Chip Shaping and Channel Coding for CDMA

J.P. Chaib

B. Eng.

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of
Master of Engineering.

Department of Electrical Engineering
McGill University
Montréal, Québec

Contents

List of Tables

List of Figures

Abstract

This thesis considers waveform shaping and error control coding methods to improve the performance and capacity of an asynchronous DS-CDMA communications system. The uplink of the IS-95 cellular CDMA standard is an example of such a system. In this work, we develop a criterion to measure the merits of a waveform shaping filter from the points of view of interference reduction and bandwidth occupancy. This criterion allows us to derive quasi optimal waveforms. It is shown, for example, that the pulse shaping filter of the IS-95 standard is close to optimal.

We also show, in the first part of this thesis, that pseudo random sequence spreading is an inefficient way of expanding bandwidth. Indeed, we prove that the optimal sharing of bandwidth spreading between PN sequences and error control coding is obtained when all the spreading is due to the error control code. The role of PN sequences for user separation is not diminished, while the system benefits from the added coding gain.

In order to realize as much as possible of the potential coding gain, good very low rate codes are needed. The second part of this thesis focuses on the design of specific low rate error control codes for CDMA systems. We consider a new coding scheme, based on the combination of trellis codes and first-order Reed-Muller codes. We develop two families of codes based on this scheme, and study their performance both analytically and through simulations. We find the performance of our codes to be superior to that of other families of very low rate codes, such as the orthogonal, biorthogonal, and superorthogonal convolutional codes, and the error control code specified in the IS-95 standard.

Sommaire

Ce mémoire porte sur des méthodes de filtrage de signal et de codage de contrôle d'erreur afin d'améliorer la performance de systèmes de communications asynchrones utilisant l'Accès Multiple à Répartition par Code (AMRC) à séquences directes (DS-CDMA). Le lien des usagers vers les stations de base du standard IS-95 (système cellulaire employant l'AMRC) en est un exemple. Dans un premier temps, nous nous employons à mettre au point un critère quantitatif qui mesure la capacité d'une forme d'onde à rejeter les interférences, tout en tenant compte de la bande passante qu'elle occupe. Ce critère nous permet de trouver des formes d'ondes quasi-optimales. Nous montrons par exemple que la forme d'onde spécifiée dans le standard IS-95 est voisine des formes d'ondes optimales.

Dans la première partie de ce mémoire, nous démontrons aussi que l'extension du spectre par des séquences pseudo-aléatoires, bien que très facile d'implémentation, se révèle inefficace du point de vue du codage de canal. En effet, il s'avère que le partage de l'extension du spectre entre les séquences pseudo-aléatoires et le codage de canal devient optimal lorsque le codage de canal est entièrement responsable de l'extension du spectre. Le rôle des séquences pseudo-aléatoires afin de distinguer les différents usagers n'est pas diminué, tandis que le système bénéficie, en plus, du gain de codage.

Afin de réaliser, même en partie, les gains de codage potentiels, de bons codes de contrôle d'erreurs à bas débit sont requis. La deuxième partie de ce mémoire porte sur la conception de codes de contrôle d'erreurs pour les systèmes AMRC asynchrones. Nous proposons un nouveau procédé de codage, basé sur l'association de codes Reed-Muller du premier ordre avec des codes treillis (trellis codes). Par la suite, nous construisons deux familles de codes en utilisant ce procédé, et étudions leur performance grâce à des méthodes analytiques et à des simulations par ordinateur. Il ressort que les codes ainsi construits sont supérieurs à ceux déjà connus, dont les codes convolutionaux orthogonaux, biorthogonaux, et superorthogonaux, de même que le code du standard IS-95.

Acknowledgements

First and most of all, a big "thank you" to my family and to my friends! They are the ones who make everything worthwhile. My friends of the TSP lab have been especially supportive during my graduate studies, and had to put up with my loud conversations and my sometimes extensive use of the computers.

I must acknowledge the invaluable help and guidance of my supervisor, Dr. Harry Leib, and his important contribution to my studies. Without him, I would not have reached this point. I would like to also thank the external reviewer of my thesis, Dr John Lodge, for his careful review of the manuscript and his helpful comments.

Finally, I am grateful to the Canadian Institute of Telecommunications Research (CITR) and to the Fonds pour la Formation des Chercheurs et l'Aide à la Recherche (FCAR) for their financial support.

Merci à tous ceux qui m'ont témoigné de leur aide, leur amitié et leur soutien. Merci aussi à ceux (et à celles!) qui, d'un sourire parfois, ont su éclairer toute une journée.

Chapter 1
Introduction

The last decade has witnessed a phenomenal expansion in wireless communications which is now enjoying its fastest growth ever. Cordless and cellular phones, paging systems, and wireless data networks are becoming part of our every day lives. This growth is mainly attributable to analog technologies of the 1970s which are mature today [1]. By the first couple of decades of the 21st century, there will be an equal number of wireless and conventional wireline customers [1]. The wireless systems subscribers will expect a wide range of services together with a high reliability from their service providers. In the difficult environment of radio channels, where bandwidth is scarce and interference of many kinds severe, only advanced digital techniques combined with intelligent and flexible network management can help meet the growing demand.

In the first section of this chapter, we introduce the fundamentals of multiple user wireless communications, and discuss the different multiple access methods. We then emphasize Code-Division Multiple Access (CDMA) and show the potential advantages that it provides over other methods. Finally, we define the scope of our research, and present the structure of the thesis.

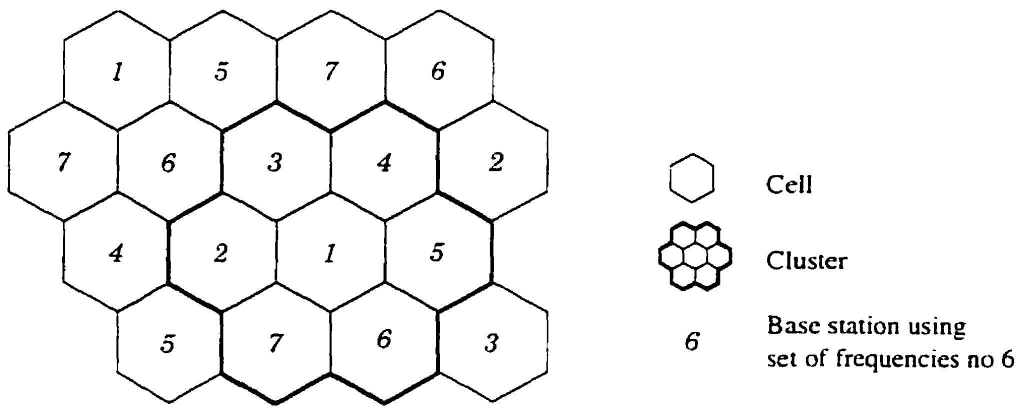


Figure 1.1: Frequency reuse in cellular communications systems. Cells with the same index use the same set of frequencies.

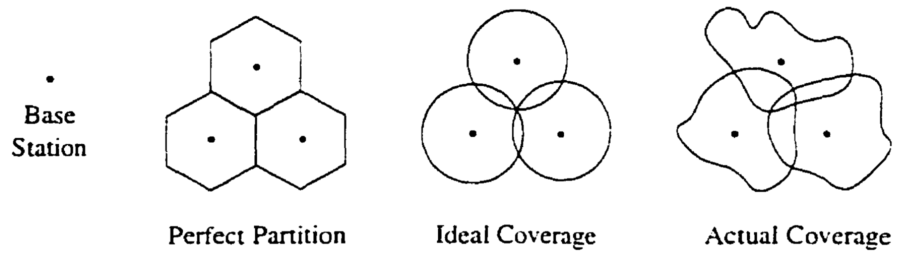


Figure 1.2: Ideal and actual cell coverage. adapted from [2].

1.1 Multi-User Mobile Communications

1.1.1 The Cellular Concept

The cellular concept is at the heart of effective wireless communications systems. It allows high capacities within a limited spectrum. and reduced mobile transmitted power. In a cellular system. the service area is partitioned into a number of smaller regions called *cells*. each of which is served by a single base station. as illustrated in Figure 1.1. If each cell can accommodate M simultaneous users. the total capacity is M times the number of cells. with a density of M users per cell area. Naturally. the regular hexagonal shape used to depict a cell is only an imaginary visual aid. There are no physical boundaries to the electro-magnetic waves. which means that cells actually interfere with each other. a major problem for network planners. As shown in Figure 1.2, a cell is determined by the area it covers. which. ideally. is a circle. and

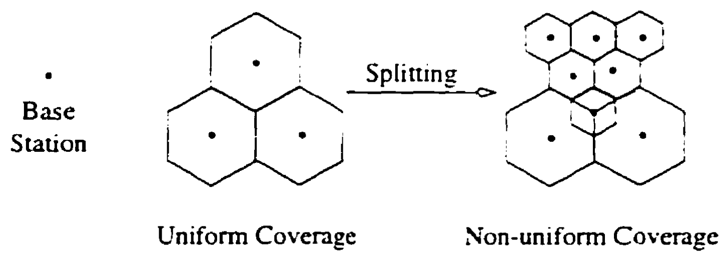


Figure 1.3: Cell splitting to accommodate non-uniform traffic density.

in practice can deviate largely from the ideal. For example. an active user might not be communicating with the closest base station. due to shadowing. multipath fading. or maybe just because of the unavailability of channels. Hence the area covered by a cell is not only loosely defined. but also varies with time.

The problem of inter-cell interference can be solved by allocating different frequency bands to each cell. Naturally. if this is done over the whole service area. the capacity becomes no better than that of a huge single cell. However. a good compromise is reached through *frequency reuse*. which limits the interference between *adjacent* cells. All cellular radio networks employ some degree of frequency reuse. For example. in Figure 1.1. each cell employs a set of frequencies which is distinct from that of its neighbors. The frequency reuse factor is the number of sets of frequency which partition the allocated spectrum (in Figure 1.1. the frequency reuse factor is 7). A set of adjacent cells which. together. use the whole available spectrum with no overlap is called a cluster.

The communication link from the mobile to the base station is called the *uplink*. whereas the *downlink* refers to the inverse direction. In most wireless systems. the uplink and downlink employ disjoint sets of frequencies: this is called frequency division duplexing (FDD). We assume that all wireless systems discussed in this thesis employ FDD.

The capacity of a cellular system can be improved in dense traffic areas by reducing the cell radii. an operation called *cell splitting*. and which is illustrated in Figure 1.3. This technique allows the coverage of areas with non-uniform traffic density. and can also adapt the coverage to long-term variations in traffic density. Cell splitting is accompanied by a corresponding reduction in transmitted power to avoid interference. Side advantages include a reduction in the average transmitted power. increasing battery life and/or reducing the weight of the mobile units. However. the number of base stations also increases. When a mobile crosses from one cell to another. it switches its communication to the corresponding base station. in an operation called a *hand-off.* The smaller is the cell size. the larger is the frequency of hand-offs that must be handled. which implies the need for robust and fast hand-off strategies. Hence. reducing the cell radii in a given area improves the system capacity. but implies higher costs and complexity. Moreover. if a hand-off cannot be completed. the call is dropped. an event which should be made as rare as possible. Indeed. the blocked call rate (when a call cannot be started) and dropped call rate are the main indicators of the system saturation. Users can tolerate small blocked call rates (at peak hours) but virtually no dropped calls. Since each hand-off is a potential dropped call. there is a limit on how small cell sizes could be made. even if the cost of a large number of base stations is acceptable.

Other methods of improving the capacity of a cellular system include dynamic channel allocation. whereby cells can "borrow" unused frequencies from neighboring cells. and sectorization where directional antennas divide each cell into non-interfering *sectors.* Power control. carefull site selection. and antenna downtilt on sectorized sites. can help reduce the interference to other cells. For voice services. efficient speech coding provides a great reduction in the data rate. which in turn. allows more voice channels per unit bandwidth. We will not examine further these methods. but the interested reader is referred to [1]. Although these techniques might seem very powerful in achieving high system capacities. one should not forget the very severe environment of wireless systems which makes all these techniques necessary.

Even if the severe radio environment were somehow tamed. the major obstacle to accommodating a large number of users in a given cell is the restricted bandwidth available to service providers. Indeed. the radio frequency spectrum is a scarce resource. and regulatory agencies only parsimoniously allocate chunks of bandwidth to service providers. who need to use it as efficiently as possible. Hence the cell capacity (and therefore the network capacity) is determined. to a large extent. by the spectrum sharing strategy among the users within a single cell. Equally important. are the coding and modulation which determine the bandwidth of a single user signal.

1.1.2 Multiple Access in Wireless Communications

In this section. we look at the main three methods of providing multiple-access communications between users and their base station: frequency-division multiple access (FDMA). time-division multiple access (TDMA). and code-division multiple access (CDMA). Although we discuss these three techniques separately. wireless systems may use a combination of two or even all three of them.

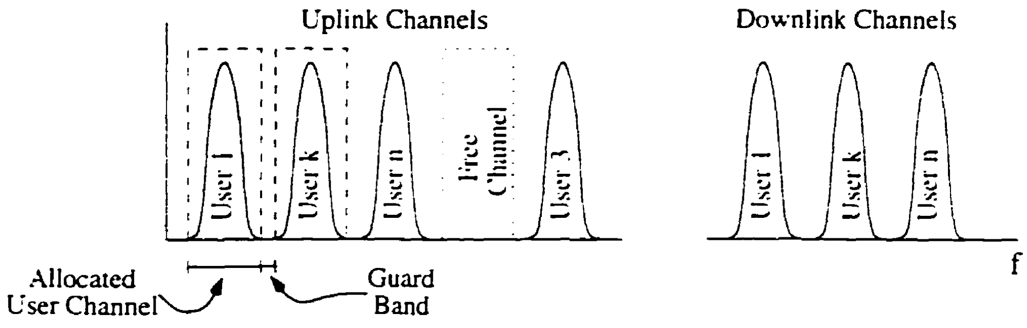


Figure 1.4: Frequency-Division Multiple Access.

Frequency-division multiple access is the user separating technique employed in current analog cellular systems in North America (Advanced Mobile Phone System or AMPS). During a call. a user is assigned a pair of frequency channels. one for the uplink and one for the downlink. He (or she) remains the only user of these channels until the end of the call. or until a hand-off occurs.

Time-division multiple access divides time into time frames and time slots. as shown in Figure 1.5. Since frequency division duplexing is employed. during a call. each user is assigned a time slot on the uplink frequency. and a time slot on the downlink frequency. Such a system requires that users be synchronized. which adds to the system complexity. On the other hand. a single demodulator followed by a demultiplexer is required at the base station receiver end. to demodulate all the users. Most TDMA systems actually combine TDMA with FDMA. in order to relax

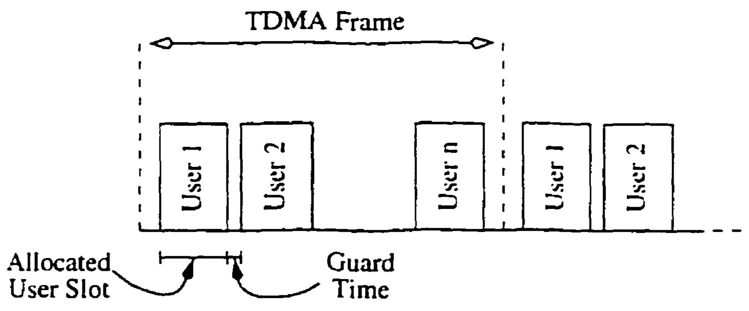


Figure 1.5: Time-Division Multiple Access.

the synchronization requirements and therefore the system complexity. For example. the GSM standard (Global System for Mobile. originally Groupe Spécial Mobile) adopts this technique. with 8 time slots per frequency carrier.

TDMA and FDMA are sometimes referred to as orthogonal multiple access. since users are orthogonal to each other. either in time or in frequency. Code-division multiple access. instead. uses "spreading sequences" to distinguish between users. which therefore employ the same bandwidth at the same time. In asynchronous CDMA. the signals of different users are not orthogonal. and therefore interfere with each other. The same is true for synchronous CDMA with non-orthogonal spreading sequences. However. as we will see in the next section. this interference can be controlled. and even completely removed. at least in theory.

1.2 Code-Division Multiple Access for Wireless Communications

Code-division multiple access employs spread spectrum techniques to generate a transmitted signal with a much larger bandwidth than the data rate dictates. Although several bandwidth spreading techniques are possible. direct-sequence (DS) is the main candidate for wireless communications. In DS-CDMA. a pseudo-random binary sequence (PN sequence) is used to transform the message into a wide-band signal. as illustrated in Figure 1.6. Each binary symbol of the spreading sequence is called a chip. The chip rate divided by the data rate yields the spreading factor.

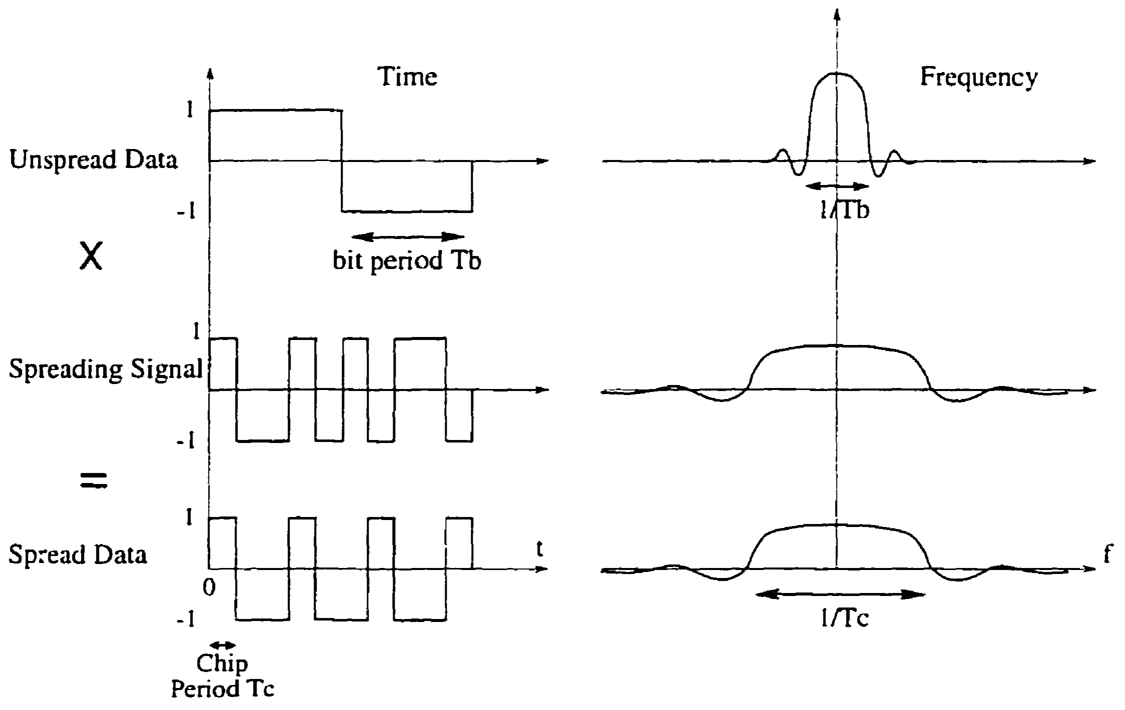


Figure 1.6: Direct-sequence CDMA in time and frequency.

When the spreading factor, sometimes loosely referred to as the processing gain, is very large, the power spectral density of the transmitted signal resembles that of background noise, and is much lower than that of the original narrow-band signal.

