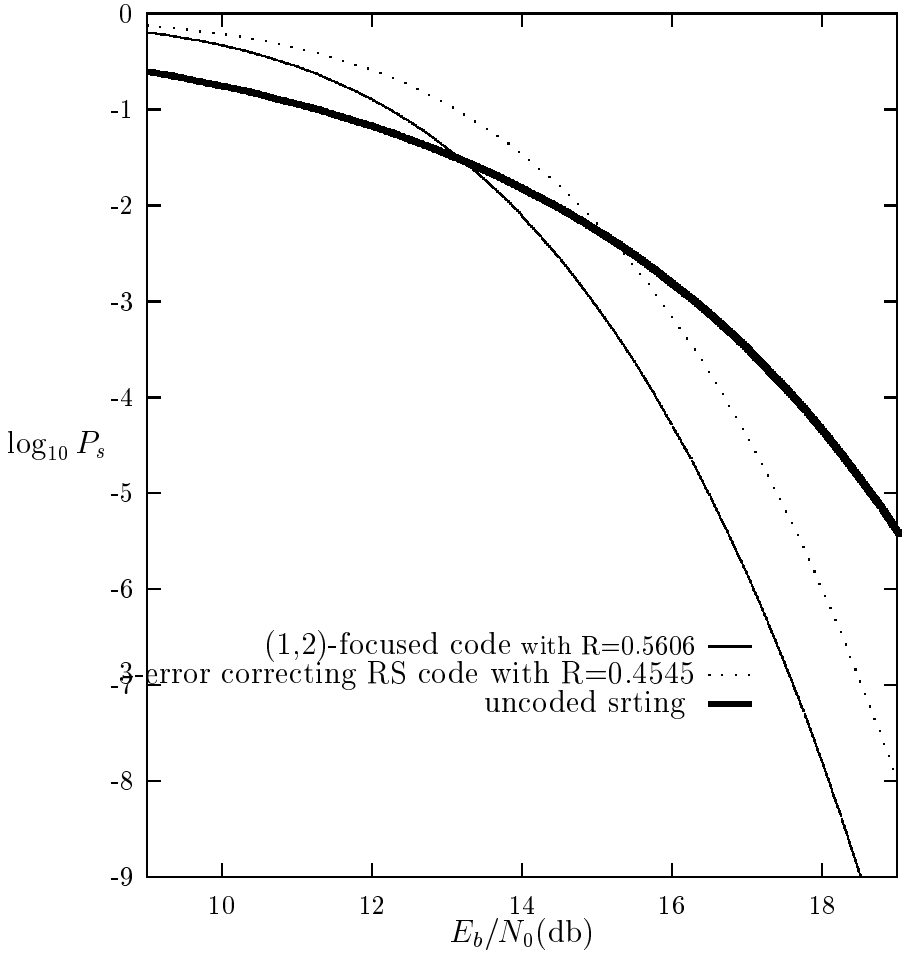

Figure 6.8: 8×8 square constellation with n=11; $t_1 + t_2 = 3$

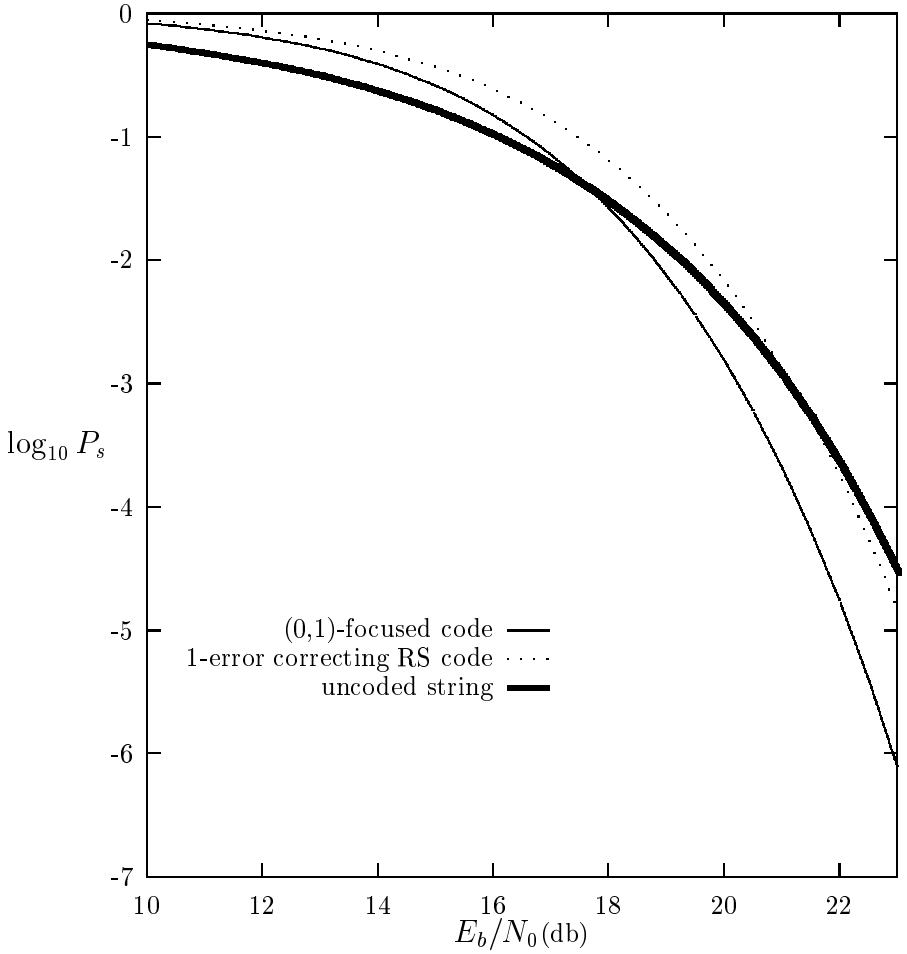

Figure 6.9: 16×16 square constellation with n=5; $t_1 + t_2 = 1$

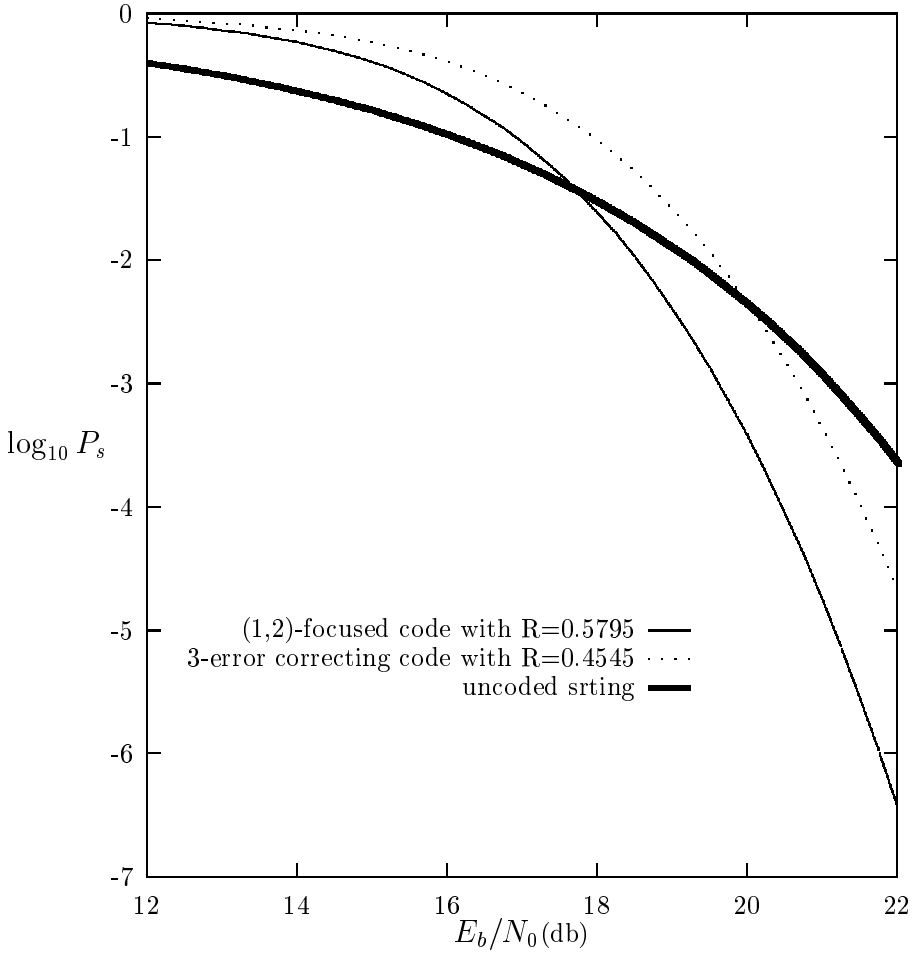

Figure 6.10: 16×16 square constellation with n=11; $t_1 + t_2 = 3$

We can clearly see from the above plots that both (t_1, t_2)-focused codes and $t_1 + t_2$-error correcting Reed Solomon codes achieve substantial coding gains in E_b/N_0 over uncoded messages. Moreover, the focused codes achieve also fair amounts of coding gain over the Reed Solomon codes. For example, under a 32-ary PSK modulation with n=8, a (0,2)-focused code achieves a coding gain of 1.13 db over a 2-error correcting RS code at $P_s = 10^{-6}$ (figure (6.3)). And a (1,2)-focused code used with a 256-ary square constellation and n=11 achieves a coding gain of 1.06 db at $P_s = 10^{-6}$ over a 3-error correcting RS code. Note that the coding gain of a focused code (that is one of the above focused codes) over a RS code is *constant*. This can be explained by the fact that all the (t_1, t_2)-focused codes used above, perform identically as the $t_1 + t_2$-error correcting Reed Solomon codes and the coding gain depends directly on the rate improvement (which is a constant).