1.2.1 Multiple Access And User Separation

In a DS-CDMA system, users spread their signal by using *different* spreading sequences or spreading codes (hence the name CDMA). A single-user receiver can then be implemented as in Figure 1.7. The received signal is multiplied by a local replica of the desired PN sequence, which in effect despreads the signal. Note that if the local replica of the desired PN sequence is delayed by one chip period, then the despreading modulator yields a noise-like signal, whose level is inversely proportional to the spreading factor. Although this emphasizes the importance of synchronization in CDMA, it also reveals its resistance to multipath, since versions of the signal which are delayed by more than a chip period will only contribute to a small fraction of noise like perturbation. Moreover, replicas of the signal which are delayed by less than a chip period will also be partly rejected. Since chip periods are relatively short. DS-CDMA is inherently robust to multipath interference.

If the system is synchronous, then orthogonal sequences may be used to completely suppress the other-user interference. However, the processing gain then limits the maximum number of users. In asynchronous DS-CDMA, or in synchronous DS-CDMA with non-orthogonal sequences, the other-user interference is inversely proportional to the processing gain. The number of users is not hard-limited, but the system performance degrades with increasing traffic. In a cellular system which employs DS-CDMA, it is difficult to synchronize the uplink, whereas the downlink (point to multi-point) is naturally synchronous. Hence, in the remaining part of this thesis, we assume that the uplink uses asynchronous DS-CDMA and the downlink synchronous DS-CDMA.

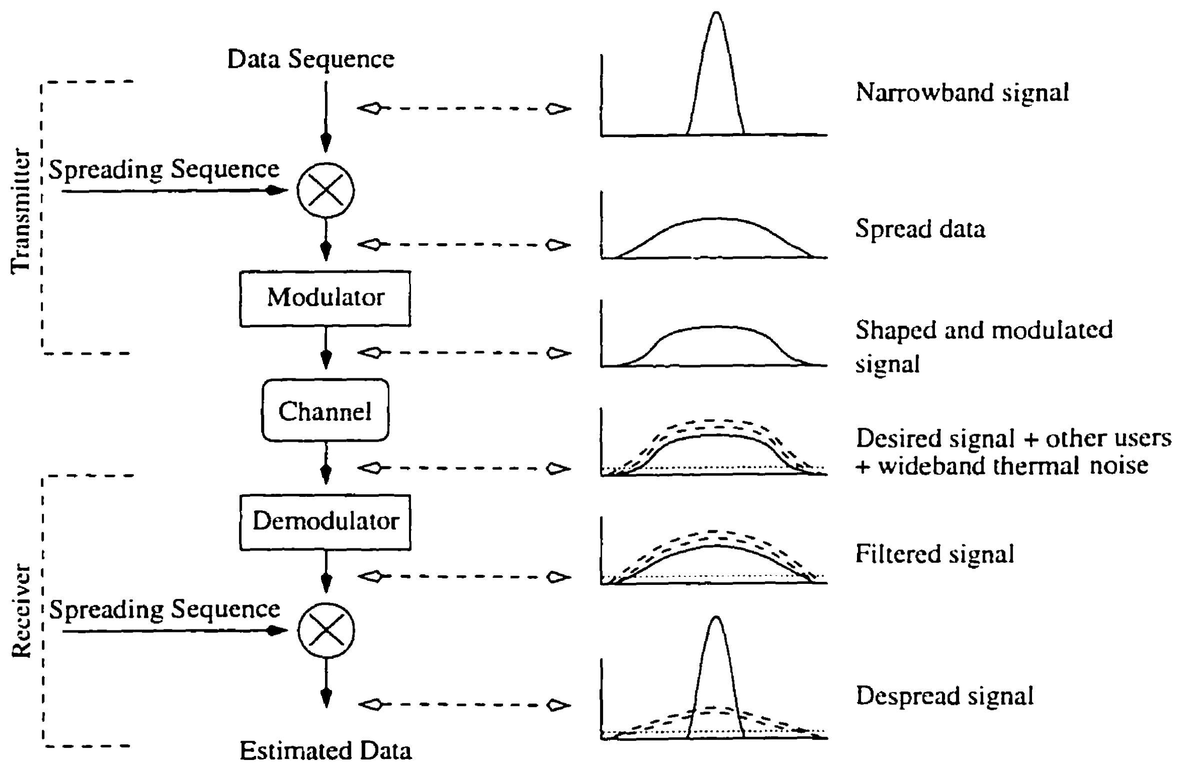


Figure 1.7: DS-CDMA with single-user receiver.

The single-user receiver of Figure 1.7 considers all interference as noise. Naturally, the other-user interference is not random, and contains information. Multi-user receivers which jointly demodulate the multi-user signal can significantly outperform the single-user receiver. Actually, the optimal multi-user receivers can completely eliminate the effect of multi-user interference [3], so that the performance for each user is limited by thermal noise and other channel perturbations. Naturally, it would be very inefficient to implement multi-user receivers on the downlink, since each mobile unit would have to demodulate the signals of all the users in the cell. Hence multi-user receivers are intended for the uplink channel. Unfortunately, the complexity of an optimal asynchronous multi-user receiver is huge (exponential in the number of users), whereas a bank of single-user receivers is only linear in the number of user. A substantial amount of research has thus been conducted on sub-optimal multi-user receivers, in an attempt to trade performance for a reduction in complexity (see for example [4, 5]). However, despite their poor performance, single-user receivers are simpler, cheaper, and more flexible since they work independently. For example, one could just add a bank of single user receivers to a base station to increase the maximum number of supported users.

1.2.2 Advantages of DS-CDMA

By spreading the signal over a large frequency band, DS-CDMA is very efficient in combating frequency selective fading, which is a common perturbation in wireless communications. Indeed, multipath components may be independently received which reduces dramatically the effect of multipath fading. The wide-band CDMA approach offers a form of diversity to combat frequency selective fading, since typically only a portion of the signal bandwidth is affected. Indeed, typical fade bandwidths are of the order of 100-200 kHz, which is comparable to the bandwidth of a GSM channel, but much less than the IS-95 bandwidth (1.25 MHz).

Unlike orthogonal multiple access schemes, DS-CDMA makes a full use of the allocated bandwidth, since no guard times or guard bands are necessary to separate the users. Moreover, the ability of CDMA to average co-channel interference over a wide bandwidth allows for much higher frequency reuse factors, which makes CDMA a prime candidate for multiple access in cellular communications. Typical cell reuse patterns (number of cells per cluster) are 7 for FDMA, 4 for TDMA (GSM standard for example), and 1.33 for CDMA [6]. Using these figures, it is argued in [6] that the capacity of a cellular CDMA system would provide 4 times the capacity of TDMA and 20 times that of AMPS (analog FDMA). These gains however have not been reached in practice.

The large bandwidth spreading that is required in DS-CDMA allows the use of powerful very low rate error control codes in asynchronous systems. In turn, such codes lower the required E_b/N_0 to obtain the desired communications quality, which allows more users to join in (or a higher level of interference), or reduces the interference to other cells. The net effect is an improvement of the overall system capacity.

Another primordial advantage of DS-CDMA is the so-called *soft capacity*. In orthogonal multiple access schemes, it is impossible to add even one more channel if the system is full, resulting in a blocked or dropped call. Meanwhile, the grade of service is independent of the number of users as long as the system has not reached its capacity. Such behavior is sometimes called hard capacity, in contrast with DS-CDMA where the quality of the communication degrades gracefully as the number of users increases. Hence, if a system is operating at capacity, then all the users enjoy the required grade of service, but adding one or a few more users is possible, resulting in some small performance degradation. Often, it is preferable from the service provider's point of view to allow a small degradation in the quality of the transmission for a short period of time (until some call is completed), avoiding therefore a blocked or a dropped call. The soft capacity property which is conveniently exploited in CDMA results in a traffic capacity improvement.

Voice activity can be well exploited in the voice channels of a DS-CDMA system. In a typical conversation, each party talks for less than 50% of the time. The actual figure is between 30% and 45%. Since speech pauses contain no information, they need not be transmitted (a background noise generator at the mobile receiver can be provided for listening comfort, or background noise can be coded at a much lower rate). In orthogonal multiple access, it is virtually impossible to exploit the voice activity by relinquishing the uplink channel when the user is not talking (and the downlink channel when the party is not talking), due to the delay involved in requesting and obtaining a channel, and to the increased dropped call probability if a channel cannot be allocated. However, with CDMA, it is possible to reduce the data rate during speech pauses, thereby increasing the processing gain. The mobile can then reduce its transmit power for the same received SNR (because of the larger processing gain) which saves battery life and most of all reduces interference to other users, thereby improving the system capacity [7].

The fact that the same frequency can be used over and over from cell to cell allows the mobile to perform *soft handoffs*. In a soft handoff between two base stations, the mobile communicates with both until the handoff is complete. This is possible because the signals coming from the two stations appear to be delayed versions of each other to the mobile, which are combined by a Rake receiver. Both stations must however send the same data and control bits. Soft handoffs can greatly reduce the probability of dropped calls during handoffs, and makes the handoff unnoticeable to the user, unlike AMPS or current GSM.

1.2.3 The CDMA Challenges and Promises

The main disadvantage of CDMA is its high sensitivity to the near-far effect [8]. The near-far effect occurs in the uplink, where the received power levels at the base station may not be equal. For example, if all mobile units had the same fixed transmitter output power, then the received signal would be dominated by the mobile closest to the base station. This would have a disastrous effect on the system capacity. The near-far effect is combatted with the use of tight power control, which combines both open and closed-loop power control. In open-loop power control, the mobile adjusts its transmit power based on the received power from the base station. Open-loop power control is fast, and can track rapid variations in the signal strengths that may occur with fading. However, it is not accurate since the uplink and downlink channels are not symmetric. Because of the high sensitivity of CDMA to received power imbalances at the base station (a 1 dB imbalance can lead to a 30% reduction in capacity), closed-loop power control is also necessary. In closed loop-power control, the base station sends a few control bits at regular interval, telling the mobile to increase or decrease its power. Although slower than open-loop power control, it is much more accurate. Therefore, a combination of open and closed-loop power control, where the latter refines the power level determined by the former, results in a tight power control scheme which ensures equal received power levels at the base station.

Many of the advantages of CDMA that were outlined in the above sections also come at some cost. Rake receivers, decoders of powerful low rate codes, voice activity detectors, power control schemes, all require extra hardware (and maybe some software) which adds to the system cost/complexity. On the other hand, CDMA does not require user synchronization, nor does it need equalizers [6], which represents significant savings in cost and complexity.

The promise of CDMA is an efficient cost-effective cellular system with a capacity larger than that obtainable with other multiple-access schemes. The challenge is to deliver on the promise. The IS-95 standard developed by Qualcomm, and the recent bandwidth allocation for CDMA are first steps in this direction. In Canada, some cellular service provides, such as Bell Mobility and ClearNet, have chosen CDMA technology, whereas others such as Microcell are going with GSM (TDMA/FDMA). For the moment, many GSM systems are already up and running, throughout the world, whereas CDMA has been used only in some limited areas (LA, Hong Kong, Korea). Is there room for two multiple-access technologies, or will one take over the market? Will there be one world standard, or will each global market more likely go with their own standards? The question is already difficult to answer from a technological point of view. When we take into account the heavy weight that politics have in regulatory decisions, it appears clearly that these questions are beyond the scope of this thesis.

1.3 Thesis Scope and Outline

In this work, we look at ways to improve the capacity of the uplink of a DS-CDMA system, at the physical layer level. As in the proposed IS-95 standard, we assume single-user receivers at the base station. We concentrate our efforts at the modulation and coding levels, and investigate methods to improve the reliability of the transmission, which in turn would provide a capacity increase for the same transmission quality. Since we are looking at modulation and coding, we consider single cell systems, because the effect of other cells is only to raise the interference level.

In the first part of this thesis, we analyze the performance of a DS-CDMA system without coding. We show that careful waveform shaping specific to the DS-CDMA environment can dramatically improve the performance over rectangular shaping, and we construct such effective waveforms for DS-CDMA. We then show, through random coding arguments, that the system performance can be greatly increased through powerful low rate codes, and that channel coding should account for all of the bandwidth spreading as opposed to PN sequence spreading. In Chapter 3, we look at known very low rate codes, and construct our own code family, based on the combination of trellis and first-order Reed-Muller codes. By default, we call these codes Trellis/Reed-Muller codes. Finally, in Chapter 4, we provide a complete comparison of our codes with the other known very low rate codes. The conclusion of this comparison is a net performance advantage of our codes. In the conclusion part, we elaborate on the benefits of using careful pulse shaping and Trellis/Reed-Muller codes in a DS-CDMA cellular systems, and give directions for further refinements of our work.

Chapter 2

Chip Shaping and Channel Coding

In this work, we focus our attention on asynchronous DS-CDMA with BPSK modulation and single-user detectors. It is shown in [11] that, for a fixed bandwidth spreading, the best performance is obtained when the error control code is responsible for the entire bandwidth spreading. The work in [11] however does not consider the issue of chip waveform shaping. This chapter aims at quantifying the potential improvements that chip shaping and channel coding can bring to a DS-CDMA system. The approach is essentially based on a random coding argument, and does not put any prior restriction on the chip pulse shape, except that the intersymbol interference, if any, is neglected. In [12], it is shown that for a strict bandwidth definition (i.e. when all the signal power is contained in the allocated band), the signal to interference ratio is maximized when the pulse shape is a sinc function. In this work, we relax the bandwidth definition, and derive a criterion for chip pulse design, which takes into account the bandwidth of the pulse shapes. We also explore the effects of chip waveform shaping in conjunction with error control coding not only to gain an insight into the fundamental capabilities of CDMA, but also to show how chip shaping with bandwidth limitation can improve the system performance.

The chapter is structured as follows. A CDMA system model is presented in Section 2.1. A comparison criterion for chip pulse shapes is introduced in section 2.2. Section 2.3 considers the advantage of expanding bandwidth by error control coding over bandwidth expansion via PN sequences only. Furthermore, random coding techniques are employed to analyze the effects of chip shaping when error control is used with CDMA. Section 2.4 presents a comparison between CDMA and orthogonal multiple access (TDMA/FDMA) when both schemes use error control coding.

2.1 CDMA System Description and Analysis

We describe here the model that we adopt for an asynchronous DS-CDMA system. We view the system in three parts: the modulation scheme, the channel, and the receiver structure.

2.1.1 Modulation

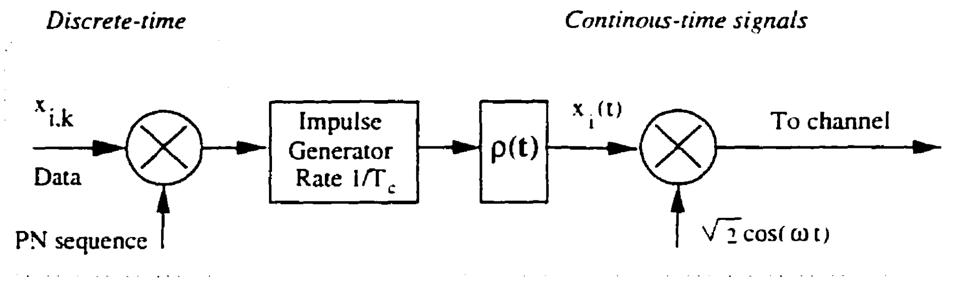


Figure 2.1: Model of a DS-CDMA modulator.

Figure 2.1 illustrates the model of a DS-CDMA modulator for user i. The data is assumed to be in bipolar form: the data symbol of user i at time index k is denoted by $x_{i,k}$ ($x_{i,k} \in \{-1, 1\}$). The data symbols are independent for different users, but not for different time indices of the same user, allowing for channel coding.

The sequence $c_{i,k}$ $(-\infty < k < \infty)$ models the PN spreading sequence of user j. The symbols $c_{i,k} \in \{-1, 1\}$ are two-valued random variables generated independently with equal probability. Hence $c_{i,k}$ and $c_{j,m}$ are independent unless $i = j$ and $k = m$.

$$E[c_{i,k}\, c_{j,m}] = \begin{cases} 1 & i = j \text{ and } m = k \\ 0 & \text{Otherwise} \end{cases} \tag{2.1}$$

In practice, the sequence $c_{i,k}$ $(-\infty < k < \infty)$ is a pseudo-random sequence with statistical properties well approximated by the above model. Every T_c seconds, where T_c is the duration of a chip symbol, the impulse generator outputs an impulse with the polarity determined by the state of its input (which can be ± 1). The chip pulse $\rho(t)$ has an arbitrary shape, energy E_c, and Fourier transform $F_\rho(f)$. Hence,

$$\int \rho^2(t)\,dt \;=\; \int |F_\rho(f)|^2\,df \;=\; E_c \tag{2.2}$$

We use the symbol $\int$ to denote the infinite integral $\int_{-\infty}^{\infty}$. The spreading signal of user i, $c_i^{(k)}(t)$, during the kth data time interval $[klT_c, (k+1)lT_c]$, is not necessarily time-limited to that interval, depending on whether $\rho(t)$ itself is time-limited to $[0, T_c]$. The chip rate is therefore $1/T_c$. Each user j transmits a signal $x_j(t)\sqrt{2}\cos(\omega t)$, where

$$x_j(t) \;=\; \sum_{k=-\infty}^{\infty} x_{j,k}\, c_j^{(k)}(t) \tag{2.3}$$

and
$$c_j^{(k)}(t) \;=\; \sum_{n=0}^{l-1} c_{j,n+kl}\, \rho(t - nT_c - klT_c) \tag{2.4}$$

In the absence of channel coding, the parameter l is the number of chips per data symbol, which is often taken as the definition of the processing gain in a spread-spectrum system. With channel coding, l is the number of chips per code symbol.

From the above considerations, and from Figure 2.1 the modulation scheme is a BPSK modulation of the spread data, followed by some waveform shaping.

2.1.2 The Channel

There are two sources of interference in the channel, as illustrated in Figure 2.2. The first one is due to thermal noise or other sources of zero-mean additive white Gaussian noise. The additive zero-mean white Gaussian noise $\eta(t)$ has an autocorrelation function $E[\eta(t)\,\eta(\tau)] = \frac{N_0}{2}\delta(t-\tau)$. The second source of interference is due to the other active users, and we will refer to it as the multiple-access interference (MAI). We let M denote the total number of active users in the channel at a given time. Note that by active users we mean any users whose signals are received with non-negligible power at the receiver. In particular, in a cellular environment, this may include some users from neighboring cells. We assume that the channel does not introduce any other distortion such as filtering or non-linearities. In other words, this

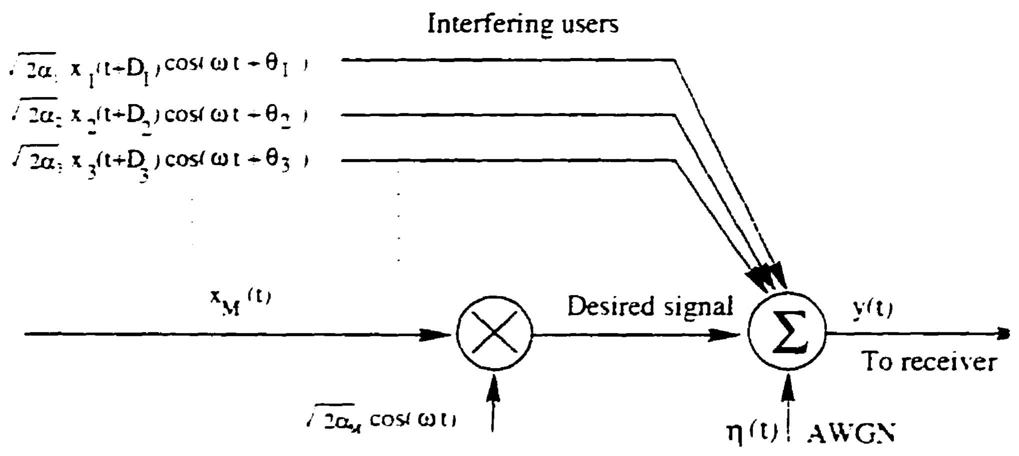

Interfering users
$\sqrt{2\alpha_1}\, x_1(t+D_1)\cos(\omega t+\theta_1)$
$\sqrt{2\alpha_2}\, x_2(t+D_2)\cos(\omega t+\theta_2)$
$\sqrt{2\alpha_3}\, x_3(t+D_3)\cos(\omega t+\theta_3)$
$x_M(t)$
Desired signal
$y(t)$
To receiver
$\sqrt{2\alpha_M}\cos(\omega t)$
$\eta(t)$ AWGN

Figure 2.2: Model of the multi-user AWGN channel.

means that the channel is considered ideal over the bandwidth occupied by the users. and does not introduce further intersymbol interference (ISI). Thus. throughout this work. we disregard any intersymbol interference. assuming that the chip pulse satisfies the Nyquist criterion for zero ISI.

At the channel output. each user signal will suffer a delay. and the corresponding carrier a phase shift. The assumption of a completely asynchronous uplink implies that the delays D_j are i.i.d. uniformly distributed on $[0, T_c]$ (they could be called advances). and that the phase shifts θ_j are i.i.d. uniformly distributed on $[0, 2\pi]$. The received signal at the channel output is then

$$y(t) \;=\; \sum_{j=1}^{M} \sqrt{\alpha_j}\, x_j(t-D_j)\, \sqrt{2}\cos(\omega t-\theta_j) \;+\; \eta(t) \qquad (2.5)$$

where α_j is a dimensionless gain factor which allows for different received power levels. We further assume that these gain factors are constant over the interval of observation. Without loss of generality. if i is the index of the desired user. we can set $\alpha_i = 1$ and $D_i = \theta_i = 0$. The model used insofar is very similar to that proposed in [12]. except that the carrier is not quadriphase modulated and that non-equal received user power levels are allowed.

Due to the multi-user interference. the channel is not memoryless. since a chip of another user interferes in general with many successive chips of the desired user. Hence the total interference is not white. strictly speaking. but interleaving can be used to destroy the channel memory.

2.1.3 Demodulation: Statistics of the Decision Variable

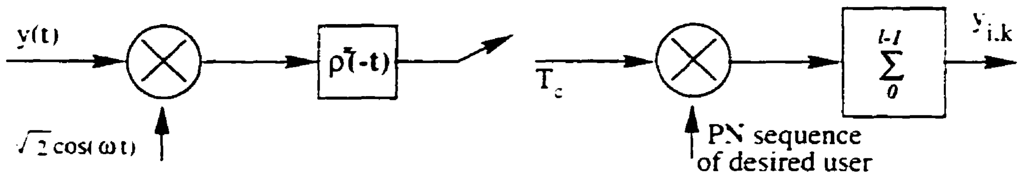


Figure 2.3: Model of a single-user receiver.

Demodulation is performed by a single-user receiver, as sketched in Figure 2.3. This is easily recognized as the optimal receiver for a point to point binary communications system over an AWGN channel. Indeed, such a receiver completely ignores the potential information contained in the interference which it treats as noise. Let $y_{j,k}$ denote the decision variable (output of the matched filter) of user j, at time index k. In order to simplify the notation, we will assume that the desired user is user M, and drop the subscript M from corresponding variables. The indices $1 \ldots M-1$ refer then to the interfering users. In particular, we have $y_k \equiv y_{M,k}$, $c^k(t) \equiv c_M^k(t)$, $x_k \equiv x_{M,k}$, $\alpha_M \equiv 1$, and $D_M = \theta_M = 0$. The expression for the decision variable is given by

$$y_k = \sum_{l\ chips} \int y(t)\sqrt{2}\cos\omega t\, c^k(t)\, dt \qquad (2.6)$$

where the sum is over the l chips per bit of the k-th bit interval, and the chip waveform shape $p(t)$ is included in the expression of $c^k(t)$. . The integral ranges from $-\infty to \infty$ (in practice, it ranges over the duration of a chip pulse). A straightforward computation yields

$$y_k = \int \left(\sum_{j=1}^{M} x_j(t - D_j)\cos\theta_j - \sqrt{2}\eta(t)\cos\omega t \right) c^k(t)\, dt \qquad (2.7)$$

where we used the standard assumption that the carrier frequency is much larger than the signal bandwidth, thereby neglecting the double frequency term.

The computation of the mean and variance of the y_k given x_k is a straightforward albeit lengthy calculation, which is detailed in section A.1 of Appendix A. We repeat here the conclusions of the computation.

$$E[y_k|x_k] \;=\; lE_c\, x_k \;=\; E_b\, x_k \tag{2.8}$$

$$\text{Var}[y_k|x_k] \;=\; \frac{lE_c}{2}(N_o + E_c\chi_\rho\gamma(M)) \tag{2.9}$$

$$\text{where}\quad \gamma(M) \equiv \sum_{j=1}^{M-1}\alpha_j \qquad \text{and} \qquad \chi_\rho \equiv \frac{1}{T_c}\frac{\int |F_\rho(f)|^4 df}{E_c^2} \tag{2.10}$$

Also, $E_b = lE_c$ is the energy per bit of the desired user when no channel coding is used.

The above result corroborates the findings of [12] when we assume a perfect power controlled environment with no inter-cell interference, and set all the received signal levels to be equal.

One of the most important outcomes of the above analysis is the revelation of the factor χ_ρ in the MAI contribution to the total interference. We refer to χ_ρ as the *pulse shape factor*, as it depends only on the shape of the chip waveform. The pulse shape factor is a dimensionless quantity which measures the other-user interference rejection capabilities of the pulse, since the product $\chi_\rho\gamma(M)$ is the other-user interference perceived by the desired user.

The signal to total interference ratio (STIR) can be written as

$$\text{STIR} \;=\; \frac{E^2[y_k|x_k]}{\text{Var}[y_k|x_k]} \;=\; \frac{2}{\frac{N_0}{E_b} + \frac{\chi_\rho}{l}\gamma(M)} \tag{2.11}$$

2.2 Pulse Shaping in CDMA

In this section, we analyze the effects of pulse shaping on the system performance. We restrict ourselves to chip pulses which satisfy the first Nyquist criterion for zero intersymbol interference, and derive a figure of merit which measures the MAI rejection capabilities of pulse shapes, taking into account their bandwidth. It turns out that several of pulses achieve a figure of merit extremely close to a theoretically computed bound on the best achievable figure of merit.

2.2.1 Computation of Some Pulse Shape Factors

In this section, we give the shape factor χ_ρ of a number of chip waveforms, some of which are sketched in Figure 2.4. Note that the pulse shapes of Figure 2.4 are all time-limited to one chip interval and that they thereby generate no intersymbol interference. Moreover, all chip waveforms examined in this section are assumed to satisfy (2.2), which means that they have energy E_c.

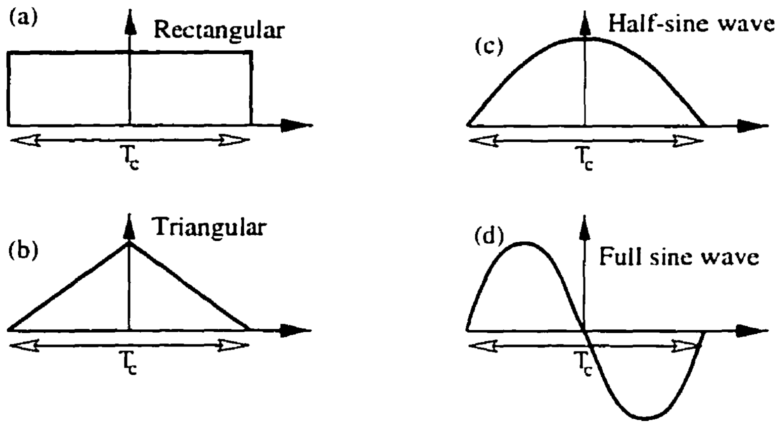


Figure 2.4: A few time-limited pulse shapes.

The pulse shape factor is computed by evaluating (2.10) in the time or frequency domain, either analytically, or with arbitrarily good accuracy by numerical integration. Table 2.1 lists the value of χ_ρ for the time-limited pulses that we consider. In the list of Table 2.1, those pulses that are not naturally time-limited are truncated to the interval $[-T_c/2, T_c/2]$.

Pulse Shape	χ_ρ	Pulse Shape	χ_ρ
Rectangular Fig. 2.4a	2/3	$\sin^3\left(\frac{2\pi t}{T_c}\right)$	0.31
Triangular Fig. 2.4b	$151/280 \approx 0.54$	$\sin^4\left(\frac{2\pi t}{T_c}\right)$	0.28
$\sin\left(\frac{\pi t}{T_c}\right)$ Fig. 2.4c	$1/3 + 5/2\pi^2 \approx 0.58$	Gaussian $e^{-a^2(t-\frac{T_c}{2})^2}$ $a = \frac{7.4}{T_c}$	0.59
$\sin\left(\frac{2\pi t}{T_c}\right)$ Fig. 2.4d	0.39	Gaussian $e^{-a^2(t-\frac{T_c}{2})^2}$ $a = \frac{25}{T_c}$	0.35
		Gaussian $e^{-a^2(t-\frac{T_c}{2})^2}$ $a = \frac{50}{T_c}$	0.25

Table 2.1: The value of χ_ρ for some time-limited pulse shapes.

We also consider non time-limited pulse shapes, in particular the family of square root raised cosine Nyquist pulses. These pulses are strictly bandlimited and their autocorrelation function is given by [13, pp. 534–536]

$$R_\rho(t) \quad = \quad \sqrt{E_c}\,\mathrm{sinc}(t/T_c)\,\frac{\cos(\pi\delta t/T)}{1-4\delta t^2/T^2} \tag{2.12}$$

where δ is the *roll-off factor* ($0 \leq \delta \leq 1$) which specifies the "excess bandwidth" of the pulse (for a sinc function, $\delta = 0$). It is shown that in Appendix A that the shape factor of square root raised cosine pulses is $\chi_\rho = 1 - \delta/4$.

Hence, as confirmed by Table 2.1, the sharper the pulse, the smaller is the value of χ_ρ. This is an expected behavior, since narrower pulses will overlap less, on average (given our assumption of i.i.d. uniformly distributed delays). However the narrower the pulse, the greater is its bandwidth. Indeed, chip pulses of negligible duration but of finite energy would overlap with a very small probability, yielding a shape factor of nearly zero, but would have an essentially infinite bandwidth. Hence for a meaningful comparison of chip pulses, we must compare pulses with the same bandwidth.

2.2.2 Chip Pulse Design Criterion

As discussed above, the pulse shape factor χ_ρ has an important effect on the STIR (2.11) and more generally on the performance of the system, as it multiplies directly the power of the other-user interference. Indeed, when (2.11) is the criterion of interest, a reduction in the pulse shape factor is translated in a proportional increase in the system capacity.

The pulse shape factor bears a special relation to the ambiguity function used in radar theory, and in particular to the time resolution constant [14, 15]

$$T_\rho \quad \equiv \quad \frac{1}{E_c^2}\int |F_\rho(f)|^4 df \tag{2.13}$$

In fact, χ_ρ is a normalized expression of T_ρ, whose physical interpretation is a measure of the spread of energy of the matched filter output [15, pp. 341]. In radar theory, the time resolution constant is a useful pulse design criterion in cases where the Doppler shift is negligible (stationary or slowly moving clutter) [15, ch. 10]. The rationale for this is that, for good clutter rejection, a time-shifted pulse (with respect to the information conveying pulse) should contribute as little as possible to the energy of the receiver matched filter output. In asynchronous CDMA, time-shifted chip pulses are due to the other users, and therefore the same rationale applies.

In [15, pp. 341-345], it is recognized that radar pulse design, based on the normalized time-resolution function (χ_ρ in our case), depends on bandwidth considerations. Let $R_b \equiv \frac{1}{lT_c}$ be the bit rate of the desired user. Then (2.11) can be written as

$$\text{STIR} = \frac{2}{\frac{N_0}{E_b} + R_b T_\rho \gamma(M)} \tag{2.14}$$

From (2.14), it is seen that $1/R_bT_\rho = lT_c/T_\rho = l/\chi_\rho$ gives a natural definition of the effective processing gain in a CDMA system. Hence the effective processing gain does not only depend on the spreading factor, but also on the chip shape. Furthermore, since the crosstalk rejection capabilities of CDMA are determined by the time resolution constant, $1/T_\rho$ is also the natural definition of bandwidth for CDMA. The argument is similar to that of [16], in which it is argued that, for a spread spectrum system with tone jamming in the band center, the noise equivalent bandwidth is the natural definition of bandwidth. However, the bandwidth definition does not depend on its pertinence to the particular application but appears as an external design constraint imposed by the necessity to limit the interference in neighboring channels, and is usually determined by regulatory agencies. As we observed earlier, the lower is T_ρ (or χ_ρ), the larger is the pulse bandwidth. Chip shaping should therefore aim at finding the best compromise between the MAI rejection capability and the bandwidth efficiency of the chip pulse.

As shown in [17], when comparing pulse shapes, the objective is to minimize the time-bandwidth product $T_\rho B_\rho$. We will refer to this product as the figure of merit or FOM of the pulse. Naturally, the actual value of the FOM of a pulse depends on the definition of bandwidth, and hence the design of chip waveforms requires the knowledge (or selection) of the bandwidth definition. From now on, we will restrict ourselves to the energy containment bandwidth which is defined as the frequency band around zero containing a fixed fraction μ of the energy of the pulse, as illustrated in Figure 2.5. Although μ can be any number in $(0, 1]$, usually, $\mu \geq 80\%$.

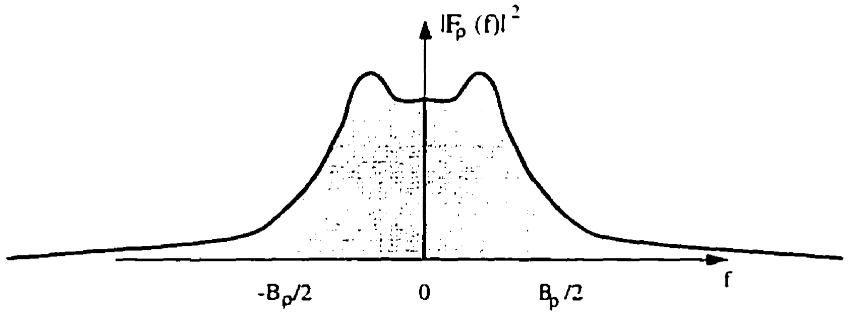


Figure 2.5: Energy containment bandwidth: the shaded region contributes μE_c to the total pulse energy E_c.

From the definition of bandwidth, we have

$$\int_{-B_p/2}^{B_p/2} |F_p(f)|^2 df \;=\; \mu E_c \tag{2.15}$$

where E_c is the energy of the pulse. Recall that

$$\begin{aligned} T_p &\equiv \frac{\int |F_p(f)|^4 df}{E_c^2} \\ &= \frac{1}{E_c^2}\left\{\int_{-B_p/2}^{B_p/2} |F_p(f)|^4 df + \int_{-\infty}^{-B_p/2} |F_p(f)|^4 df + \int_{B_p/2}^{\infty} |F_p(f)|^4 df\right\} \end{aligned} \tag{2.16}$$

On the other hand, we have

$$\int_{-B_p/2}^{B_p/2} \left(|F_p(f)|^2 - \frac{\mu E_c}{B_p}\right)^2 df \geq 0 \tag{2.17}$$

$$\Leftrightarrow \quad \int_{-B_p/2}^{B_p/2} |F_p(f)|^4 df - 2\frac{\mu E_c}{B_p}\mu E_c + B_p\left(\frac{\mu E_c}{B_p}\right)^2 \geq 0 \tag{2.18}$$

$$\Leftrightarrow \quad \int_{-B_p/2}^{B_p/2} |F_p(f)|^4 df \geq \frac{\mu^2 E_c^2}{B_p} \tag{2.19}$$

Using this in (2.16) yields

$$T_p \;>\; \frac{\mu^2}{B_p} \qquad \Leftrightarrow \; B_p T_p \;>\; \mu^2 \tag{2.20}$$

Hence (2.20) is a lower bound on the best possible time-bandwidth product for pulse shapes (when the bandwidth definition is a $\mu\%$ energy containment bandwidth). Since we neglected the last two integrals in (2.16) which are strictly positive unless $\mu = 1$, (2.20) is a strict lower bound on the FOM of a pulse for $\mu < 1$. In the case $\mu = 1$, the lower bound can be reached (for example, by a sinc function).

2.2.3 FOM of some pulse shapes

Pulse Shape	T_ρ/T_c	$B_\rho T_c$	$T_\rho B_\rho$	Pulse Shape	T_ρ/T_c	$B_\rho T_c$	$T_\rho B_\rho$
Rectangular	0.667	4.15	2.77	$e^{-a(t-\frac{T_c}{2})^2} \quad a = 25/T_c$	0.354	3.12	1.11
Triangular	0.539	2.00	1.08	$e^{-a(t-\frac{T_c}{2})^2} \quad a = 7.4/T_c$	0.598	1.80	1.07
$\sin\left(\frac{\pi t}{T_c}\right)$	0.587	1.83	1.07	$\text{sinc}\, a\left(t - \frac{T_c}{2}\right) \quad a = \frac{9}{T_c}$	0.112	8.32	0.93
$\sin\left(\frac{2\pi t}{T_c}\right)$	0.397	3.00	1.19	$\text{sinc}\, a\left(t-\frac{T_c}{2}\right) * \text{kaiser}(b)$ $a=11.5/T_c,\, b=2.65$	0.0876	10.58	0.926
$\sin^3\left(\frac{2\pi t}{T_c}\right)$	0.309	5.66	1.75	IS-95 Uplink Chip Shape	1.008	0.938	0.946

Table 2.2: Comparison between time-limited pulses (95% energy containment bandwidth).

Table 2.2 compares several time-limited pulse shapes using the FOM criterion, and an energy containment bandwidth with $\mu = 95\%$. The second column in each table gives the pulse shape factor, the third column shows the normalized 95% bandwidth and the last column gives the product which is the FOM. All pulse shapes of Table 2.2 but the IS-95 uplink pulse are time-limited to one chip interval: those pulses that are not naturally time-limited are truncated to the interval $[-T_c/2, T_c/2]$. The reader is referred to [17] for additional information.

Among all the pulses of Table 2.2, it is the last one (a shaped sinc function) that achieves the best FOM. For a 95% containment bandwidth, any FOM has to be greater than $\mu^2 = 0.9205$. Hence we have found pulse shapes that come very close to that value. Indeed, even if the performance of the CDMA system (or equivalently the capacity) were determined solely by the average STIR perceived by each active user, and assuming that the thermal noise is negligible compared to the MAI (interference-limited system), then the improvement in capacity (or reduction in interference) cannot be larger than the ratio $0.926/0.9205 \lesssim 1.006$ (this is just an upper bound on the ratio of the two STIR values). Therefore no other pulse shape can improve the capacity of a system using the shaped sinc function of Table 2.2 by more than 0.6%. For all practical purposes, we can consider the shaped sinc function to be an optimal pulse shape.