6.2.1 Coding gain vs Blocklength

Finally, we plot the coding gain achieved at $P_s = 10^{-6}$ by the focused codes for different values of the blocklength n.

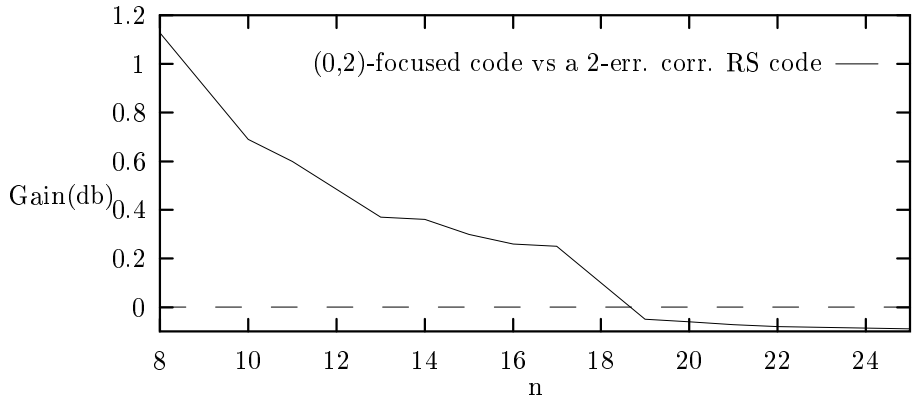

Figure 6.11: Coding gain using a 32-ary PSK modulation

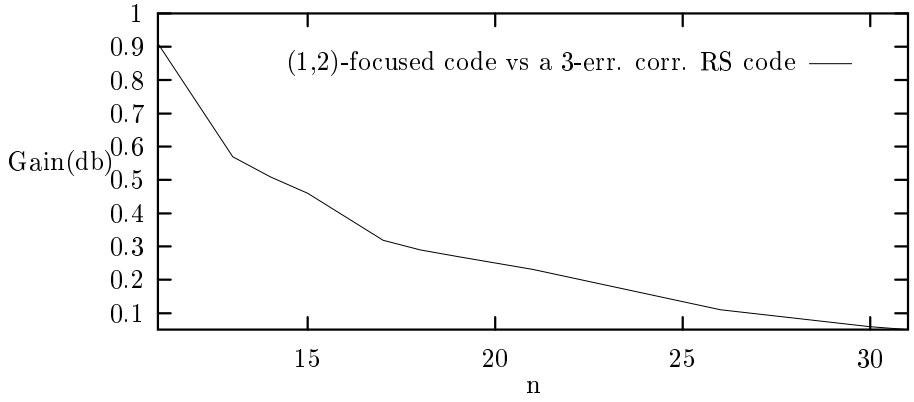

Figure 6.12: Coding gain using an 8×8 square constellation

We can remark from the above plots that the coding gain in E_b/N_0 focused codes achieve over the Reed Solomon codes decreases as the blocklength n increases until it becomes negative for very large values of n. This is due to the particular construction technique we used.

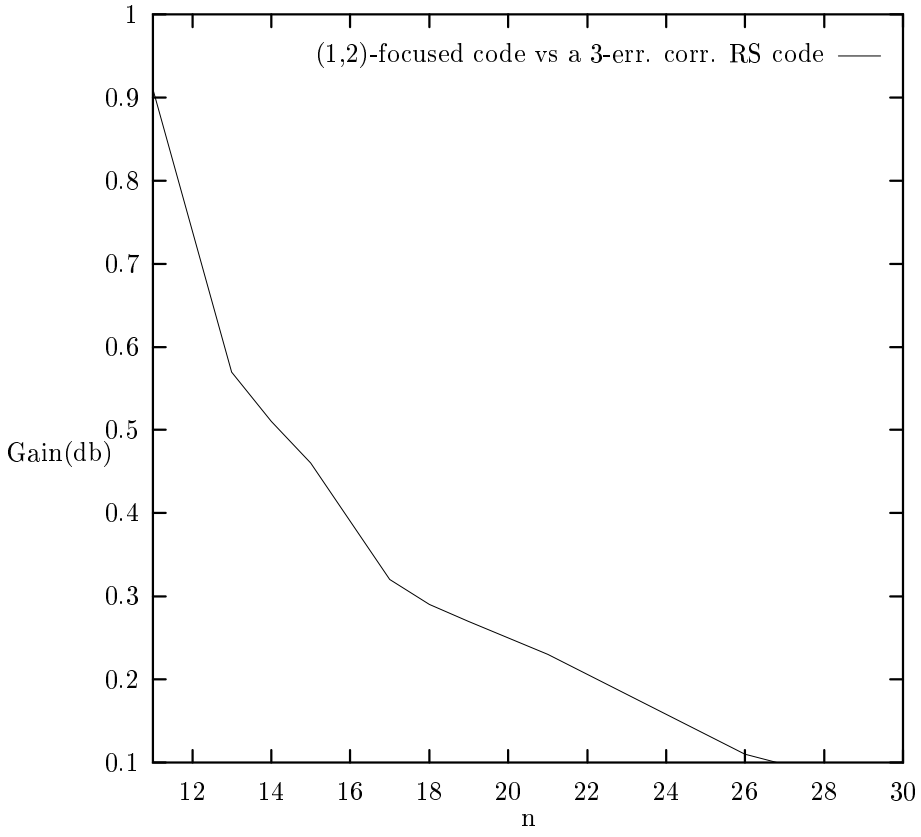

Figure 6.13: Coding gain using a 16×16 square constellation

Chapter 7

Adaptive Decoding Of Focused Codes

7.1 Adaptive Decoding Of Focused Codes Over Idealized Channels

We start by introducing the concept of "adaptive decoding" for focused codes over "idealized" skewed channels (i.e., channels with "independent" parameters ϵ and γ).