Given our criterion and a 95% energy containment definition of bandwidth, the chip pulse shape used in the IS-95 standard is also optimal for all practical purposes. Note that although the IS-95 uplink pulse does not satisfy, strictly speaking, the Nyquist criterion for zero ISI, the amount of ISI introduced is negligible, and we will ignore it.

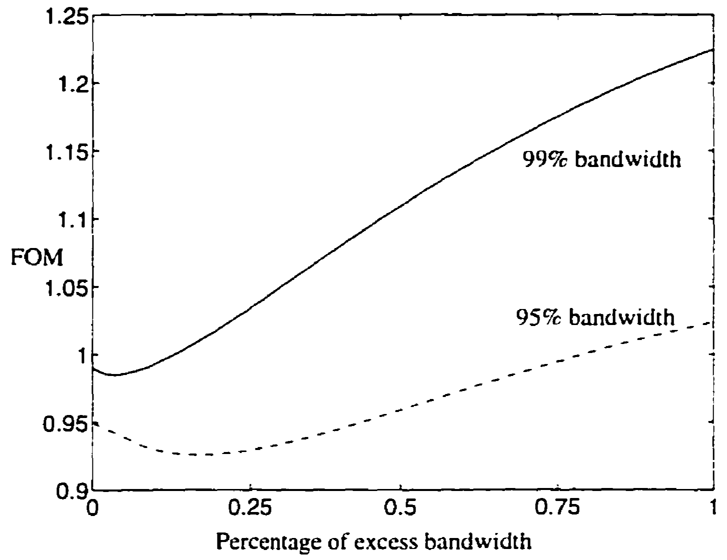


Figure 2.6: FOM of square-root raised cosine pulses (95% and 99% energy containment bandwidth.

We also investigate the FOM of square root raised cosine pulses. As shown in Appendix A, the energy containment bandwidth of square root raised cosine pulses can be found easily by solving numerically a single-variable equation (A.28). Moreover the pulse shape factor is shown to be $\chi_p = 1 - \delta/4$, where δ is the percentage of excess bandwidth. Hence one can compute the FOM of square root raised cosine pulses as a function of the percentage of excess bandwidth. The corresponding plot of the FOM versus δ is shown in Figure 2.6 for both the 95% and 99% energy containment bandwidths.

Figure 2.6 shows that the more stringent is the bandwidth definition, the closer does the optimal pulse converge to the sinc function. This is exactly what we expect since when $\mu = 100\%$ the sinc function meets the bound on the best achievable FOM, a conclusion reached in [12]. Moreover, we can achieve quasi-optimal pulse shaping

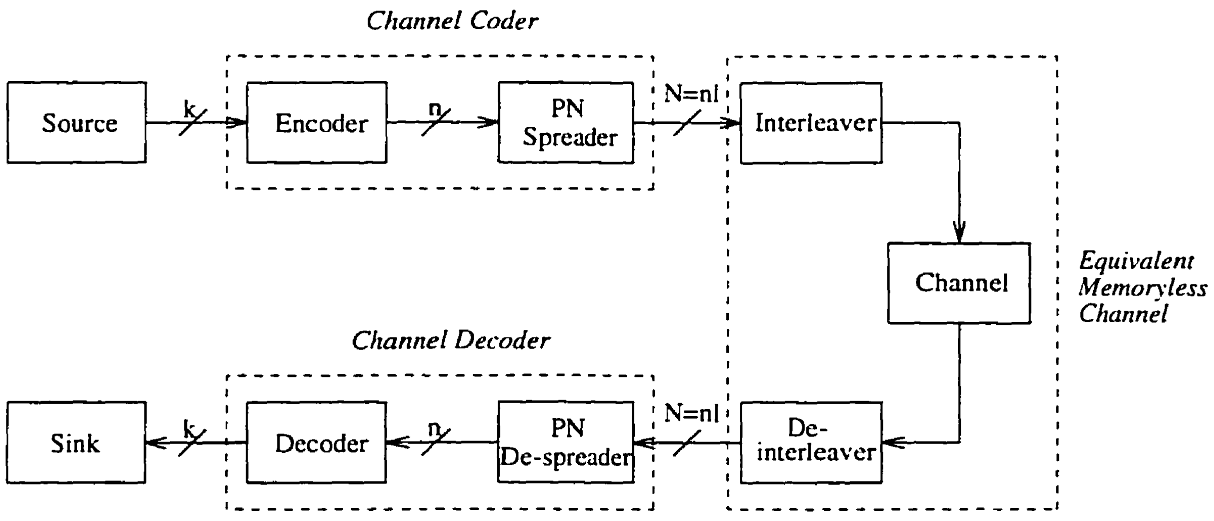


Figure 2.7: Block diagram of a block-coded DS-CDMA system.

with both time-limited and band-limited pulses.

Table 2.2 and Figure 2.6 show the potential improvement that can be obtained from careful chip pulse design. For example, for a 95% energy containment bandwidth, the improvement in FOM from the rectangular pulse shape to the shaped sinc pulses is of a factor of 3. When the average STIR is the performance measure of interest, and when the system is interference-limited (the thermal noise is much smaller than the MAI), this corresponds to a 300% improvement in the system capacity. Granted that such a prediction could be optimistic since the average STIR is only a crude measurement of the system performance, the potential improvement is nevertheless substantial, and reveals the particular importance of chip shaping in asynchronous DS-CDMA.

2.3 Channel Coding and Bandwidth Spreading

In this section, we show that the performance of the asynchronous DS-CDMA system under consideration is optimized when all the bandwidth spreading is accomplished through error control coding.

We consider an (n, k) block-coded asynchronous CDMA system, with l chips per code symbol. A model for such a system is shown in Figure 2.7. As discussed in section 2.1.2, chip interleaving ensures an equivalent memoryless channel. Referring to Figure 2.7, let $R_s \equiv k/nl$ and $N \equiv nl$. For each sequence of k data symbols, we transmit nl chips. Hence $1/R_s = nl/k$ is the bandwidth expansion factor. Note that this does not preclude scrambling, i.e. the use of non-spreading PN sequences, on top of the spreading scheme, in order to differentiate between users, an approach used in [12]. For a given code size 2^k (k is then fixed), and a fixed total bandwidth expansion R_s, we would like to find the values of the parameters n and l (the product $nl = N$ being fixed) which optimize the performance of the system.

Since a large part of this section was published in [17], we will often refer to that publication to shorten our discussions. For example, a concatenated coding approach can be used to show that the best solution is $n = N$ and $l = 1$ [17]. We will focus here on a random coding approach.

2.3.1 Random Coding Viewpoint

A commonly used approximation, based on central limit arguments, is to assume that the MAI interference is a Gaussian random process (see for example [11, 20, 21, 22]). The Gaussian approximation has been shown to be over-optimistic when the number of users is small, the bandwidth is large, and the system is interference-limited [23, 24]. Such a situation corresponds to a low probability of error regime of operation. In CDMA however, system performance is often measured in terms of capacity, e.g. the maximum number of users for a given outage probability. Acceptable outage probabilities (worst case error probabilities), when the system is saturated, are relatively high (of the order of 10^{-3}). In such cases, the results of [23, 24] indicate that the Gaussian approximation yields results that are reliable, and therefore can be used.

Consider the system depicted in Figure 2.7, with fixed total spreading rate R_s and fixed $N = nl$. Given (2.11), the single-user average probability of error, over all (n, k)

block codes. is upper bounded by (see for example [13. pp. 365-370])

$$\overline{P_e} < 2^{k-1}\left[\frac{1}{2}\left(1+\exp\left(\frac{-l}{\frac{N_0}{E_c}+\chi_\rho\gamma(M)}\right)\right)\right]^n \tag{2.21}$$

For a given code size 2^k. and a fixed total bandwidth expansion (R_s is fixed). we would like to find the values of the parameters n and l (the product $nl = N$ being fixed) which minimize the random coding bound of (2.21). We rewrite (2.21) as

$$\overline{P_e} < \frac{1}{2}\, 2^{-N(R_o(l)-R_s)} \tag{2.22}$$

$$\text{where}\quad R_o(l) \equiv \frac{1}{l}\left[1-\log_2\left(1+e^{-\alpha l}\right)\right] \quad\text{and}\quad \alpha \equiv \frac{1}{\frac{N_0}{E_c}+\chi_\rho\gamma(M)} \tag{2.23}$$

From (2.22), the quantity $R_o(l)$ is the cut-off rate of the multi-user channel perceived by the desired user. One can also interpret $1/R_o(l)$ as the minimum bandwidth expansion required to force the random coding bound to zero by increasing N. Since N is fixed in (2.22), the goal is to maximize $R_o(l)$ on the interval $1 \leq l \leq N$.

As shown in [17], the cut-off rate is a monotonously decreasing function of l. and is thus maximized (and the random coding bound minimized) when $l = 1$ and $n = N$. This is illustrated in Figure 2.8. where we plot the cut-off rate for different values of l for an M-user system with a total spreading factor of 64. $\gamma(M) = 49.0$ and square pulse shaping. This corresponds to a 50-user system if the average of the received power levels of the interfering users equals the received power level of the desired user. A stronger (more restrictive) condition would be that all received power levels are equal. requiring perfect power control. but the weaker condition suffices.

We can thus improve the system performance by using coding instead of PN sequences to spread the signal bandwidth. The improvement gets better as the code size increases, provided that the total spreading rate R_s is lower than the cut-off rate $R_o(l)$. as suggested by (2.22). We illustrate that argument in Figure 2.9. where we plot the probability of error for a PN sequence spread system with the Gaussian approximation. which is given by

$$P_e = \frac{1}{2}\operatorname{erfc}\left(\sqrt{\frac{l}{\frac{N_0}{E_c}+\chi_\rho\gamma(M)}}\right) \tag{2.24}$$

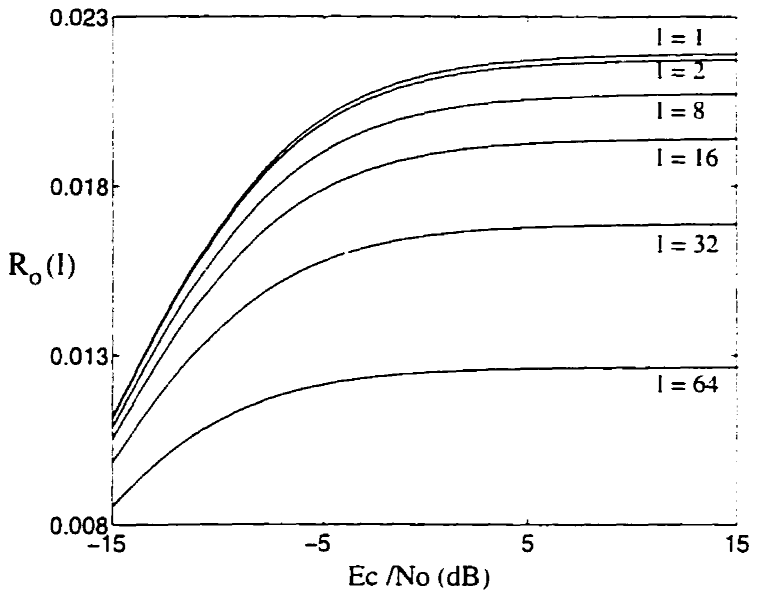


Figure 2.8: Cut-off rate versus channel E_c/N_o for various values of l, and $\gamma(M) = 49.0$

We also show the corresponding random coding bound for systems using all the bandwidth expansion for coding. These plots are shown in function of $E_b/N_o = E_c/(R_s.N_o)$. The system considered in that example has $\gamma(M) = 31.0$, a rectangular pulse shape ($\chi_\rho = 2/3$), and a bandwidth expansion factor of 64, which implies that $E_b/N_o = E_c/N_o + 18$ in dB. The E_b/N_o interval considered is such that the channel cut-off rate is greater than the total spreading rate R_s. The performance of the system improves with increasing the code size. However, with these parameters, we need $k \geq 6$ to have better performance than PN spreading alone. For $k = 10$, the bound on the average probability of error over all (150,6) block coded systems is already well below the probability of error of a PN sequence spread system.

Consider now increasing the value of $\gamma(M)$. Figure 2.10 shows the exact probability of error (based on the Gaussian assumption) and the corresponding random coding bounds for $\gamma(M) = 63.0$. If perfect power control is available, we then have 64 users. Notice that for $E_b/N_o \approx 12$ dB, the channel cut-off rate equals the spreading rate $R_s = 1/64$, as all random coding bounds intersect at that point. For $k = 50$, the average (1250,50) code achieves a lower probability of error than PN sequences. When $k = 100$, the value of the random coding bound at high E_b/N_o is below 3.10^{-4}.

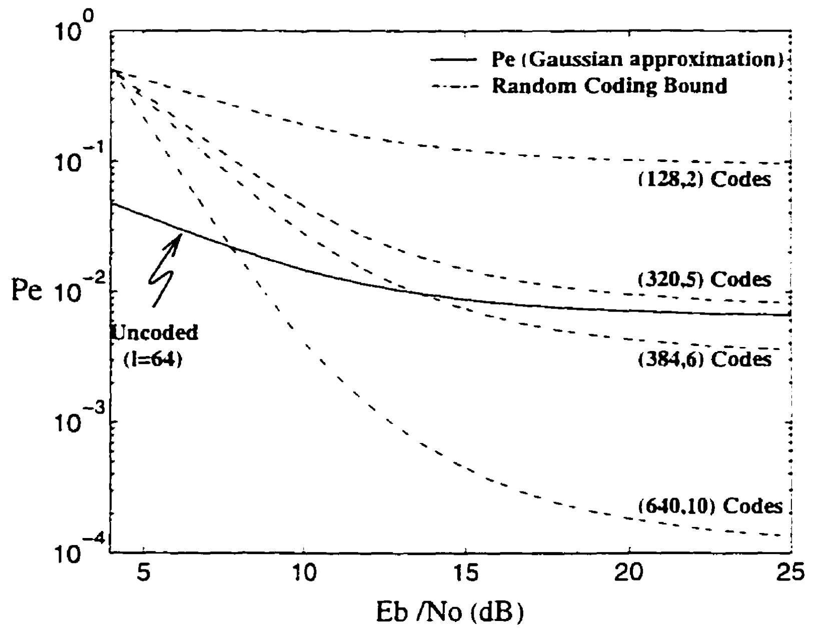


Figure 2.9: Probability of error and bounds (rectangular pulse) for $\gamma(M) = 31.0$ and $R_s = 1/64$

This shows that it is possible to accommodate a number of users equal to $1/R_s$ at an acceptable probability of error, provided $R_s < R_o$ and the code size is large enough.

In this section we have shown that the system performance is optimized when all the bandwidth expansion is accomplished by an error control code. In practice, such a strategy is limited by synchronization issues, which is difficult to achieve and maintain at very low values of E_s/N_0 (where E_s is the energy per coded symbol). Hence, in order to simplify the synchronization, it is sometimes necessary to provide a "spreading margin" (i.e. have the PN sequence accomplish part of the overall spreading). This approach is used in the uplink of the IS-95 standard, where the error control code has a rate of 1/32, and the PN sequence further spreads the signal by a factor of 4, for an overall spreading factor of 128.

2.3.2 The Chip Pulse Shape

The improvement due to coding can be further increased by chip pulse shaping. Consider two systems S_1 and S_2 using pulse shapes $\rho_1(t)$ and $\rho_2(t)$ respectively, with $\beta \equiv B_1/B_2 > 1$. For equal system bandwidths, S_2 uses a spreading rate R_{s_2} which

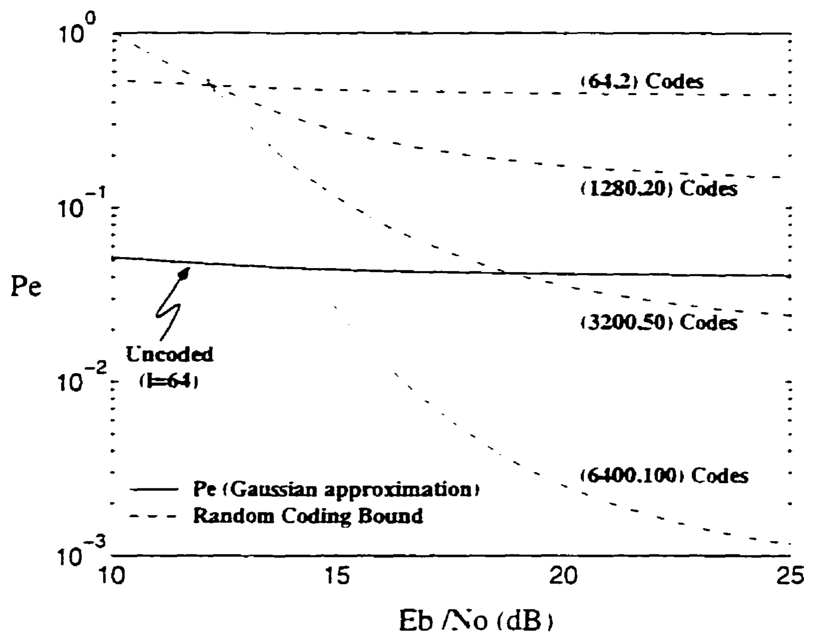


Figure 2.10: Probability of error and bounds (rectangular pulse) for $\gamma(M) = 63.0$ and $R_s = 1/64$

is lower by a factor of 3 than that of S_1. If both systems spread their bandwidth through error control coding of the same size, then $R_{s_1} = k/n$ and $R_{s_2} = k/3n$, showing that the available dimensions of S_2 are increased by the factor 3. With equal distribution of power levels in both systems, the random coding bounds are given by

$$\overline{P_{e,1}} < \frac{1}{2} 2^{-n(R_{o,1}-R_{s_1})} \tag{2.25}$$

where

$$R_{o,1} \equiv 1 - \log_2\left(1 - \exp\left(\frac{-1}{\frac{N_0}{E_c} - \chi_{\rho_1}\gamma(M)}\right)\right) \tag{2.26}$$

and E_c is the energy per user per dimension in S_1, and

$$\begin{aligned}\overline{P_{e,2}} &< \frac{1}{2} 2^{-3n(R_{o,2}-R_{s_2})}\\ &= \frac{1}{2} 2^{-n(3R_{o,2}-R_{s_1})}\end{aligned} \tag{2.27}$$

where

$$R_{o,2} \equiv 1 - \log_2\left(1 - \exp\left(\frac{-1}{\frac{3N_0}{E_c} - \chi_{\rho_2}\gamma(M)}\right)\right) \tag{2.28}$$

In order to maintain the same E_b/N_o in the two schemes, the energy per user per dimension of $\mathcal{S}_2$ must be lower than that of $\mathcal{S}_1$ by a factor of β. The ratio of the bound on $\overline{P_{e,2}}$ (2.27) to the bound on $\overline{P_{e,1}}$ (2.25) is equal to $2^{-n(\beta R_{o,2}-R_{o,1})}$, and is an indication of the relative performance of the two systems. It follows that $\mathcal{S}_2$ will outperform $\mathcal{S}_1$ whenever $\beta R_{o,2} - R_{o,1} > 0$. Note that this is equivalent to comparing the difference of cut-off rates per unit bandwidth. The value of $\beta R_{o,2} - R_{o,1}$ is significant since the ratio of the random coding bounds decreases exponentially with this quantity. For example, if $\beta R_{o,2} - R_{o,1} = 0.1$, then for $n = 30$ only, we have that the bound on the probability of error for $\mathcal{S}_1$ (2.25) is 8 times larger than that of $\mathcal{S}_2$ (2.27). As a rule of thumb, we can include as significant any region such that $|\beta R_{o,2} - R_{o,1}| > 3/n$ since this corresponds to a factor of 8 (almost one order of magnitude) in the bounds of (2.25) and (2.27).

In [17], it is shown analytically that given $\chi_{\rho_2} \leq \chi_{\rho_1}$ and $\beta \geq 1$, we have $\beta R_{o,2} - R_{o,1} > 0$. This is an expected result since $\rho_2(t)$ has then a lower pulse shape factor (reducing the other-user interference), and a lower bandwidth (allowing for a lower coding rate). In some sense, we win from both ends.

We further show in [17] that $\beta R_{o,2}$ is an increasing function of β, while it is clearly also a decreasing function of χ_{ρ_2}. Since $R_{o,1}$ does not depend on β nor on χ_{ρ_2}, it follows that $\beta R_{o,2} - R_{o,1}$ increases with β and decreases with χ_{ρ_2}. Therefore, if both $\chi_{\rho_2} < \chi_{\rho_1}$ and $\beta < 1$, it is not ensured anymore that $\mathcal{S}_2$ outperforms $\mathcal{S}_1$, since the reduction in MAI obtained by a lower pulse shape factor is offset by a smaller number of available dimensions for channel coding.

In [17], we compare a system with a rectangular pulse shape $\mathcal{S}_r$ and a system $\mathcal{S}_s$ with the truncated sinc pulse shape of Table 2.2 (the one with a normalized bandwidth of 8.32 and a pulse shape factor of 0.112), and show that $R_{o,\text{sinc}}/2.005 - R_{o,\text{rect}} > 0$ for $\gamma(M) > 2$, i.e. for all practical purposes. One can argue that this result is not surprising since the rectangular pulse shape has a FOM much larger than that of the truncated sinc pulse. This confirms that the FOM is a reasonable measure of the quality of a pulse shape.

As a second example, we compare a system $\mathcal{S}_{95}$ using the IS-95 uplink pulse shape

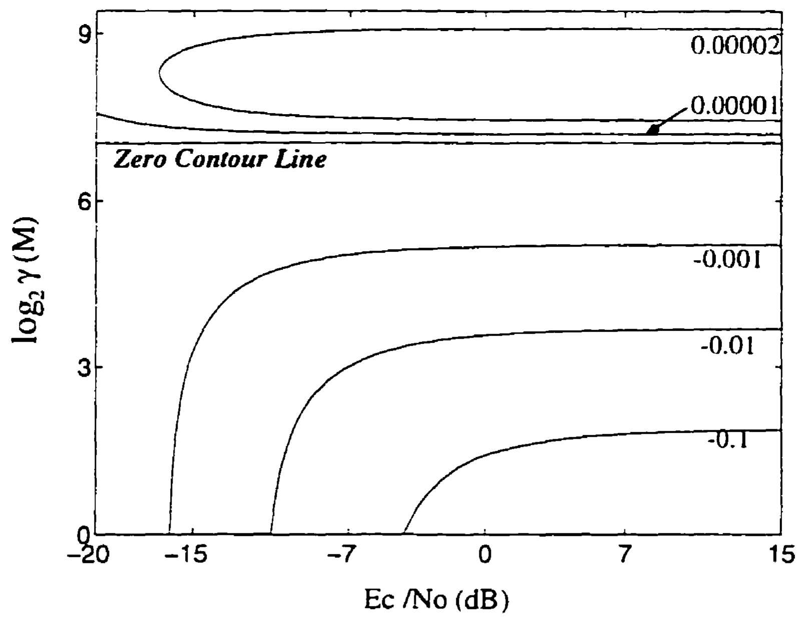


Figure 2.11: Contour plot of $R_{o,sinc}/11.28 - R_{o,\mathrm{IS\_95}}$ as a function of E_c/N_o and $\log_2 \gamma(M)$

of Table 2.2 and a system S_s using the shaped sinc pulse of Table 2.2 (last entry). This choice is motivated by the fact that both pulses have similar figure of merits, but while the shaped sinc pulse has a large bandwidth and a small time resolution constant, the IS-95 chip pulse has a small bandwidth and a comparatively large time resolution constant. We hope to gain some insights into the tradeoff between bandwidth expansion and interference rejection capabilities of a pulse, in the presence of error control coding, and check whether the FOM criterion is still meaningful. The comparison is performed again by showing in Figure 2.11 the contour plot of $R_{o,sinc}/11.28 - R_{o,95}$ where

$$R_{o,sinc} = 1 - \log_2\left(1 + \exp\left(\frac{-1}{\frac{N_0}{E_c} + 0.0876\gamma(M)}\right)\right) \tag{2.29}$$

$$R_{o,95} = 1 - \log_2\left(1 + \exp\left(\frac{-1}{\frac{N_0}{11.28E_c} + 1.008\gamma(M)}\right)\right) \tag{2.30}$$

since the ratio of the IS-95 chip pulse bandwidth to that of the shaped sinc pulse is 1/11.28. From the zero contour line in Figure 2.11, S_{95} performs uniformly better than S_s whenever $\log_2 \gamma(M) \lesssim 7$, or $\gamma(M) < 130$. On the other hand, for the range of values of E_c/N_o shown. $\mathcal{S}_s$ outperforms $\mathcal{S}_{95}$ provided $\gamma(M) \geq 130$. The zero contour line is in fact independent of E_c/N_o on the chosen range, defining a threshold value of $\gamma(M) \approx 130$ which delimits the regions where one system outperforms the other. The IS-95 chip shaped system can outperform significantly the system using the shaped sinc pulse only for very small values of the MAI, which even in a perfect power controlled environment corresponds to less than 10 users. This is determined by examining the region were $R_{o,sinc}/11.28 - R_{o,95} < -0.01$. On the other hand, for most of the region where $\gamma(M) \geq 130$, $|R_{o,sinc}/11.28 - R_{o,95}| < 0.00002$, which means that the potential advantage of system $\mathcal{S}_s$ is negligible since we would need extremely powerful error control coding to exploit that advantage (we would be looking at block codes of length greater than 200000, and require that they achieve the average-code performance of (2.21)).

Therefore, the IS-95 chip pulse shape and the windowed sinc pulse shape under consideration are comparable from the point of view of the resulting cut-off rates, at least when the MAI is not too small. This confirms that the FOM of a chip pulse is a good indication of its performance in an asynchronous DS-CDMA system, even when error control coding is considered, since both pulse shapes have comparable FOMs.

2.4 CDMA and Orthogonal Multiple Access

The previous sections of this chapter have revealed some critical issues in asynchronous DS-CDMA, and shown that chip shaping and error control coding can increase the system performance in a very significant way. However, being able to carefully design a CDMA system in order to optimize the performance does not justify the use of CDMA as a multiple accessing scheme. Although the debate about which of CDMA and TDMA is the better multiplexing scheme is not yet closed (and depends on a large number of parameters), we hope to shed some light onto this debate by comparing CDMA with TDMA on the basis of their cut-off rates.

Given M blocks of k data symbols to be transmitted (one block per user), how do we allocate n channel dimensions to the users? With orthogonal multiple access (OMA), which encompasses TDMA and FDMA, we allocate n/M separate dimensions to each user. Since each user is constrained to its own signal sub-space, this scheme avoids all cross-talk interference. In asynchronous DS-CDMA, all n dimensions are assigned to all M users. The price to pay is the presence of cross-talk interference, but we gain an increase in the number of dimensions per user. A careful mix of chip pulse shaping and channel coding can both decrease the crosstalk interference and exploit the available dimensions for coding.

A comparison of CDMA with binary OMA on the basis of the cut-off rate per unit bandwidth is provided in [17]. It is shown that CDMA with the windowed sinc pulse of Table 2.2 is equivalent to OMA with a roll-off parameter of about 85% (square-root raised cosine pulse, and 95% energy containment bandwidth). For a 99% energy containment bandwidth, the same CDMA system is equivalent to an OMA system with a roll-off parameter of about 60%. For smaller values of the roll-off parameter, OMA is superior to CDMA. It is understood however that this conclusion is for a single cell analysis. When the much smaller frequency reuse factor of CDMA is taken into account, its full potential is revealed.

2.5 Conclusion

This chapter considered CDMA systems with single user receivers. We have shown that coding and chip shaping are integral parts of a CDMA system, and that the time resolution constant of the chip waveform indicates its crosstalk rejection capabilities. The chip figure of merit (FOM) has been introduced as a quantitative measure of the "goodness" of a chip pulse shape, i.e. a *normalized* measure of the crosstalk rejection capabilities which takes the bandwidth of the chip waveform into account. We have also shown that, given the definition of bandwidth, it is possible to find many pulse shapes with practically optimal FOMs.

The conclusion that bandwidth spreading with error control coding is preferable to spreading with PN sequences was reached from a random coding argument. We emphasize that this does not exclude the use of non-spreading PN sequences for user differentiation purposes. Only a combination of coding and chip pulse shaping can increase the performance of binary DS-CDMA to be almost comparable to that of binary OMA, in a single-cell interference free environment. This work shows that with appropriate chip shaping, the CDMA performance is equal to that of orthogonal multiple access with binary signaling and raised cosine shaping of 85% excess bandwidth. The bandwidth definition for both is the 95% energy containment bandwidth. This figure drops to 60% when the 99% energy containment bandwidth is used.

The advantages of CDMA in an interference dominated environment are well known. This work shows that with error control coding and proper chip shaping, CDMA performance is comparable to that of binary OMA even in an interference free environment. Therefore if other natural attributes of CDMA (such as robustness to interference, asynchronous operation, soft capacity, soft handoffs, etc $\cdots$) are of interest, then not much of its error rate performance with respect to binary OMA is sacrificed even in the least favorable operation environment for CDMA. Hence, in cellular systems, the low frequency reuse factor required by cellular CDMA supports the many claims of superiority of CDMA over OMA [6].

Chapter 3

Construction of Very Low Rate Codes

We have just shown in the previous chapter that bandwidth expansion through channel coding improves the performance of a spread spectrum multiple-access (SSMA) system. The arguments we used showed the existence of good codes that can yield the promised performance improvement. In this chapter, we look at several families of very low rate codes and introduce a new such family based on the combination of block and convolutional codes.

3.1 Known Very Low Rate Codes

Some good very low rate codes can be constructed by combining convolutional and block codes, such as orthogonal convolutional codes [12, 25, 19], biorthogonal convolutional codes [26], superorthogonal codes [27] and the IS-95 uplink code. In this section, we present the above mentioned error control codes that will be used for benchmark comparisons with the new family of codes that will be introduced later.

3.1.1 Notational Conventions

Let $\underline{u} = (u_1, \cdots, u_n)$ and $\underline{v} = (v_1, \cdots, v_n)$ be two vectors of length n

- The component-wise product of $\underline{u}$ and $\underline{v}$ is the vector $\underline{u} \otimes \underline{v} = (u_1 v_1, \cdots, u_n v_n)$
- The dot product of $\underline{u}$ and $\underline{v}$ is the scalar $<\underline{u}|\underline{v}> = \sum_{i=1}^{n} u_i v_i$

- The norm of a vector is the scalar $||\underline{u}||$ such that $||\underline{u}||^2 = <\underline{u}|\underline{u}> = \sum_{i=1}^{n} u_i^2$
  In particular, the Euclidean distance between $\underline{u}$ and $\underline{v}$ is the scalar $||\underline{u}-\underline{v}||$ such that
  $$||\underline{u}-\underline{v}||^2 = \sum_{i=1}^{n}(u_i - v_i)^2 = ||\underline{u}||^2 + ||\underline{v}||^2 - 2 <\underline{u}|\underline{v}>$$
- The maximum component of a vector $\underline{u}$ is the scalar
  $$\mathcal{M}ax\ \underline{u} = \max_{i=1,\cdots,n} u_i, \quad \text{and} \arg \mathcal{M}ax\ \underline{u} = \left\{ j \middle| u_j = \max_{i=1,\cdots,n} u_i \right\}$$
- The absolute value of a vector $\underline{u}$ is the vector $|\underline{u}| = (|u_1|, \cdots, |u_n|)$
  A vector $\underline{u}$ is said to be bipolar if $|\underline{u}| = (1, \cdots, 1)$.
- The Hamming weight of a bipolar vector $\underline{u}$ is the number of its components which are equal to -1. and it is denoted by $w_{\mathrm{H}}(\underline{u})$. If $\underline{u}$ and $\underline{v}$ are two bipolar vectors, then
  $$<\underline{u}|\underline{v}> = \sum_{i=1}^{n} u_i v_i = n - 2\, w_{\mathrm{H}}(\underline{u}\otimes\underline{v}) \tag{3.1}$$
- The Hamming distance between two bipolar vectors $\underline{u}$ and $\underline{v}$ is $d_{\mathrm{H}}(\underline{u}, \underline{v}) = w_{\mathrm{H}}(\underline{u}\otimes\underline{v})$

3.1.2 Orthogonal and Biorthogonal Codes

Let the Hadamard matrix H_m be defined by the following recursive relation

$$H_m = \begin{bmatrix} H_{m-1} & H_{m-1} \\ H_{m-1} & -H_{m-1} \end{bmatrix} \qquad m = 1, 2, \cdots \qquad H_o = 1 \tag{3.2}$$

We define the code $\mathcal{O}_m$ whose codewords are the rows (or equivalently columns) of H_m. Hence $\mathcal{O}_m$ is a $(2^m, m)$ code. Furthermore, let $\underline{c}_1 = (c_{11}, \cdots, c_{12^m})$ and $\underline{c}_2 = (c_{21}, \cdots, c_{22^m})$ be two distinct codewords of $\mathcal{O}_m$. From (3.2), $\underline{c}_1$ and $\underline{c}_2$ must differ in exactly half their components. Therefore

$$<\underline{c}_1|\underline{c}_2> = \sum_{i=1}^{2^m} c_{1i}c_{2i} = 0 \quad d_{\mathrm{H}}(\underline{c}_1, \underline{c}_2) = 2^{m-1} \tag{3.3}$$

Therefore, the Hamming distance between any two codewords of $\mathcal{O}_m$ is equal to 2^{m-1}. It follows that $\mathcal{O}_m$ is a $(2^m, m)$ orthogonal code of minimum distance 2^{m-1}.

Let $\underline{r}$ be any real valued vector of length 2^m. We define the Hadamard transform of $\underline{r}$ as the 2^m-component vector

$$\mathcal{H}(\underline{r}) \equiv 2^{-m/2}\, \underline{r}\, H_m \tag{3.4}$$

Since the columns of H_m are the 2^m codewords of $\mathcal{O}_m$, the Hadamard transform of $\underline{r}$ consists of the 2^m correlations values $2^{-m/2} <\underline{r}|\underline{c}_i>$, $(i = 1, \cdots, 2^m)$ with each codeword of $\mathcal{O}_m$. Formally, $\mathcal{H}(\underline{r})_i = 2^{-m/2} <\underline{r}|\underline{c}_i>$. The maximum likelihood (ML) decision rule for $\mathcal{O}_m$ over an AWGN channel, assuming that each code symbol is mapped into a separate signal space dimension, is given by

$$\begin{aligned} \arg \min_{i=1,\cdots,2^m} \|\underline{r} - \underline{c}_i\| &= \arg \max_{i=1,\cdots,2^m} <\underline{r}|\underline{c}_i> = \arg \max_{i=1,\cdots,2^m} 2^{-m/2} <\underline{r}|\underline{c}_i> \\ &= \arg \mathcal{M}ax\ \mathcal{H}(\underline{r}) \end{aligned} \tag{3.5}$$

Therefore the ML decoding of $\mathcal{O}_m$ is equivalent to taking the Hadamard transform of the received vector and finding the index i of its largest component.

Consider now the code $R(1, m) = \mathcal{O}_m \cup \overline{\mathcal{O}_m}$, where $\overline{\mathcal{O}_m}$ is the set of codewords which are the complements of the codewords of $\mathcal{O}_m$. $R(1, m)$ is a linear biorthogonal $(2^m, m+1)$ code of minimum Hamming distance 2^{m-1}, referred to as the first order Reed-Muller (RM) code [18, chap. 14]. The ML decision rule for this code is

$$\arg \min_{i=1,\cdots,2^{m+1}} \|\underline{r} - \underline{c}_i\| = \begin{cases} j & \text{if } \mathcal{M}ax\ |\mathcal{H}(\underline{r})| = \mathcal{M}ax\ \mathcal{H}(\underline{r}) \\ j^c & \text{if } \mathcal{M}ax\ |\mathcal{H}(\underline{r})| = -\mathcal{M}ax\ \mathcal{H}(\underline{r}) \end{cases} \tag{3.6}$$

where $j \equiv \arg \mathcal{M}ax\ |\mathcal{H}(\underline{r})|$, and j^c is such that $\underline{c}_{j^c} = -c_j$. The last step takes advantage of the fact that $R(1, m) = \mathcal{O}_m \cup \mathcal{O}_m^c$ by computing only half the correlation values, and using $<\underline{r}|\underline{c}_i> = - <\underline{r}| - \underline{c}_i>$. Maximum likelihood decoding is then performed in the following way: we first obtain the Hadamard transform $\mathcal{H}(\underline{r})$ of the received vector. We then let j be the index of the largest component of $|\mathcal{H}(\underline{r})|$. If the value of this jth component of $\mathcal{H}(\underline{r})$ is positive, we decide in favor of $\underline{c}_j$, otherwise we decide in favor of $-\underline{c}_j$. Note that the bit error probability is minimized, for a fixed word error probability, if complement data words are mapped into complement codewords [28, pp. 212].

Cosets of $R(1, m)$ can also be decoded by using the Hadamard transform. Indeed, let $\underline{\lambda}$ be the coset leader of some $R(1, m)$ coset, i.e. we consider the coset code $C = \{\underline{\lambda} \otimes \underline{c}_i \ : \ \underline{c}_i \in R(1, m)\}$. The received vector is

$$\underline{r} \ = \ \underline{\lambda} \otimes \underline{c}_i + \underline{n} \tag{3.7}$$

where $\underline{n} = (n_1, \cdots, n_{2^m})$ is the noise vector and $\underline{\lambda}$ is in bipolar form. Note that

$$\begin{aligned} <\underline{r}|\underline{\lambda}\otimes\underline{c}_j> &= <\underline{\lambda}\otimes\underline{c}_i + \underline{n}|\underline{\lambda}\otimes\underline{c}_j> = \sum_{k=1}^{2^m}(\lambda_k c_{ik} + n_k)\lambda_k c_{jk} \\ &= \sum_{k=1}^{2^m}(c_{ik} + n_k\lambda_k)c_{jk} = <\underline{c}_i + \underline{n}\otimes\underline{\lambda}|\underline{c}_j> = <\underline{r}\otimes\underline{\lambda}|\underline{c}_j> \end{aligned} \tag{3.8}$$

Thus the maximum-likelihood decision rule is

$$\arg \max_{j=1,\cdots,2^{m+1}} <\underline{r}|\underline{\lambda}\otimes\underline{c}_j> = \arg \max_{j=1,\cdots,2^{m+1}} <\underline{r}\otimes\underline{\lambda}|\underline{c}_j> \tag{3.9}$$

Maximum likelihood decoding of a coset of $R(1, m)$ can be done in the same way (i.e. with the exact same operations) as for $R(1, m)$, provided the received vector is pre-multiplied by the coset leader. Refer to [18, ch. 13-14] for more on first-order Reed-Muller codes, and their decoding using the Hadamard transform.