7.1.1 Observation

Up till now, we have constructed specific (t_1, t_2)-focused codes over $GF(2^b)$ using a code C_0 capable of detecting all the errors lying in the focus set $\mathbf{B}$, an inner code C_1 with minimum distance $d_1 = 2t_1 + 2t_2 + 1$ and an outer code C_2 with minimum distance $d_2 = 2t_1 + t_2 + 1$, where t_1 and t_2 are given.

We recall briefly (refer to section (2.2) and [1]) the decoding algorithm

used to decode the focused codes. Without any loss of generality, we assume the set of common errors $\mathbf{B}$ as being the set of odd-weight errors. For a given transmitted codeword from a (t_1, t_2)-focused code over $GF(2^b)$, we compute the mod-two sum of the bits of each component of the received codeword $\mathbf{r}$ and compare the result $\mathbf{b(r)}$ to the closest codeword $\mathbf{x}$ in C_1. We mark the locations where $\mathbf{b(r)}$ differs from $\mathbf{x}$ as erasures, take off the last bit in each symbol of $\mathbf{b(r)}$ and pass the resulting block over $GF(2^{b-1})$ to C_2. This decoding scheme will correct all the combinations of $t_1 + t_2$ errors provided *at most* t_1 are uncommon errors (i.e., have an even-weight binary representation) - and thus must be corrected by C_2 without the benefit of erasure. We therefore remark that by using this decoding algorithm, we cannot decode correctly any of the codewords having *less* than $t_1 + t_2$ errors out of which *more than* t_1 are uncommon errors.

7.1.2 Focused Codes Associated with d_1 and d_2

Suppose now we are given the minimum distances d_1 and d_2 of C_1 and C_2 respectively, and using C_0, C_1 and C_2, we construct a focused code "associated" with d_1 and d_2.

Given d_1 and d_2, we can have different combinations of t_1 and t_2 yielding different (t_1, t_2)-focused codes as long as t_1 and t_2 satisfy:

$$d_1 \geq 2t_1 + 2t_2 + 1 \tag{7.1}$$
$$d_2 \geq 2t_1 + t_2 + 1 \tag{7.2}$$

Obviously, we are only interested in the highest possible values of t_1 and t_2 that satisfy the above equations since we want to generate (t_1, t_2)-focused codes that correct the maximum number of errors (we are thus interested in the values of t_1 and t_2 that achieve at least one equality among the above two inequalities).

For example if $d_1 = 15$ and $d_2 = 11$, we get the following focused codes: a (5,0)-focused code, a (4,2)-focused code, a (3,4)-focused code, a (2,5)-focused code, a (1,6)-focused code and a (0,7)-focused code. Assume we transmit a

codeword (of blocklength n equal to 50) from a focused code associated with $d_1 = 15$ and $d_2 = 11$. At the receiver, the decoder can treat the received codeword as a codeword of any of the above stated (t_1, t_2)-focused codes. So in order to study the decoding scheme we intend to apply, we first look at the individual performance of each of the above (t_1, t_2)-focused codes.

Assuming we are working over an idealized symmetric channel skewed on the set $\mathbf{B}$ of odd-weight errors with the probability of symbol channel error, ϵ, fixed to the value of 0.01, we obtain the following curves for P_d versus γ:

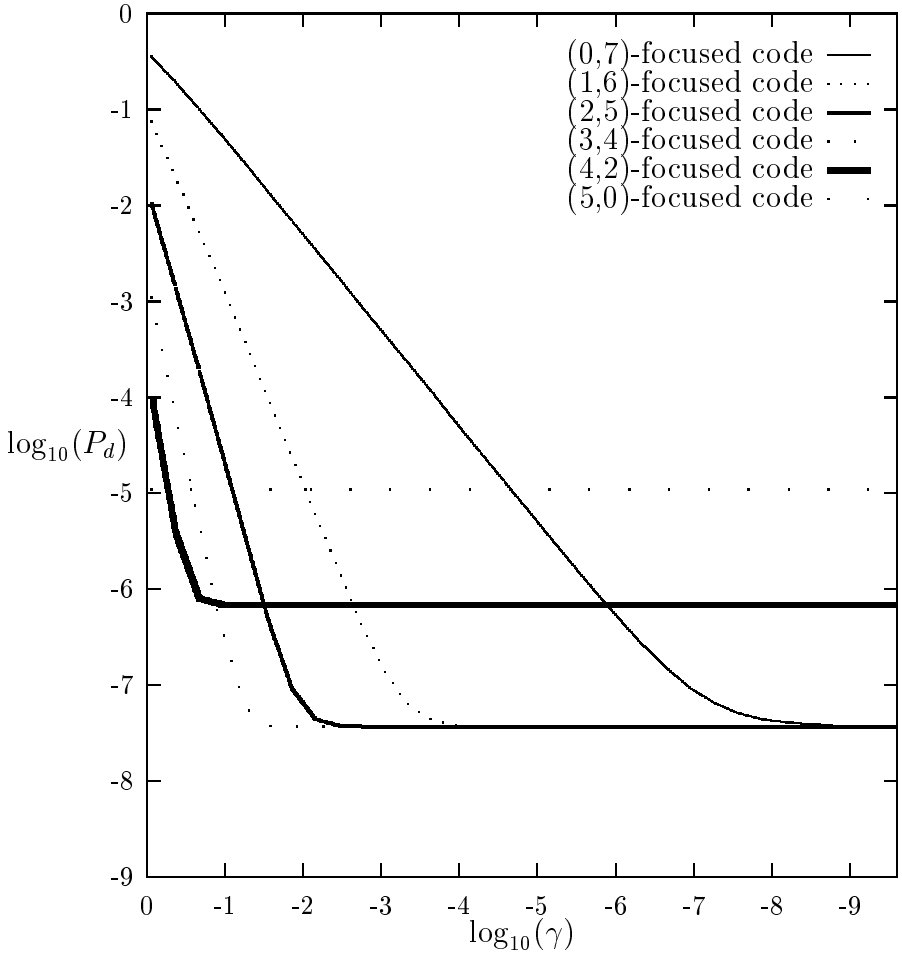

Figure 7.1: P_d versus γ for n=50 and $\epsilon = 0.01$

We can remark that we should be able to do at least as well as the minimum of all the performance curves in figure (7.1), which is the following:

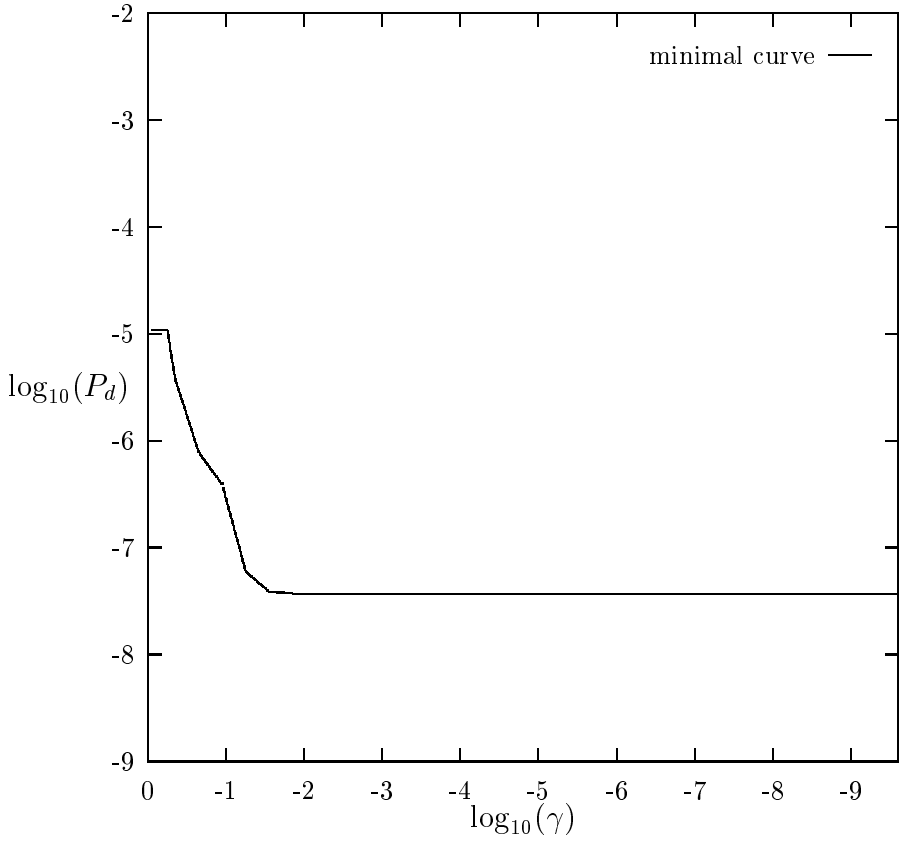

Figure 7.2: Minimal curve performance in decoding a focused code associated with $d_1 = 15$ and $d_2 = 11$ for n=50 and $\epsilon = 0.01$

7.1.3 Adaptive Decoding of the Focused Code Associated with $d_1 = 15$ and $d_2 = 11$

We now consider our focused code associated with $d_1 = 15$ and $d_2 = 11$ and try to decode it by "adapting" the correcting capability of the outer decoder C_2 according to the number of erasures the inner decoder C_1 delivers.

We know that the maximum number of erasures C_1 can deliver (i.e., the maximum number of common errors C_1 can correct) is $\mathbf{e_{max}} = \lfloor \frac{d_1-1}{2} \rfloor = 7$. So for each value of $\mathbf{e}$ ($\mathbf{e} = 0, 1, ..., \mathbf{e_{max}}$), the outer decoder C_2 will correct up to $\mathbf{u} = \lfloor \frac{d_2-\mathbf{e}-1}{2} \rfloor$ uncommon errors. We get the following table:

e	0	1	2	3	4	5	6	7
u	5	4	4	3	3	2	2	1

Table 7.1: Values of $\mathbf{e}$ and $\mathbf{u}$ for $d_1 = 15$ and $d_2 = 11$

From the above data, we deduce the probability of correctly decoding the focused code associated with $d_1 = 15$ and $d_2 = 11$:

$$\begin{aligned} P_c \;=\; & P(\mathbf{e} = 7 \;;\; \mathbf{u} \leq 1) \\ & + P(\mathbf{e} = 6 \;;\; \mathbf{u} \leq 2) \\ & + P(\mathbf{e} = 5 \;;\; \mathbf{u} \leq 2) \\ & + P(\mathbf{e} = 4 \;;\; \mathbf{u} \leq 3) \\ & + P(\mathbf{e} = 3 \;;\; \mathbf{u} \leq 3) \\ & + P(\mathbf{e} = 2 \;;\; \mathbf{u} \leq 4) \\ & + P(\mathbf{e} = 1 \;;\; \mathbf{u} \leq 4) \\ & + P(\mathbf{e} = 0 \;;\; \mathbf{u} \leq 5) \end{aligned} \tag{7.3}$$

We can notice from the above expression of P_c that with such algorithm, we can correct up to 8 errors (7 common and 1 uncommon, or 6 common and 2 uncommon) and up to 5 uncommon errors while in the "minimal curve" method we can correct up to 7 errors and up to 3 uncommon errors.

Thus the probability of decoding error for the adaptive decoding will be:

$$P_d^{ad} = 1 - P_c \tag{7.4}$$

If we plot the performance of the adaptive decoder and compare it to the performance of the "minimal" curve we obtained earlier, we get:

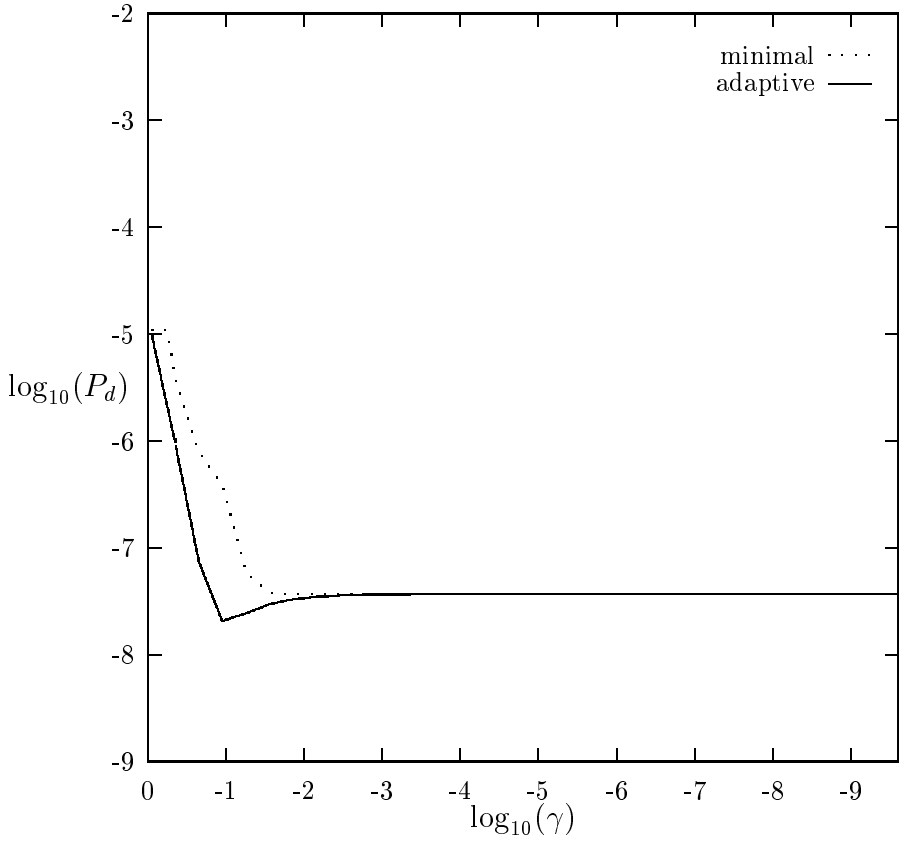

Figure 7.3: Adaptive performance vs "minimal curve" performance for a focused code associated with $d_1 = 15$ and $d_2 = 11$, for n=50, $\epsilon = 0.01$

From the above plot we clearly remark that the performance curve of the adaptive decoder is *better* than that of the "minimal" curve for high values of γ before behaving identically (as a 7-error correcting code performance) as γ decreases. Moreover by using adaptive decoding, we do not even need to determine the regions of γ that correspond to the decoding of different (t_1, t_2)-focused codes (like in the previous section).