3.1.3 The Fast Hadamard Transform

The first-order Reed-Muller code would not be so widely proposed as a building block of low rate codes if it were not for the existence of an efficient method to compute the Hadamard transform. Indeed, the Hadamard transform of a vector $\underline{v}$ is nothing more than the correlation values of $\underline{v}$ with all 2^m Hadamard codewords. This implies the computation of 2^m values, each of which takes $O(2^m)$ additions to compute, since the length of the vectors involved is 2^m. Therefore the computational complexity of a straightforward implementation of the Hadamard transform (for example that suggested by (3.4)) is $O(2^{2m})$.

The efficient method to compute the Hadamard transform is the Fast Hadamard Transform (FHT) also called Green's machine [29, pp. 28–35]. The FHT is analogous to the Fast Fourier Transform (FFT) method of computing the discrete-time Fourier Transform of a vector. The only difference between the two transforms is that the coefficient values of the FHT are ± 1. This implies that the FHT does not involve any multiplication, but only additions and subtractions (we ignore the normalizing factor $2^{-m/2}$ in (3.4)). We will not reproduce here the derivation of the FHT which can be obtained from the analogy of the Hadamard transform with the discrete-time Fourier transform, or from matrix partitioning techniques. The interested reader is referred to [18, section 14.4] and [30, section 6.3], the latter containing also further references to other derivations of the FHT.

3.1.4 Orthogonal, Biorthogonal, and Superorthogonal Convolutional (OBS) Codes

Orthogonal convolutional codes were introduced in [25], and discussed in [12] in the context of SSMA. The encoding structure is shown in Figure 3.1. The data is fed into a K-bit shift register, one bit at a time. The content of the shift register selects a codeword in the $(2^K, K)$ orthogonal code $\mathcal{O}_K$. For each data bit fed in the shift register, a 2^K-bit codeword is transmitted. Hence the rate of this code is $1/2^K$.

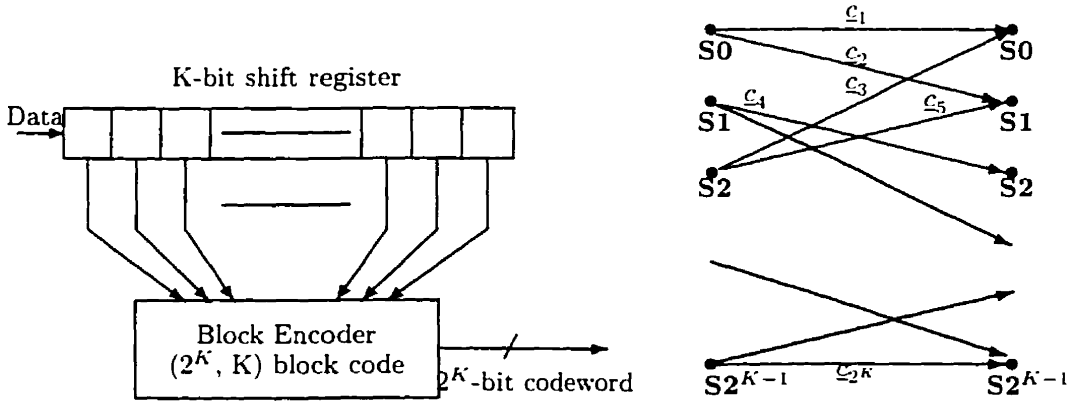


Figure 3.1: Encoder and trellis representation of a convolutional orthogonal code.

From the encoder structure, it is clear that any two non-equal data sequences generate at least K consecutive non-equal codewords. The Hamming distance between any two non-equal codewords is 2^{K-1}. Hence the free distance of the code is

$d_{\text{free}} = K2^{K-1}$, and its asymptotic coding gain is [32, pp. 243]

$$G_{\infty(\text{ortho})} = \frac{1}{2^K} \times K2^{K-1} = K/2 \tag{3.10}$$

The Viterbi algorithm (VA) provides an effective method for maximum likelihood decoding of orthogonal convolutional codes. The trellis representation of an orthogonal convolutional code (Figure 3.1) contains 2^{K-1} states with 2 branches per node. There are thus 2^K branches for each trellis stage. Each of these branches represents a 2^K-bit codeword belonging to $\mathcal{O}_K$. From the encoding structure, each of these 2^K codewords are different, since they are uniquely determined by a one-to-one map from the state transition to the orthogonal code $\mathcal{O}_K$. At each stage of the trellis, the 2^K branch metrics are the 2^K correlation values $<\underline{r}|\underline{c}_i>$, where $\underline{r}$ is the received vector and $\mathcal{O}_K = \{c_1, \cdots, c_{2^K}\}$. Hence a single Hadamard transform provides the 2^K branch metrics for each trellis stage. Figure 3.1 illustrates the one-to-one mapping between the trellis branches and the codewords of $\mathcal{O}_K$. References [12, 25], [19, chap. 10], and [27, chap. 5] provide a more detailed discussion of orthogonal convolutional codes.

Biorthogonal convolutional codes [26] are a modification of orthogonal convolutional codes. Instead of using a $(2^K, K)$ orthogonal code in Figure 3.1, a $(2^K, K+1)$ biorthogonal (first order Reed-Muller) code is used. The encoder of a biorthogonal convolutional code is shown in the dotted box of Figure 3.2. The coding rate is $1/2^K$ and the constraint length of the associated convolutional code is $K+1$. Hence any two non-equal data sequences generate at least $K+1$ consecutive non-equal codewords.

In biorthogonal convolutional codes, the first order Reed-Muller code is constructed as follows: the last K bits of the shift register select a codeword of $\mathcal{O}_K$, while the first bit of the shift register determines the polarity of the transmitted codeword. This ensures that any two sequences of exactly $K+1$ consecutive non-equal codewords must contain antipodal codewords (namely the first non-equal codewords of the sequence). Since the minimum Hamming distance of a first order Reed-Muller code is 2^{K-1}, the free distance of the biorthogonal convolutional code is $d_{\text{free}} = (K+2)2^{K-1}$. Note that complement data words are *not* mapped into complement codewords of the first order Reed-Muller code [28, pp. 212]. Otherwise, the free distance of the corresponding biorthogonal convolutional code would only be $d_{\text{free}} = (K+1)2^{K-1}$.

Maximum likelihood decoding is very similar to that of orthogonal convolutional codes. As outlined earlier, half the branch metrics are calculated with a Hadamard transform of the received vector. The remaining metrics are just the opposites of these. It is shown in [26] that the biorthogonal code provides a significant coding gain over the orthogonal convolutional code. The asymptotic coding gain is given by

$$G_{\infty(\text{biortho})} = \frac{1}{2^K} \times (K+2)2^{K-1} = (K+2)/2 \qquad (3.11)$$

This improvement comes at the cost of an increase in the constraint length of the convolutional code, from K to $K+1$.

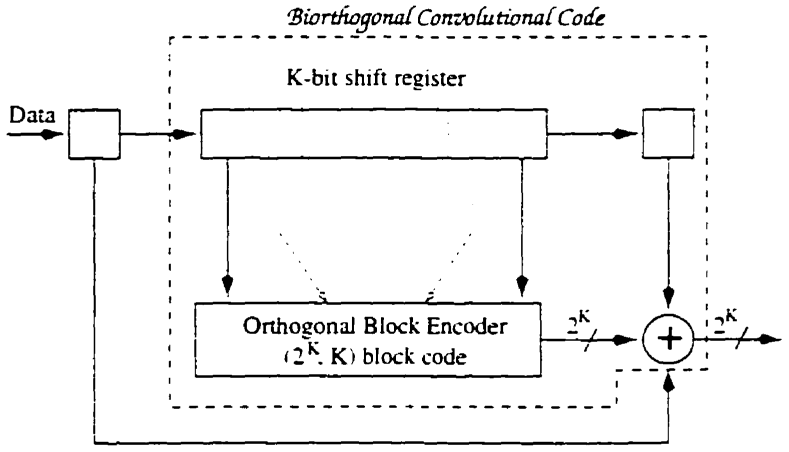


Figure 3.2: Encoder of a superorthogonal convolutional code.

Finally, superorthogonal convolutional codes are a variation on the same theme. Figure 3.2 shows the superorthogonal convolutional encoder, where the relationship to biorthogonal convolutional codes is emphasized. The coding rate is $1/2^K$ and the constraint length of the associated convolutional code is $K+2$. Hence any two non-equal data sequences generate at least $K+2$ consecutive non-equal codewords. Moreover, in such a sequence, the first and last pairs of non-equal codewords must be antipodal pairs. This ensures that the free distance of the superorthogonal convolutional code is $d_{\text{free}} = (K+4)2^{K-1}$. The asymptotic coding gain is given by

$$G_{\infty(\text{suportho})} = \frac{1}{2^K} \times (K+4)2^{K-1} = (K+4)/2 \qquad (3.12)$$

This improvement over the orthogonal and biorthogonal convolutional codes comes at the cost of an increase in the constraint length but not in the associated bandwidth expansion. Note that we have chosen to compare these codes on the basis of equal rate. Hence the asymptotic coding gains of (3.10). (3.11). and (3.12) are all for coding rates of $1/2^K$.

The maximum-likelihood decoding of superorthogonal convolutional codes involves the use of the VA and is practically identical to that of the biorthogonal convolutional code. Again. the branch metrics are obtained from the Hadamard transform of the received vector. and the only difference lies in the trellis structure.

3.1.5 IS-95 Uplink Code

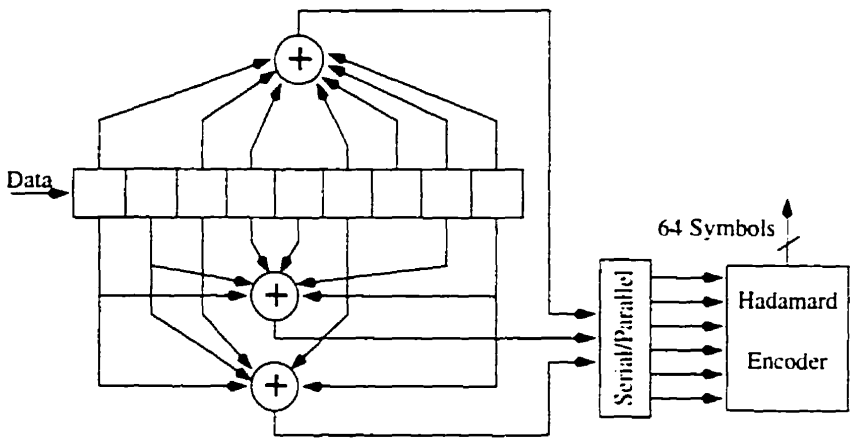


Figure 3.3: Encoder of the IS-95 uplink code.

The error control code used in the IS-95 uplink consists of the combination of a rate 1/3. constraint length 9 convolutional code and a (64. 6) Hadamard code. An encoder for such a code is shown in Figure 3.3. The serial/parallel block converts two successive input vectors of length 3 into a vector of length 6. which is then mapped into a Hadamard codeword. Hence every two data bits get mapped into a 64-symbol codeword. The coding rate of the IS-95 uplink code is therefore 1/32. Note that we ignore here the issues of interleaving and PN sequence spreading that are part of the actual IS-95 standard, since we are interested in the pure coding aspect.

The convolutional code used in the IS-95 uplink code is the maximal free distance code of rate 1/3 and constraint length 9. and has $d_{\text{free}} = 18$. The free distance of the overall IS-95 uplink code is more complicated to find. since it may not be due to the input sequence that generates the minimum-weight output sequence of the convolutional code. We thus resort to heavier machinery. and use the Viterbi decoding algorithm of the IS-95 uplink code to determine the path or paths of minimum weight [33]. This reveals that the free Hamming distance of the IS-95 uplink code is 160, and its asymptotic coding gain is

$$G_{\infty(\text{IS-95})} = 5 \approx 7\,\text{dB} \tag{3.13}$$

The IS-95 uplink code, the superorthogonal convolutional code of rate 1/64. the biorthogonal convolutional code of rate 1/256. and the orthogonal convolutional code of rate 1/1024 have the same asymptotic coding gain. Hence. based on asymptotic coding gains. the IS-95 uplink code is superior to OBS codes.

3.2 Trellis/Reed-Muller Coding

In this section. we present the general idea that led to the construction of Trellis/Reed Muller codes (patent pending). Part of this work was published in [34]. As outlined earlier, these codes are based on the combination of a trellis code with a first-order Reed-Muller code. We explore some methods to estimate analytically the performance of Trellis/Reed-Muller codes, thereby revealing some important code construction rules and constraints. Finally, we explain how the Viterbi algorithm is combined with the fast Hadamard transform to obtain a maximum-likelihood decoding algorithm for Trellis/Reed-Muller codes.

3.2.1 Structure of Trellis/Reed-Muller Codes

In the proposed coding scheme of Figure 3.4. a block code is used jointly with a b-bit input trellis code of constraint length K. The role of the trellis code is to select cosets of the block code. The selected coset code is then used to encode the $m + 1$

remaining data bits into an n bit codeword. The overall code rate is $\frac{m-b+1}{n}$, and is independent of the constraint length of the trellis code. We shall focus on the case where the block code is a $(2^m, m+1)$ first-order Reed Muller (RM) code. Hence all transmitted codewords belong to some coset of a first-order Reed-Muller code, and the overall code rate is $\frac{m+b+1}{2^m}$.

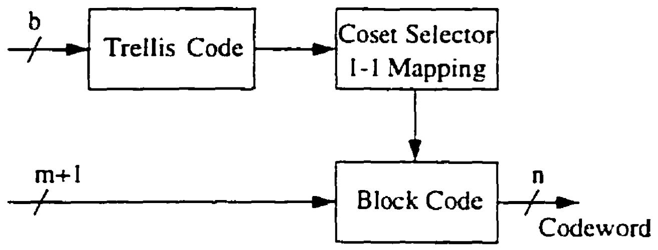


Figure 3.4: Trellis/Reed-Muller coding scheme.

Trellis/Reed-Muller (TRM) codes are a special case of trellis codes. As illustrated in Figure 3.5, the trellis of a TRM code is derived from the trellis of its coset-selecting code by replacing each branch with a set of 2^{m+1} parallel branches. Each of these branches represents a codeword in a RM coset. We shall refer to the trellis of the coset-selecting trellis code as the primary trellis of the TRM code, as opposed to the full TRM trellis.

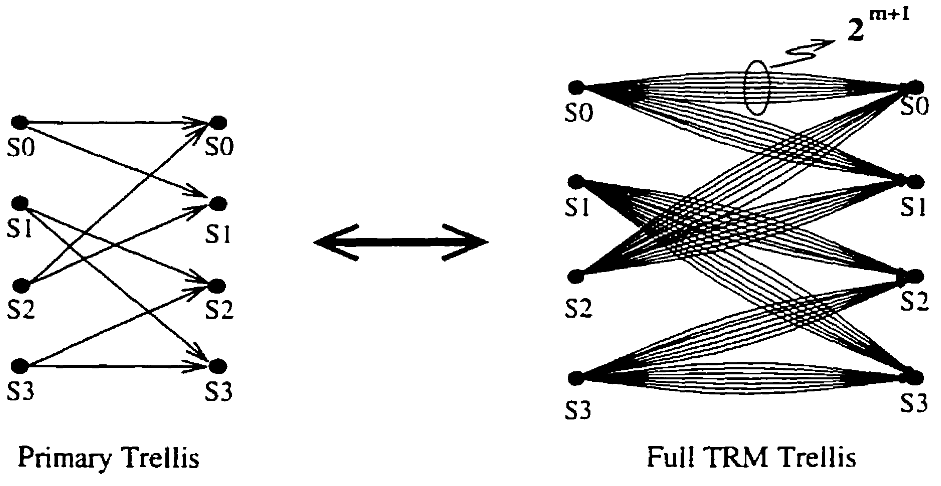


Figure 3.5: Trellis representation of TRM coding.

The Viterbi algorithm can be used directly for maximum likelihood decoding of TRM codes. To each state transition in the trellis, there corresponds a given coset leader. The metrics of each group of 2^{m+1} parallel branches can be obtained by a Hadamard transform of the received vector, after pre-multiplication by the appropriate coset leader.

Let the received vector be $\underline{r} = \underline{c}_i \otimes \underline{\lambda} + \underline{n}$, where $\underline{c}_i$ is a RM codeword, $\underline{\lambda}$ a coset leader, and $\underline{n}$ the noise vector. The samples of the noise vector $\underline{n} = (n_1, \cdots, n_{2^m})$ are i.i.d. Gaussian random variables with mean zero and variance σ^2. The received vector $\underline{r} = (r_1, \cdots, r_{2^m})$, given $\underline{c}_i$ and $\underline{\lambda}$, is a Gaussian random vector with 2^m independent components r_k. Each r_k is then a Gaussian random variable of mean $c_{ik}\lambda_k$ and variance σ^2.

For each primary trellis branch, a Hadamard transform of $\underline{r} \otimes \underline{\lambda}'$ is computed; $\underline{\lambda}'$ is the coset leader associated with that particular primary trellis branch. This computation provides sufficient information to discard all but one of the 2^{m+1} parallel branches of the full TRM trellis. The metric of the remaining branch in the full trellis is the largest component (in absolute value) of the Hadamard transform of $\underline{r} \otimes \underline{\lambda}'$. The Hadamard transform of $\underline{r} \otimes \underline{\lambda}'$ can be written as

$$\mathcal{H}(\underline{r} \otimes \underline{\lambda}') = 2^{-m/2} (\underline{r} \otimes \underline{\lambda}') H_m = 2^{-m/2} \underline{r} H'_m \tag{3.14}$$

where the matrix H'_m is obtained from H_m by multiplying each column of H_m by $\underline{\lambda}'$. Hence $\mathcal{H}(\underline{r} \otimes \underline{\lambda}')$ is obtained by a linear transformation of $\underline{r}$. Since $\underline{r}$ is a Gaussian random vector, the Hadamard transform of $\underline{r} \otimes \underline{\lambda}'$ is also a Gaussian random vector with the following properties (see section B.1 of Appendix B for the details of the computation)

$$E\left[\mathcal{H}(\underline{r} \otimes \underline{\lambda}')\right] = \begin{cases} (0, \cdots, 0, \pm 2^{m/2}, 0, \cdots, 0) & \lambda = \lambda' \\ 2^{-m/2} (\underline{c}_i \otimes \underline{\lambda} \otimes \underline{\lambda}') H_m & \lambda \neq \lambda' \end{cases} \tag{3.15}$$

$$\mathrm{Cov}\left[\mathcal{H}(\underline{r} \otimes \underline{\lambda}')\right]_{kl} = \begin{cases} 2^{-m}\sigma^2 & k = l \\ 0 & \text{Otherwise} \end{cases} \tag{3.16}$$

Therefore the vector $\underline{r} \otimes \underline{\lambda}'$ is a Gaussian random vector with independent components of variance $2^{-m}\sigma^2$ and mean given by (3.15). When $\underline{\lambda} \neq \underline{\lambda}'$, the considered path is incorrect: the corresponding metric is given by the maximum (in absolute value) component of a Gaussian random vector (with independent components of equal variance) and mean given by (3.15). In order to minimize this metric (which is equivalent to maximizing the distance of an incorrect path to the correct path) we need to minimize the maximum value of the mean vector (3.15) given by

$$\begin{aligned} 2^{-m/2}\max|(\underline{c}_i\otimes\underline{\lambda}\otimes\underline{\lambda}')H_m| &= 2^{-m/2}\max_{k=1,\cdots,2^m}|<\underline{c}_i\otimes\underline{\lambda}\otimes\underline{\lambda}'|\underline{c}_k>| \\ &= 2^{-m/2}\max_{k=1,\cdots,2^m}|<\underline{\lambda}\otimes\underline{\lambda}'|\underline{c}_i\otimes\underline{c}_k>| \\ &= 2^{-m/2}\max_{l=1,\cdots,2^m}|<\underline{\lambda}\otimes\underline{\lambda}'|\underline{c}_l>| \\ &= \mathcal{M}ax\,|\mathcal{H}(\underline{\lambda}\otimes\underline{\lambda}')| \end{aligned} \qquad (3.17)$$

where $\underline{c}_l \equiv \underline{c}_i\otimes\underline{c}_k$ is also a RM codeword (because the first-order Reed-Muller code is linear). From Parseval's equality, if $\underline{F} = (F_1,\cdots,F_{2^m})$ is the Hadamard transform of a vector $\underline{f} = (f_1,\cdots,f_{2^m})$ then

$$\sum_{u=1}^{2^m} F_u^2 = \sum_{u=1}^{2^m} f_u^2 = ||\underline{f}||^2 \qquad (3.18)$$

Moreover, if f is a bipolar vector ($f_u = \pm 1$), then $\sum_{u=1}^{2^m} F_u^2 = 2^m$. This implies that $||\mathcal{H}(\underline{\lambda}\otimes\underline{\lambda}')|| = 2^m$. Therefore (3.17) is minimized when all components of $\mathcal{H}(\underline{\lambda}\otimes\underline{\lambda}')$ are equal to ± 1. When m is even, this property defines a bent function. Therefore, in order to minimize (3.17), the vector $\underline{\lambda}\otimes\underline{\lambda}'$ has to be a bent function.

When m is even, there exists a set of vectors with the property of being furthest away from all codewords in $R(1,m)$. Such vectors are called bent functions [18, pp. 426-428]. A bent function has Hamming distance $2^{m-1} \pm 2^{m/2-1}$ to any codeword in $R(1,m)$. Also if $\underline{v}$ is not a bent function, it has distance less than $2^{m-1} - 2^{m/2-1}$ from some codeword of R(1,m). Equivalently, the Hadamard transform of a bent function consists only of ± 1. Moreover, if $\underline{u}$ is a bent function and $\underline{c}$ a RM codeword, $\underline{u}\otimes\underline{c}$ is also a bent function. In summary, if $\underline{\lambda}\otimes\underline{\lambda}'$ is bent, then we can write (3.15) as

$$E[\mathcal{H}(\underline{r}\otimes\underline{\lambda}')] = \begin{cases} (0,\cdots,0,\pm 2^{m/2},0,\cdots,0) & \lambda = \lambda' \\ (\pm 1,\pm 1,\cdots,\pm 1,\pm 1) & \lambda \neq \lambda' \end{cases} \qquad (3.19)$$

It is therefore desirable to construct a set Λ of vectors such that the product of any two distinct vectors is a bent function. This is always possible if we choose a small number of such vectors (say 2). However, we would like to construct Λ to be as large as possible. If we allow one of the coset leaders to be the zero vector, it follows that all non-zero coset leaders which minimize (3.17) must be bent functions themselves. This is only a necessary condition. The construction of a complete set of bent functions $\Lambda = \{\underline{\lambda}_i\}$ with the property that the product $\underline{\lambda}_i \otimes \underline{\lambda}_j$, $i \neq j$, is again a bent function (not necessarily in the set Λ), is closely related to the construction of the Kerdock code [18, 35]. This construction method, which yields 2^{m-1} coset leader candidates, is described also in [36]. In the case of odd values of m, or when many coset leaders are needed, sets of vectors which yield small although non-optimal values of (3.17) could be considered. This issue is briefly mentioned in [36]. In this work, we consider only the case where m is even and the set of coset leaders constructed as in [36]. There are then 2^{m-1} coset leader candidates.

The union of the first order Reed-Muller code and its 2^{m-1} cosets described in the above paragraph form a Kerdock code of order m, $\mathcal{K}(m)$ [18, 35, 36]. The Kerdock code is a nonlinear binary block code with 2^{2m} codewords of length 2^m [18, pp. 456]. Its minimum Hamming distance is $2^{m-1} - 2^{m/2-1}$. One can view TRM coding as a partition of the Kerdock code into its Reed-Muller sub-codes (RM cosets), and the transmission of codewords from one sub-code at a time. The improved performance comes from the fact that the minimum distance of a first order Reed-Muller code is larger than that of the Kerdock code. This approach is conceptually similar to trellis-coded modulation (TCM), applied to block codes instead of signal constellations. In a spread-spectrum system where bandwidth expansion is possible, the rich structure of some low rate block codes such as the Kerdock code makes that approach very promising.

3.2.2 General Considerations in Trellis Code Construction

The previous section dealt with the coset partitioning issue, without specifying how to design the coset-selecting code. We now address the important issues in the construction of good coset selecting codes.

Each branch of the primary trellis of a TRM code is mapped into a corresponding coset leader. Let β_1 and β_2 be any two branches in the primary trellis of a TRM code, mapped into coset leaders $\underline{\lambda}_1$ and $\underline{\lambda}_2$ respectively. We define the modified distance between β_1 and β_2 as

$$d_{\mathrm{m}}(\beta_1, \beta_2) = \begin{cases} 0 & \text{if } \underline{\lambda}_1 = \underline{\lambda}_2 \\ 1 & \text{if } \underline{\lambda}_1 \neq \underline{\lambda}_2 \end{cases} \tag{3.20}$$

Similarly, let $\pi_i = (\beta_{i_1}, \beta_{i_2}, \cdots)$ and $\pi_j = (\beta_{j_1}, \beta_{j_2}, \cdots)$ be two paths in the trellis. The modified distance between path π_i and path π_j is

$$d_{\mathrm{m}}(\pi_i, \pi_j) = \sum_{k=1}^{\infty} d_{\mathrm{m}}(\beta_{i_k}, \beta_{j_k}) \tag{3.21}$$

The modified free distance of the code is the minimum modified distance between any two distinct paths in its primary trellis. Thus,

$$d_{\mathrm{mf}} = \min_{\substack{(\pi_i, \pi_j) \\ \pi_i \neq \pi_j}} d_{\mathrm{m}}(\pi_i, \pi_j) \tag{3.22}$$

The minimum Hamming distance of a first-order Reed-Muller code is 2^{m-1}. The minimum Hamming distance between two non-parallel branches in the TRM full trellis is $2^{m-1} - 2^{m/2-1}$. Hence the minimum Hamming distance between two paths π_i and π_j in the full trellis is $d_{\mathrm{m}}(\pi_i, \pi_j)(2^{m-1} - 2^{m/2-1})$. It follows that the free (Hamming) distance of a TRM code is

$$\begin{aligned} d_{\mathrm{free}} &= \min(2^{m-1}, d_{\mathrm{mf}} \times (2^{m-1} - 2^{m/2-1})) \\ &= 2^{m-1} \min(1, d_{\mathrm{mf}}(1 - 2^{-m/2})) \end{aligned} \tag{3.23}$$

Since $m \geq 2$, then $d_{\mathrm{free}} = 2^{m-1}$ whenever $d_{\mathrm{mf}} \geq 2$. The free distance of the TRM code is thus determined by the minimum Hamming distance of the RM code. Therefore, for high signal to noise ratios, the performance of a TRM code is limited by the Reed-Muller code. Also, the asymptotic coding gain of a TRM code is

$$G_{\infty(\mathrm{TRM})} = \frac{m+b+1}{2^m} \times 2^{m-1} = \frac{m+b+1}{2} \tag{3.24}$$

which is very promising when compared to the asymptotic coding gains of the OBS codes of similar rate. Moreover, we can increase the asymptotic coding gain of a TRM code while reducing the corresponding bandwidth expansion by increasing the value of b. In orthogonal and biorthogonal convolutional codes, the asymptotic coding gain can only be increased at the expense of a greater bandwidth expansion.

Since the performance of TRM codes is limited by the Reed-Muller code, there is little to be gained from using very powerful trellis codes (i.e. with large d_{mf}). A good coset-selecting trellis code can have $d_{\mathrm{mf}} = 2$, and as few states as possible, for a given number of inputs. Since the number of trellis states (and therefore the decoder complexity) increases exponentially with the product Kb, a small value of K allows for larger values of b if Kb is to be kept constant. This in turn increases the asymptotic coding gain.

The following general rules can be used to construct good coset-selecting trellis codes with b inputs. We shall say that a coset-selecting trellis code is *good* if

1. All coset leaders should occur with the same frequency. This rule is similar to that of TCM [37, 38] and [39, pp. 78]

2. No parallel branches. This implies that on the trellis diagram, each state has exactly 2^b branches going out to 2^b distinct next states. Otherwise, we would have a trellis with a modified free distance of 1.

3. All the branches emanating from or merging into the same state must be mapped to distinct coset leaders. This rule, together with rule 2, ensures that the modified free distance of the trellis code is at least 2.

4. Non-catastrophicity. The code must not allow for the existence of two infinite paths with a finite number of different coset leaders, but generated from two input bit sequences with an infinite Hamming distance between them.

An important consequence of rule 2 is that the number of delay elements needed to implement a good coset-selecting trellis code is at least b, since there must be 2^b distinct states.

3.2.3 Performance Analysis in AWGN

If the trellis code is modified-distance invariant, then one can assume, without loss of generality, that the correct path in the primary trellis is π_0, the zero path (i.e. the path generated by an input sequence of all zeroes to the trellis encoder).

Proposition: Any convolutional code is modified-distance invariant.

Proof: Let $\pi = (\underline{u}_1, \underline{u}_2, \cdots)$ and $\pi' = (\underline{v}_1, \underline{v}_2, \cdots)$ be two paths in the trellis of a convolutional code. The modified distance between path π and path π' is

$$\begin{aligned} d_{\mathrm{m}}(\pi, \pi') &= \sum_{k=1}^{\infty} d_{\mathrm{m}}(\underline{u}_k, \underline{v}_k) = \sum_{k=1}^{\infty} \begin{cases} 1 & \text{if } \underline{u}_k \neq \underline{v}_k \\ 0 & \text{if } \underline{u}_k = \underline{v}_k \end{cases} = \sum_{k=1}^{\infty} \begin{cases} 1 & \text{if } \underline{u}_k + \underline{v}_k \neq \underline{0} \\ 0 & \text{if } \underline{u}_k + \underline{v}_k = \underline{0} \end{cases} \\ &= \sum_{k=1}^{\infty} d_{\mathrm{m}}(\underline{u}_k + \underline{v}_k, \underline{0}) = d_{\mathrm{m}}(\pi + \pi', \pi_0) \end{aligned} \tag{3.25}$$

□

For the purpose of this section, we shall assume that the coset-selecting code is linear. In other words, it can be constructed from a convolutional code followed by a mapping from the codewords of the convolutional code to the set of coset leaders. One may question, at this point, the impact of such a restriction on the structure of the coset-selecting code. However, we shall see later on that the coset selecting codes of interest fall in that category.

Assuming a convolutional coset-selecting code, the TRM code is distance invariant. The bit error probability can be upper bounded by using an approach similar to that of [19, pp. 253-356]. Consider the primary trellis of the code, and let $P(i)$ denote the probability that path π_i (diverging from the zero path at some fixed node j) is selected. If we denote the bit error probability of the Reed-Muller code by p_{RM}, then the average bit error probability of the TRM code is bounded by

$$\frac{m+1}{m+b+1} p_{RM} \leq P_b \leq \frac{m+1}{m+b+1} p_{RM} P(0) + \frac{1}{m+b+1} \sum_{i=1}^{\infty} E[n_b(i)] P(i) \tag{3.26}$$

where $E[n_b(i)]$ is the average number of bit errors on path π_i. Let l_i be the number of non-equal branches between path π_i and path π_0, d_i the modified distance between path π_i and path π_0, and β_i the Hamming weight of the input sequence generating

π_i. We then have

$$E[n_b(i)] \;=\; \beta_i + (m+1)p_{RM}(l_i - d_i) + (m+1)d_i p_{\text{bent}} \tag{3.27}$$

Indeed, when path π_i is selected ($i \neq 0$), we first make an error in the β_i non-zero bits in the input sequence generating π_i. On each branch where the coset leader is correct, the bit error probability is that of a RM code, and hence the average number of errors is $(m+1)p_{RM}$. Finally, on branches where the coset leader differs from that of the correct path, the bit error probability is denoted by p_{bent}, and the average number of errors is $(m+1)p_{\text{bent}}$.

The upper bound of (3.26) follows from a union-bounding argument [19, pp. 253-356]. The lower bound can be derived by assuming a very powerful trellis code such that $P_0 \to 1$, and therefore the correct path is always selected. This is clearly an optimistic situation. The b bits of the trellis code are always error free, whereas the only errors that can occur, with probability p_{RM}, are in the $m+1$ bits of the RM codewords. Thus, in this optimistic scenario, the overall average probability of error is $\frac{m+1}{m+b+1}p_{RM}$.

The exact expression for p_{RM} is given in [28, pp. 210-212]. The computation of the exact expression for p_{bent} follows the same reasoning, but it turns out that its actual value does not vary much with the signal to noise ratio, and is around 0.67. It can be upper bounded by 1, without affecting significantly the upper bound of (3.26).

From (3.26) and (3.27), we have

$$\begin{aligned} P_b \;\le\; & \frac{m+1}{m+b+1}p_{RM}P(0) \;+\; \frac{1}{m+b+1}\sum_{i=1}^{\infty}\Big\{(m+1)(p_{\text{bent}} - p_{RM})d_iP(i) \\ & \qquad\qquad + l_i(m+1)p_{RM}P(i) + \beta_iP(i)\Big\} \\ \le\; & \frac{m+1}{m+b+1}p_{RM} \;+\; \frac{m+1}{m+b+1}(p_{\text{bent}} - p_{RM})\sum_{i=1}^{\infty} d_iP(i) \\ & + \frac{m+1}{m+b+1}p_{RM}\sum_{i=1}^{\infty} l_iP(i) \;+\; \frac{1}{m+b+1}\sum_{i=1}^{\infty}\beta_iP(i) \end{aligned} \tag{3.28}$$

where we upper bounded $P(0)$ (the probability of selecting the correct path) by 1.

We define here the modified augmented generating function $T(D,L,I)$, which is obtained from the classical definition given in [19] by replacing the Hamming distance between branches by the modified distance. Recall that the modified distance between any two branches in the trellis is either zero (if the coset leaders are the same) or one (if the coset leaders are different). Hence in the labeling of the state diagram of the code, from which the augmented generating function is derived, the variable D can only have exponents 0 or 1. Note that the free distance is often the figure of merit used in trellis code search and trellis code design. In our case, only the modified free distance matters. Hence a good code, in terms of its Hamming free distance, is not necessarily a good coset selecting code for TRM coding.

The modified augmented generating function yields the modified distance distribution of the primary trellis. Let $g(j, d, l)$ be the number of paths generated by an input sequence of weight j, at a modified distance d from the zero path, and with a diverging length of l branches. Let also $c(d, l)$ be the number of paths of diverging length l and at modified distance d from the zero path. Finally, let $a(d)$ be the number of paths at modified distance d from the zero path. If d_f is the modified free distance of the primary trellis code, then we have,

$$c(d, l) = \sum_{j=1}^{\infty} g(j, d, l) \tag{3.29}$$

$$a(d) = \sum_{j=1}^{\infty} \sum_{l=d_f}^{\infty} g(j, d, l) = \sum_{l=d_f}^{\infty} c(d, l) \tag{3.30}$$

If we define $P(d, l)$ as the probability that a path of length l and modified weight d is chosen, given that the zero path was transmitted, we can rewrite (3.28) as

$$\begin{aligned} P_b \leq\ & \frac{m+1}{m+b+1} p_{RM} + \frac{m+1}{m+b+1} (p_{\text{bent}} - p_{RM}) \sum_{l=d_{\text{mf}}}^{\infty} \sum_{d=d_{\text{mf}}}^{l} d\, c(d, l) P(d, l) \qquad (3.31) \\ & + \frac{(m+1) p_{RM}}{m+b+1} \sum_{l=d_{\text{mf}}}^{\infty} \sum_{d=d_{\text{mf}}}^{l} l\, c(d, l)\, P(d, l) + \frac{1}{m+b+1} \sum_{l=d_{\text{mf}}}^{\infty} \sum_{d=d_{\text{mf}}}^{l} \sum_{j=1}^{\infty} j g(j, d, l) P(d, l) \end{aligned}$$

and defining

$$h(d, l) \equiv \sum_{j=1}^{\infty} j g(j, d, l) \tag{3.32}$$

we end up with

$$P_b \leq \frac{m+1}{m+b+1} p_{RM} + \frac{m+1}{m+b+1} (p_{\text{bent}} - p_{RM}) \sum_{l=d_{\text{mf}}}^{\infty} \sum_{d=d_{\text{mf}}}^{l} d\, c(d, l) P(d, l) \tag{3.33}$$

$$+\ \frac{m+1}{m+b+1}p_{RM}\sum_{l=d_{mf}}^{\infty}\sum_{d=d_{mf}}^{l} c(d,l)\, lP(d,l)\ +\ \frac{1}{m+b+1}\sum_{l=d_{mf}}^{\infty}\sum_{d=d_{mf}}^{l} h(d,l)P(d,l)$$

In section B.2 of Appendix B, we show that $P(d,l)$ is upper bounded by

$$\tilde{P}(d,l)\ =\ \sum_{i=0}^{d}\sum_{j=0}^{l-d}\sum_{k=0}^{l-d-j} A(i,j,k,d,l)Q\left(\sqrt{\frac{2(iw_1+(d-i)\,w_2+k2^{m+1}+(l-d-j-k)2^m)E_s}{N_o}}\right)$$

where the energy per channel symbol E_s is related to the energy per data bit E_b and the energy per codeword E_d by

$$E_s\ =\ \frac{m+b+1}{2^m}E_b\ =\ \frac{1}{2^m}E_d \tag{3.34}$$

and

$$A(i,j,k,d,l)\ =\ \begin{cases} \binom{d}{i}2^{md}\binom{l-d}{j}\binom{l-d-j}{k}(2^{m+1}-2)^{l-d-j-k} & \begin{cases} i=0,\cdots,d \\ j=0,\cdots,l-d \\ k=0,\cdots,l-d-j \end{cases} \\ 0 & \text{Otherwise} \end{cases} \tag{3.35}$$

is the number of paths of length l, modified weight d and Hamming weight $iw_1 + (d-i)w_2 + k2^{m+1} + (l-d-j-k)2^m$.

We also derive in the same section of Appendix B a simpler albeit looser upper bound on the bit error probability, given by

$$\begin{aligned} P_b\ \le\ & \frac{m+1}{m+b+1}p_{RM} + \frac{m+1}{m+b+1}f\left(\frac{d_f w_1 E_s}{N_o}\right)\Bigg\{ (p_{\text{bent}}-p_{RM})\left[D\frac{\partial T(D,L,I)}{\partial D}\right]_{\substack{D=e^{-w_1E_s/N_o}\\ I=1,L=2^{m+1}}} \\ & + p_{RM}\left[L\frac{\partial T(D,L,I)}{\partial L}\right]_{\substack{D=e^{-\frac{w_1E_s}{N_o}}\\ I=1,L=2^{m+1}}} + \frac{1}{m+1}\left[\frac{\partial T(D,L,I)}{\partial I}\right]_{\substack{D=e^{-\frac{w_1E_s}{N_o}}\\ I=1,L=2^{m+1}}}\Bigg\} \end{aligned} \tag{3.36}$$

where $f(x) \equiv Q(\sqrt{2x})e^x$. The bound of (3.36) is in closed form, and expressed in function of the transfer function of the code (and its derivatives). It has the familiar form of the upper bounds on the error probabilities of linear convolutional codes (e.g. [19]). Such upper bounds are useful at high signal to noise ratio only .