7.1.4 Analytical Description of Adaptive Decoding

In the previous sections, we showed by using a particular example that we can achieve the *"best"* performance for the decoding of a focused code over an idealized SSC, associated with d_1 and d_2 (where d_1 and d_2 are the minimum distances of C_1 and C_2 respectively) via adaptive decoding.

The general adaptive decoding algorithm for focused codes associated with d_1 and d_2, is the same as the usual decoding algorithm for (t_1, t_2)-focused codes we described in section (7.1.1); but the analysis for which we declare a decoding error is different. For a given transmitted codeword,

- we pass the received codeword to the decoder of the detecting code C_0 and the inner code C_1 and observe the number of erasures $\mathbf{e}$ it delivers. We know that $0 \leq \mathbf{e} \leq \mathbf{e_{max}}$ where $\mathbf{e_{max}} = \lfloor \frac{d_1-1}{2} \rfloor$.
- We then pass the resulting block to the decoder of the outer code C_2. According to the number of erasures $\mathbf{e}$, C_2 will correct up to $\mathbf{u} = \lfloor \frac{d_2-\mathbf{e}-1}{2} \rfloor$ uncommon errors.

Note that the adaptive decoding concept we are presenting in this chapter, is *only* appropriate for the focused codes construction technique suggested in [1].

Using equations (7.3) and (7.4), we can directly write the analytical expression of the probability of decoding error for the adaptive decoder:

$$P_d^{ad} = 1 - \sum_{i=0}^{\mathbf{e_{max}}} \sum_{j=0}^{\mathbf{u}} \binom{n}{i+j} \binom{i+j}{j} \epsilon^{i+j} (1-\epsilon)^{n-i-j} \gamma^j (1-\gamma)^i \qquad (7.5)$$

where:

- $\mathbf{e_{max}} = \lfloor \frac{d_1-1}{2} \rfloor$ and $\mathbf{u} = \lfloor \frac{d_2-i-1}{2} \rfloor$.

- i=number of common errors.
- j=number of uncommon errors.
- n=blocklength per codeword.

For comparison, we recall the decoder error probability P_d of a $t_1 + t_2$-error correcting code (or a $(t_1 + t_2,0)$-focused code) and the decoder error probability P_d^f of a (t_1, t_2)-focused code where $t_1 + t_2 = \lfloor \frac{d_1-1}{2} \rfloor$ and $t_2 = d_1 - d_2$:

$$P_d = 1 - \sum_{i=0}^{t_1+t_2} \left(\begin{array}{c} n \\ i \end{array} \right) \epsilon^i (1-\epsilon)^{n-i} \tag{7.6}$$

$$P_d^f = 1 - \sum_{i=0}^{t_1+t_2} \sum_{j=0}^{\min(i,t_1)} \left(\begin{array}{c} n \\ i \end{array} \right) \left(\begin{array}{c} i \\ j \end{array} \right) \epsilon^i (1-\epsilon)^{n-i} \gamma^j (1-\gamma)^{i-j} \tag{7.7}$$

Given that we are working over an idealized SSC with ϵ fixed, we observe that:

- The second term in equation (7.5) is always bigger than the second term in equation (7.7) yielding $P_d^{ad} \leq P_d^f$ since by adaptive decoding we can correct more than t_1 uncommon errors..
- For very large values of γ, the second term in equation (7.5) is smaller than the second term in equation (7.6) yielding $P_d \leq P_d^{ad}$.
- For small values of γ, the second term in equation (7.5) is bigger than the second term in equation (7.6) (since for j=0 we have equality between the two terms) yielding $P_d^{ad} \leq P_d$. But as γ becomes extremely small, the terms in equation (7.5) corresponding for $j \geq 1$ can be neglected; which makes $P_d^{ad} = P_d$.

The remarks stated above are clearly illustrated in Figure (7.3) (for $d_1 = 15$ and $d_2 = 11$) where the curve of the performance of the adaptive decoder is always lower than the "minimal curve" which is a combination of the

performance curves of a (5,0)-focused code, a (4,2)-focused code and a (3,4)-focused code. Moreover the curve of the performance of the adaptive decoder is higher than that of a 7-error correcting code for high values of γ, then as γ decreases it goes below the latter before behaving identically at very small values of γ.

7.2 Adaptive Decoding For Focused Codes Used In Conjunction With PSK Modulation And Square Constellations

We can rewrite equation (7.5) as:

$$\begin{aligned} P_d^{ad} &= \sum_{i=\mathbf{e_{max}}+1}^{n} \sum_{j=0}^{n-i} \binom{n}{i+j} \binom{i+j}{j} \epsilon^{i+j}(1-\epsilon)^{n-i-j}\gamma^j(1-\gamma)^i \\ &\quad + \sum_{i=0}^{\mathbf{e_{max}}} \sum_{j=u+1}^{n-i} \binom{n}{i+j} \binom{i+j}{j} \epsilon^{i+j}(1-\epsilon)^{n-i-j}\gamma^j(1-\gamma)^i \end{aligned} \tag{7.8}$$

From the above equation, we deduce the decoded *symbol* error probability P_s as:

$$\begin{aligned} P_s &= \sum_{i=\mathbf{e_{max}}+1}^{n} \sum_{j=0}^{n-i} m_i \binom{n}{i+j} \binom{i+j}{j} \epsilon^{i+j}(1-\epsilon)^{n-i-j}\gamma^j(1-\gamma)^i \\ &\quad + \sum_{i=0}^{\mathbf{e_{max}}} \sum_{j=u+1}^{n-i} m_i \binom{n}{i+j} \binom{i+j}{j} \epsilon^{i+j}(1-\epsilon)^{n-i-j}\gamma^j(1-\gamma)^i \end{aligned} \tag{7.9}$$

where $m_i = \min\left[n, \frac{\mathbf{e_{max}}+i}{n}\right]$.

Using equations (7.5) and (7.9), we respectively plot the adaptive decoding performances (P_d vs E_s/N_0 and P_s vs E_b/N_0) of focused codes associated with d_1 and d_2 and used with M-ary PSK modulation and M-ary square constellations; and compare them to the respective performances of $t_1 + t_2$-error

correcting codes (using equation (7.6)) where $t_1+t_2 = \lfloor \frac{d_1-1}{2} \rfloor$ and $t_2 = d_1-d_2$. For different values of the blocklength n, M, d_1 and d_2, we construct codes associated with d_1 and d_2 and focused over the set $\mathbf{B}$ of odd-weight errors, compute the rate improvement they achieve over the t_1+t_2-error correcting codes and get their performance and coding gain plots.

16-ary PSK modulation with n=8, $d_1 = 5$ and $d_2 = 3$

1. Focused code C_f over GF(16):
   - C_0: (4,3) binary code with rate $R_0 = 3/4$.
   - C_1: (8,2) binary code with $d_1 = 5$ and rate $R_1 = 2/8$.
   - C_2: (8,6) shortened Reed Solomon code over GF(8) with $d_2 = 3$ and rate $R_2 = 6/8$.
   - Focused code rate: $R_f = (1-R_0)R_1 + R_0R_2 = 0.625$.
2. 2-error correcting code d=5: shortened Reed Solomon code over GF(16) with rate R=0.5.
3. Rate improvement: $\frac{R_f-R}{R} \times 100 = 25\%$.

32-ary PSK modulation with n=14, $d_1 = 9$ and $d_2 = 5$

1. Focused code C_f over GF(32):
   - C_0: (5,4) binary code with rate $R_0 = 4/5$.
   - C_1: (14,2) binary code with $d_1 = 9$ and rate $R_1 = 2/14$.
   - C_2: (14,10) shortened Reed Solomon code over GF(16) with $d_2 = 5$ and rate $R_2 = 10/14$.
   - Focused code rate: $R_f = 0.5833$.
2. 4-error correcting code d=9: shortened Reed Solomon code over GF(32) with rate R=0.4286.

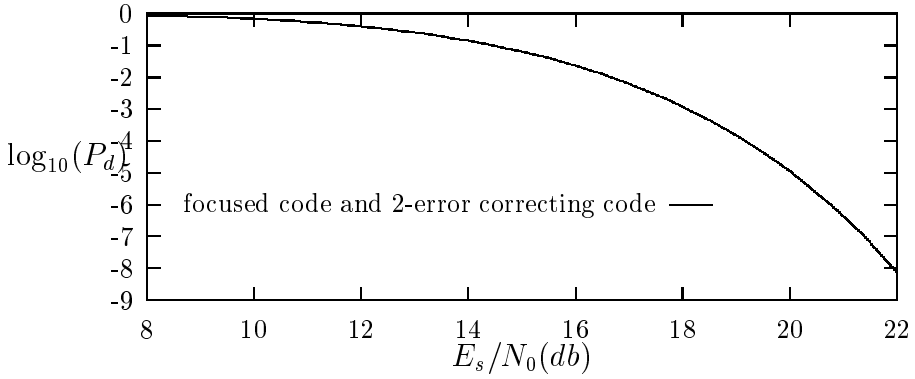

Figure 7.4: Adaptive performance of a focused code associated with $d_1 = 5$ and $d_2 = 3$ vs performance of a 2-error correcting code under 16-ary PSK modulation and n=8

3. Rate improvement: $\frac{R_f - R}{R} \times 100 = 36.1\%$.

8×8 **square constellation with n=14,** $d_1 = 9$ **and** $d_2 = 7$

1. Focused code C_f over GF(64):
   - C_0: (6,5) binary code with rate $R_0 = 5/6$.
   - C_1: (14,2) binary code with $d_1 = 9$ and rate $R_1 = 2/14$.
   - C_2: (14,8) shortened Reed Solomon code over GF(32) with $d_2 = 7$ and rate $R_2 = 8/14$.
   - Focused code rate: $R_f = 0.5$.
2. 4-error correcting code d=9: shortened Reed Solomon code over GF(64) with rate R=0.4286.
3. Rate improvement: $\frac{R_f - R}{R} \times 100 = 16.66\%$.

16×16 **square constellation with n=31,** $d_1 = 15$ **and** $d_2 = 11$

1. Focused code C_f over GF(256):

- C_0: (8,7) binary code with rate $R_0 = 7/8$.
- C_1: (31,6) binary code with $d_1 = 15$ and rate $R_1 = 6/31$.
- C_2: (31,21) shortened Reed Solomon code over GF(128) with $d_2 = 11$ and rate $R_2 = 21/31$.
- Focused code rate: $R_f = 0.6169$.

2. 7-error correcting code d=15: shortened Reed Solomon code (RS) over GF(256) with rate R=0.5484.

3. Rate improvement: $\frac{R_f - R}{R} \times 100 = 12.5\%$.

Comment

Figures (7.4)-(7.11) suggest to us that focused codes associated with d_1 and d_2 can achieve the same performance as $t_1 + t_2$-error correcting codes (where $t_1 + t_2 = \lfloor \frac{d_1 - 1}{2} \rfloor$ and $t_2 = d_1 - d_2$) under PSK and square constellation modulations (while having a higher rate) if we decode them using the adaptive decoding algorithm.

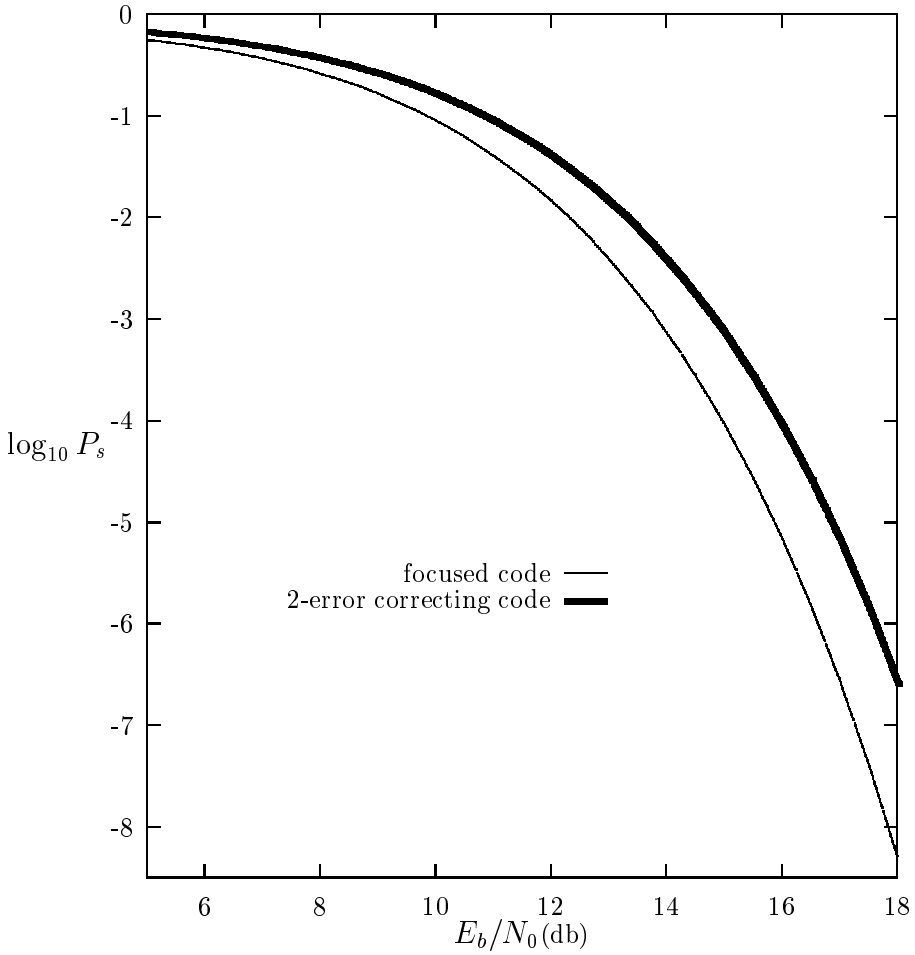

Figure 7.5: Adaptive Coding Gain of a focused code associated with $d_1 = 5$ and $d_2 = 3$ vs performance of a 2-error correcting code under 16-ary PSK modulation and n=8

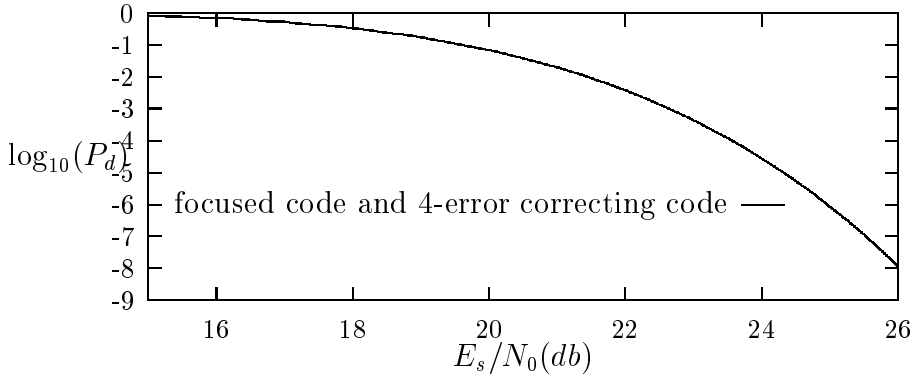

Figure 7.6: Adaptive performance of a focused code associated with $d_1 = 9$ and $d_2 = 5$ vs performance of a 4-error correcting code under 32-ary PSK modulation and n=14

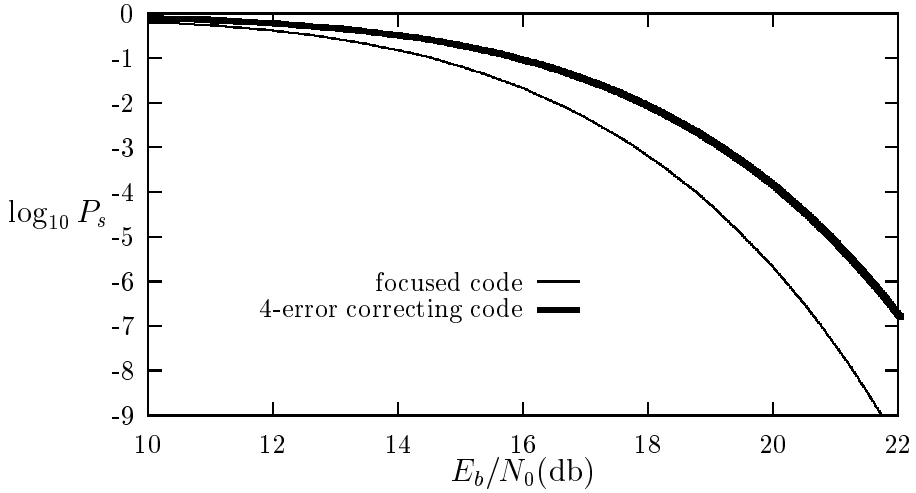

Figure 7.7: Adaptive Coding Gain of a focused code associated with $d_1 = 9$ and $d_2 = 5$ vs performance of a 4-error correcting code under 32-ary PSK modulation and n=14

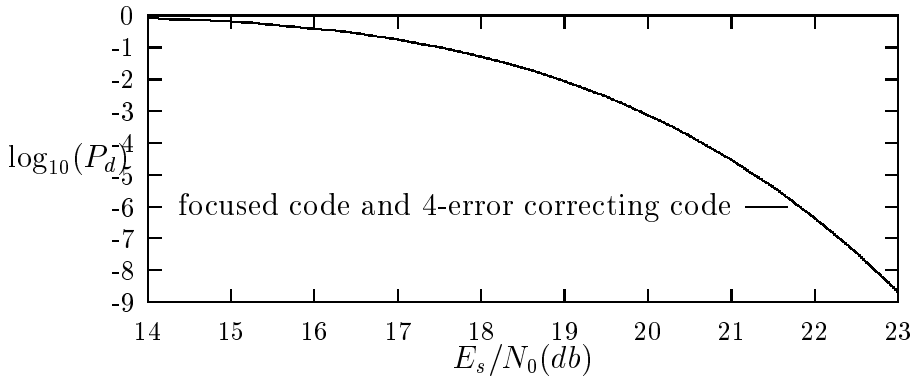

Figure 7.8: Adaptive performance of a focused code associated with $d_1 = 9$ and $d_2 = 7$ vs performance of a 4-error correcting code under 64-ary square constellation and n=14

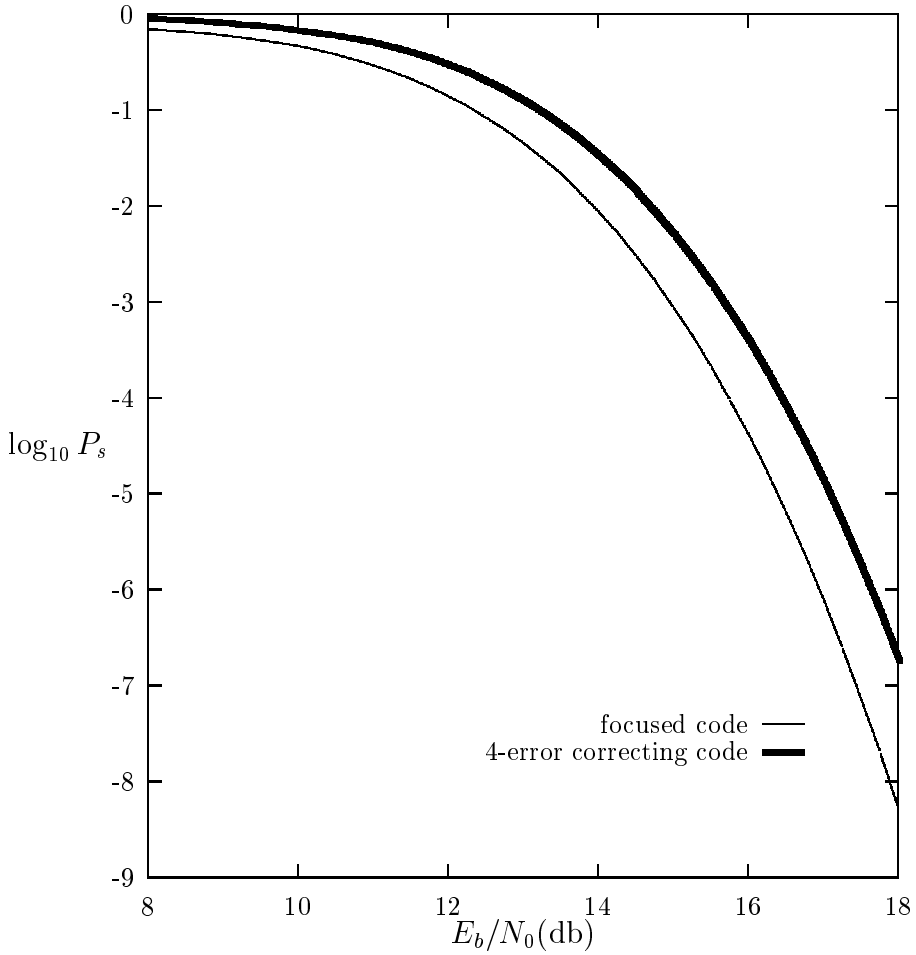

Figure 7.9: Adaptive Coding Gain of a focused code associated with $d_1 = 9$ and $d_2 = 7$ vs performance of a 4-error correcting code under 64-ary square constellation and n=14

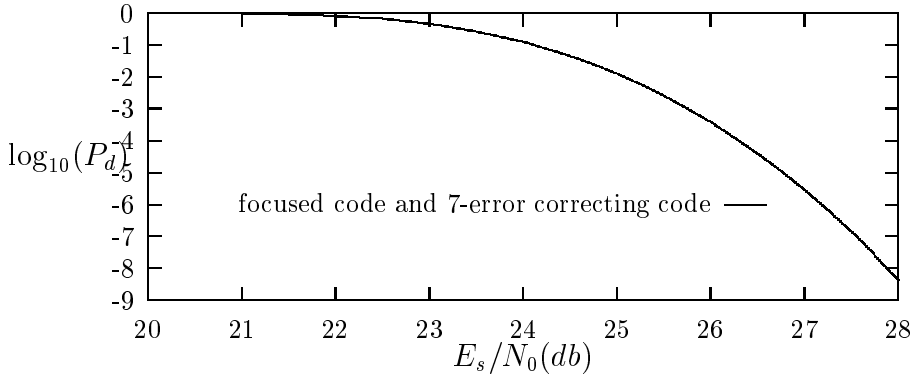

Figure 7.10: Adaptive performance of a focused code associated with $d_1 = 15$ and $d_2 = 11$ vs performance of a 7-error correcting code under 256-ary square constellation and n=31

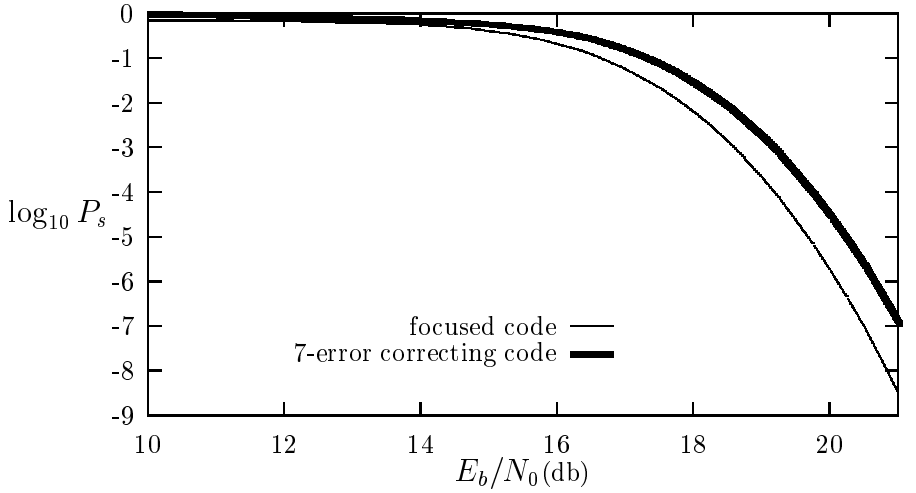

Figure 7.11: Adaptive Coding Gain of a focused code associated with $d_1 = 15$ and $d_2 = 11$ vs performance of a 7-error correcting code under 256-ary square constellation and n=31

Chapter 8

Conclusion

In this thesis, we investigated the performance of (t_1, t_2)-focused codes in terms of rate/performance tradeoffs with respect to "idealized" skewed channels as well as realistic non-binary modulation schemes.

After a brief introduction of the concept of focused error control codes on channels with skewed errors and a quick review of results concerning their construction (chapter 2), we derived in chapter 3, the general expression for the decoder error probability P_d of focused codes, and studied their performance over "idealized" skewed symmetric channels with independent parameters ϵ and γ. In the case where ϵ was held constant, we computed P_d as a function of γ and obtained the critical value of γ, γ_{crit}, beneath which a (t_1, t_2)-focused code performs identically as a traditional $t_1 + t_2$-error correcting code (i.e., P_d(focused code) $\approx$ P_d(traditional code)). Similarly, in the case where γ was held constant, we computed P_d as a function of ϵ and obtained the critical value of ϵ, ϵ_{crit}, beyond which P_d(focused code) $\approx$ P_d(traditional code). We then determined a general condition under which performance matching is achieved between (t_1, t_2)-focused codes and $t_1 + t_2$-error correcting codes.

In chapter 4 and 5, we derived respectively the analytical expressions for the parameters of additive white Gaussian noise channels associated with

M-ary PSK modulation and square constellations. We then analyzed the performance of focused control codes used in conjunction with M-ary PSK modulation and square constellations respectively and compared it to the performance of traditional codes. Some simulations on the focused codes performances were also done in order to check the accuracy of the results derived analytically.

In chapter 6, we provided some numerical results by constructing (t_1, t_2)-focused codes over GF(q) and computing the code rate improvements they achieve over $t_1 + t_2$-error correcting codes while having an identical performance, for different values of the code blocklength and q. Considerable rate improvements were obtained:

- A (0,2)-focused code over GF(16) achieved a rate gain of 25% over a 2-error correcting Reed Solomon code for a blocklength n=8.
- A (0,4)-focused code over GF(32) achieved a rate gain of 36.1% over a 4-error correcting Reed Solomon code for a blocklength n=14.
- A (1,2)-focused code over GF(64) achieved a rate gain of 23.34% over a 3-error correcting Reed Solomon code for a blocklength n=11.
- A (1 2)-focused code over GF(256) achieved a rate gain of 27.5% over a 3-error correcting Reed Solomon code for a blocklength n=11.

We also calculated the coding gains (in signal to noise ratio) (t_1, t_2)-focused codes achieve over $t_1 + t_2$-error correcting codes and uncoded block messages under PSK and square constellation modulations. We remarked that focused codes achieved substantial coding gains (1.5 db on the average) in E_b/N_0 over uncoded messages for all values of the blocklength n. Moreover, the focused codes achieved also fair amounts of coding gain over the error correcting Reed Solomon codes, especially for small values of n. For example, under a 32-ary PSK modulation with n=8, a (0,2)-focused code achieved a coding gain of 1.13 db over a 2-error correcting RS code at $P_s = 10^{-6}$ (figure (6.3)). And a (1,2)-focused code used with a 256-ary square constellation and n=11 achieved a coding gain of 1.06 db at $P_s = 10^{-6}$ over a 3-error correcting RS code.

Finally, the concept of adaptive decoding of focused codes used over "idealized" skewed symmetric channels and used in conjunction with M-ary PSK modulation or square constellations, was introduced in chapter 7. We presented some examples in which we can attain performance matching between focused codes associated with d_1 and d_2 and $t_1 + t_2$-error correcting codes (where $t_1 + t_2 = \lfloor \frac{d_1 - 1}{2} \rfloor$ and $t_2 = d_1 - d_2$) if we decode them using an adaptive decoding algorithm.

Bibliography

[1] Tom Fuja and Chris Heegard, "Focused Codes for Channels with Skewed Errors", *IEEE Transactions on Information Theory, July 1990.*

[2] Bernard Sklar, "Digital Communications", *Prentice Hall 1988.*

[3] S. Benedetto, E. Bigliery, V. Castellani, "Digital Transmission Theory", *Prentice Hall 1987.*

[4] T. Verhoff, "An Updated Table of Minimum-Distance Bounds for Binary Linear Codes", *IEEE Transactions on Information Theory, Vol. IT-33,No.5, September 1989, pp. 665-680.*

[5] Shu Lin and Daniel J. Costello, "Error Control Coding: Fundamentals and Applications", *Prentice Hall, 1983.*

[6] W. Wesley Peterson and E. J. Weldon Jr., "Error Correcting Codes", *Second Edition, The MIT Press, 1972.*