3.2.4 Unequal Error Protection

TRM codes have an inherent feature which consists in unequal error protection. Indeed, the b input bits to the coset-selecting code are better protected since the probability of path error is less than that of branch (Reed-Muller code) error. This is equivalent to the observation that the performance of the TRM code is limited by that of the Reed-Muller code. In general, unequal error protection is not necessarily a benefit. In this case however, the average bit error probability is almost equal to the bit error probability of the $m + 1$ bits fed to the the Reed-Muller encoder, which we refer to as p_m. Indeed, let p_b be the bit error probability of any of the b bits fed to the coset-selecting, code; the average bit error probability is then

$$\bar{p} = \frac{m+1}{m+b+1}p_m + \frac{b}{m+b+1}p_b \tag{3.37}$$

Given the allowable values of b, we have that $1/2 < \frac{m+1}{m+b+1} < 1$. Hence even if $p_b \ll p_m$, we obtain

$$\frac{1}{2}p_m < \bar{p} < p_m \tag{3.38}$$

Hence the average bit error probability is never less than half the bit error probability of the least protected bits. Thus, for all practical purposes, we consider the average bit error probability as a good measure of both the overall code performance and the error probability of its least protected bits.

On the other hand, at relatively high signal to noise ratio, it turns out that $p_b \ll p_m$. In applications such as speech coding, some data bits are more important than others. The use of TRM codes is naturally indicated to such situations.

3.2.5 Decoding of TRM Codes

As mentioned earlier, Trellis/Reed Muller can be viewed as trellis codes (refer to Figure 3.5). The Viterbi algorithm can then be used for maximum likelihood decoding of TRM codes. To each state transition in the primary TRM trellis, there corresponds a given coset leader. The metrics of each group of 2^{m+1} parallel branches can be obtained by a Hadamard transform of the received vector, after pre-multiplication by the appropriate coset leader.

Once the metrics of a group of 2^{m+1} parallel branches is computed, only the branch with the largest metric is kept, in accordance with the add-compare-select procedure

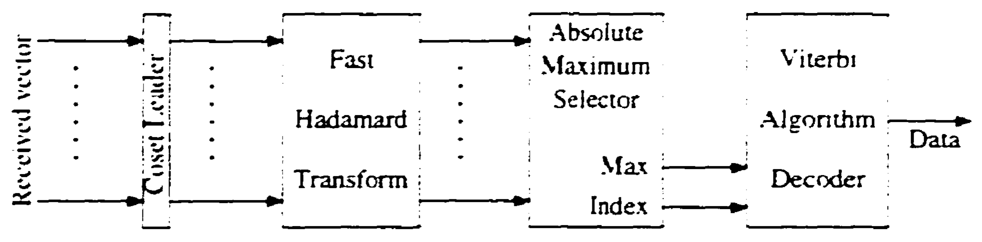


Figure 3.6: Branch metric computer for maximum-likelihood decoding of TRM codes.

of the Viterbi algorithm. Indeed, since all the branches in question emanate from the same branch, the "add" operation can be skipped, and selecting the branch with the highest metric corresponds to the compare-select operation.

In principle, the metrics of each branch can be calculated by a metric computer as illustrated in Figure 3.6. The first block in Figure 3.6 "adds" the branch coset leader to the received vector, and is the only element that varies from one branch computer to the other. The "absolute maximum selector" block finds out which component of its input vector has the largest absolute value, and outputs that component (signed) value together with the corresponding index. This allows the Viterbi algorithm to select exactly one codeword on the trellis branch together with its associated likelihood metric.

In general, there may be more than one state transition for each associated coset leader. In other words, different state transitions may have the same associated coset leaders. Obviously, the corresponding branches need only one branch metric computer. Furthermore, the high parallelism between different branch metric computers can be exploited to reduce the decoding complexity by using common intermediate results between Fast Hadamard Transforms. On the other hand, if the transmission rate is not too large, a single branch metric computer is sufficient: the same *Fast Hadamard Transform* and *Absolute Maximum Selector* blocks are used repetitively with varying coset leaders in the *coset leader block*.

3.3 Short Constraint Length TRM Codes (TRM1)

As shown in section 3.2.2. the free Hamming distance of the TRM code should be made equal to the minimum Hamming distance of the first-order Reed-Muller code. In this section. we consider the simplest good coset-selecting convolutional codes with one delay element per input bit.

3.3.1 Code Construction

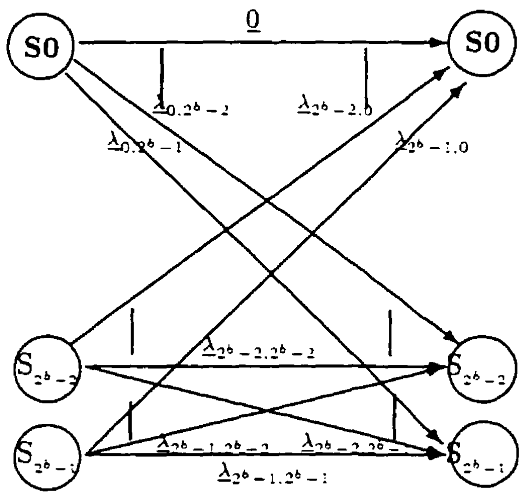
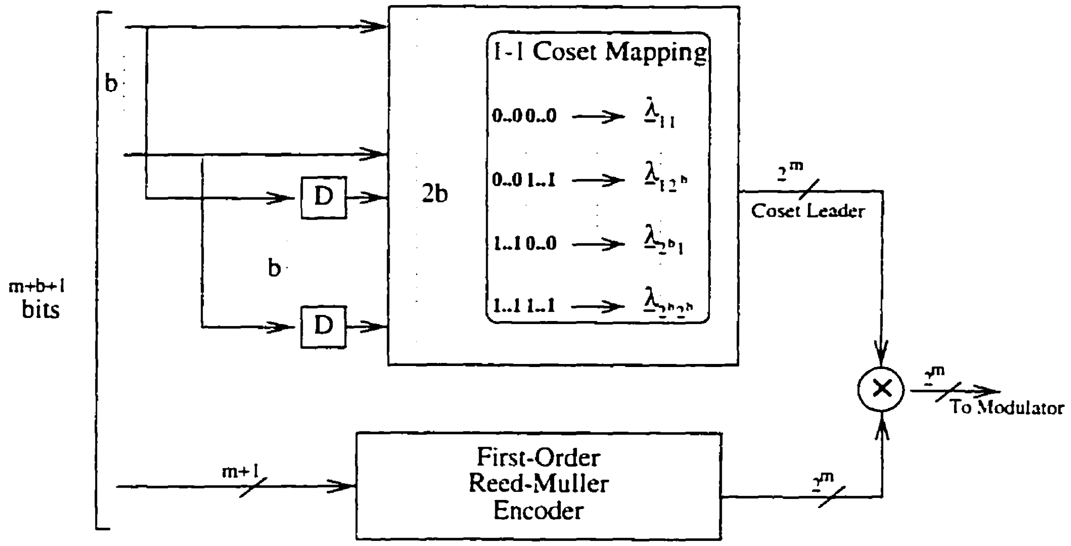


Figure 3.7: Primary trellis structure for b input bits and one delay per input bit.

Rule 2 of section 3.2.2 states that. on the trellis diagram. each state has exactly 2^b branches going out to the 2^b distinct next states. Since we have exactly 2^b states available. this rule fixes the primary trellis structure. illustrated in Figure 3.7. which is referred to as a fully connected trellis.

A simple mapping from the primary trellis branches to the coset leaders. which satisfies the rules of section 3.2.2. is a bijective map (one-to-one and onto). where each of the 2^{2b} primary trellis branches is mapped to a unique coset leader. It follows that the set of coset leaders must have at least 2^{2b} elements. Actually. since there are 2^{m-1} possible coset leaders. with the constraint that m is even. the set of coset leaders must contain 2^{2b-1} elements. In other words. for a RM code of length 2^m. the number of possible coset leaders is 2^{m-1}. which implies that the maximal value of b

is $b_{\max} = m/2 - 1$. From now on we will concentrate only on the case $b = b_{\max}$, and refer to the resulting codes as TRM1 codes.



b
m+b+1 bits
D
D
b
2b
1-1 Coset Mapping
0..00..0 → λ_{11}
0..01..1 → λ_{12^b}
1..10..0 → λ_{2^b1}
1..11..1 → $\lambda_{2^b2^b}$
2^m
Coset Leader
2^m
To Modulator
m+1
First-Order Reed-Muller Encoder
2^m

Figure 3.8: TRM1 codes with a constraint length of 2.

A possible implementation of a TRM1 code is shown in Figure 3.8. Note that the coset selecting code consists of a convolutional code followed by a mapping. The TRM1 code of Figure 3.8 requires exactly 2^{2b} coset leaders: even for $b = b_{\max}$, only half the available coset leaders are employed.

The convolutional code of Figure 3.8 has a modified free distance of 2. Hence the free distance of the TRM1 code is

$$d_{\text{free}} = \min(2^{m-1}, 2(2^{m-1} - 2^{m/2-1})) = \min(2^{m-1}, 2^m - 2^{m/2}) = 2^{m-1} \quad (3.39)$$

which shows that its asymptotic coding gain is

$$G_{\infty(\text{TRM1})} = \frac{m+b+1}{2} \quad (3.40)$$

3.3.2 Performance Analysis

We shall now analyze the performance of the TRM1 code of Figure 3.8. Without loss of generality, we can assume that the correct path is the "zero" path (i.e. all the input bits to the trellis code are zero), since the coset-selecting code of TRM1 codes is linear, and hence modified distance invariant. We shall first base our performance evaluation on the closed form upper bound of (3.36), and then on the tighter bound of (3.33–3.35).

Closed-Form Upper Bound

The state equations of the convolutional coset-selecting code of a TRM1 code are easily derived from the trellis diagram of Figure 3.7. A trellis branch linking two states S_i and S_j (i and j not both zero) has a modified weight weight of 1, and is generated by an input of Hamming weight $\omega(j)$, where $\omega(j)$ is the Hamming weight of the binary representation of j. Therefore, the state equations are

$$\xi_j = DLI^{\omega(j)} \sum_{i=0}^{2^b-1} \xi_i \qquad j = 1, \cdots, 2^b - 1 \tag{3.41}$$

$$\xi_0' = DL \sum_{i=1}^{2^b-1} \xi_i \tag{3.42}$$

Summing over j in (3.41),

$$\begin{aligned}\sum_{j=1}^{2^b-1} \xi_j &= \sum_{j=1}^{2^b-1} DLI^{\omega(j)} \sum_{i=0}^{2^b-1} \xi_i = DL\left(\sum_{k=1}^{b} \binom{b}{k} I^k\right)\left(\xi_0 + \sum_{j=1}^{2^b-1} \xi_j\right) \\ &= DL\left[(1+I)^b - 1\right]\left(\xi_0 + \sum_{j=1}^{2^b-1} \xi_j\right)\end{aligned} \tag{3.43}$$

Thus,

$$\sum_{j=1}^{2^b-1} \xi_j = \frac{DL\left[(1+I)^b - 1\right]}{1 - DL\left[(1+I)^b - 1\right]}\xi_0 \tag{3.44}$$

Using this in (3.42)

$$T(D, L, I) = \frac{D^2L^2\left[(1+I)^b - 1\right]}{1 - DL\left[(1+I)^b - 1\right]} \tag{3.45}$$

The upper bound of (3.36) is useful provided that the signal to noise ratio is large enough so that the infinite sums of (B.11–B.15) converge. An equivalent condition is that the variable D (evaluated at $D = e^{-w_1 E_s/N_o}$) in the transfer function is smaller than the smallest pole of $T(D, L, I)|_{I=1, L=2^{m+1}}$. Then the condition is

$$\begin{aligned} D &< \frac{1}{2^{m+1}(2^b - 1)} \\ \Leftrightarrow w_1 E_s/N_o &> \ln\left(2^{m+1}(2^b - 1)\right)\end{aligned} \tag{3.46}$$

Using (3.34), and the fact that $2^b - 1 < 2^b$, convergence is ensured if

$$\frac{2^{m-1} - 2^{m/2-1}}{2^m} E_b/N_o > \ln 2 \tag{3.47}$$

For large m, this condition is equivalent to $E_b/N_o > 1.42$ dB. For the smallest value of m that we consider ($m = 6$), the condition is $E_b/N_o > 2$ dB. Naturally, this condition ensures that the upper bound of (3.36) is finite, but does not guarantee its tightness.

Truncated Summation Method

We now consider using (3.33–3.35) directly to upper bound the bit error probability of TRM1 codes, in the search for tighter evaluations of performance. In this method, the Q functions are not replaced by exponentials, which results in a tighter upper bound on the overall bit error probability. The improvement is most substantial at low signal to noise ratios, where the exponential upper bound on the Q function is rather loose. In section B.3 of Appendix B, we derive the following upper bound on the bit error probability of TRM1 codes:

$$\begin{aligned} P_b \leq \; & \frac{m+1}{m+b+1} p_{RM} + \frac{m+1}{m+b+1} p_{\text{bent}} \sum_{d=d_{\text{mf}}}^{\infty} d(2^b - 1)^{d-1} \sum_{i=0}^{d} A(i, d, d) P_{iw_1+(d-i)w_2} \\ & + \frac{b 2^{b-1}}{m+b+1} \sum_{d=d_{\text{mf}}}^{\infty} (d-1)(2^b - 1)^{d-2} \sum_{i=0}^{d} A(i, d, d) P_{iw_1+(d-i)w_2} \end{aligned} \tag{3.48}$$

For computation purposes, we actually approximate the upper bound by taking a sufficiently large number of terms in the infinite sums of (3.33–3.35).

We shall denote the right-hand side of (3.48) by $\tilde{P}_b$.

The lower bound $\frac{m+1}{m+b+1} p_{RM}$ and upper bounds of (3.48) and (3.36) (transfer function bound) are plotted in Figure 3.9 for TRM1 codes with $m = 6, 10, 12$ and 14. It is clear that the truncated path-enumerating bound is tighter than the transfer function upper bound, particularly for the smaller values of m. The upper and lower bounds get increasingly tight as the bit error probability decreases. For bit error probabilities around 10^{-5}, all the computed bounds are indistinguishable, which allows a precise estimation of the required E_b/N_0 to achieve bit error rates of 10^{-5} or less. Reliable estimates are also available for bit error rates of 10^{-4} especially for the lower values of m.

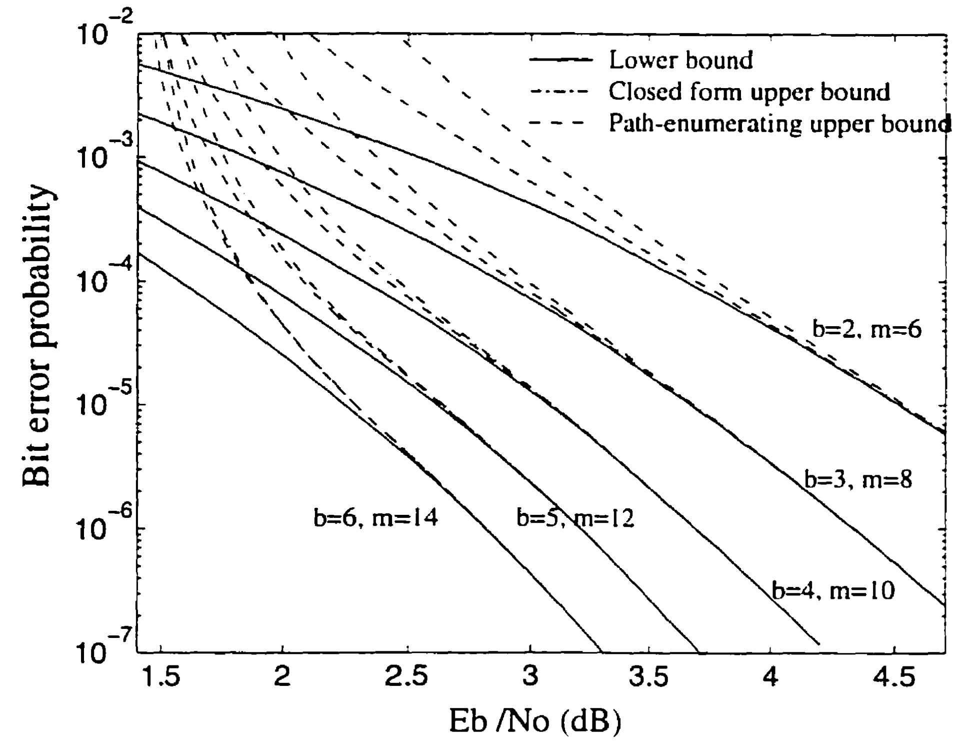


Figure 3.9: Bit error probability bounds for TRM1 codes.

Table 3.1 summarizes the conclusions that can be drawn from Figure 3.9, by displaying the required E_b/N_0 to achieve bit error probabilities of 10^{-4}, 10^{-5}, and 10^{-6}. Whenever the difference between the upper and lower bound of Figure 3.9 is less than 0.1 dB, we consider the average of the upper and lower bound as a reliable estimate, thus ensuring that the maximal error is never greater than 0.05 dB. Otherwise, we list the corresponding E_b/N_0 range (lower and upper bound). Bit error probabilities of 10^{-3} and larger are not considered because the bounds are not tight enough in this region.

At low values of E_b/N_o and relatively high bit error probabilities, the upper bound becomes too loose to characterize reliably the performance of the code. Fortunately enough, simulations are particularly practical at moderate to high probability of errors. Thus in the region where the upper and lower bounds are not tight enough, simulations can take over to provide reliable predictions of the bit error probabilities. Simulation results are presented in section 4.2

TRM Code: m	**6**	**8**	**10**	**12**	**14**
Bandwidth Expansion ($b = b_{\max}$)	7.11	21.33	68.27	227.6	780.2
$\frac{E_b}{N_o}\Big\vert_{\mathrm{dB}}$ $@P_b = 10^{-4}$	3.675	2.9	2.37	1.9-2.08	1.55-1.86
$\frac{E_b}{N_o}\Big\vert_{\mathrm{dB}}$ $@P_b = 10^{-5}$	4.525	3.675	3.075	2.62	2.27
$\frac{E_b}{N_o}\Big\vert_{\mathrm{dB}}$ $@P_b = 10^{-6}$	5.22	4.34	3.7	3.2	2.82
Asymptotic Coding Gain dB	6.53	7.78	8.75	9.54	10.2

Table 3.1: Coding gain of short constraint length TRM codes at different bit error probabilities, in dB.

3.4 TRM2 Codes

In this section, we construct another class of good coset-selecting codes. Unlike TRM1 codes, these codes achieve the maximal possible number of inputs (maximum coding rate, for a given $R(1,m)$ block code), while satisfying the code construction rules of section 3.2.2 with $d_{\mathrm{mf}} = 2$. We will refer to these codes as TRM2 codes.

3.4.1 Code Construction

Consider first a general coset-selecting trellis code with b inputs. Any node in the corresponding primary trellis diagram has 2^b branches departing from it. According to rule 3 for the construction of good coset-selecting codes, these 2^b branches must select distinct coset leaders, $\underline{\lambda}_0, \cdots, \underline{\lambda}_{2^b-1}$.

Suppose there are 2^b coset leaders available, $\underline{\lambda}_0, \cdots, \underline{\lambda}_{2^b-1}$. Then, for each trellis node, there is exactly one outgoing branch with coset leader $\underline{\lambda}_0$, one with $\underline{\lambda}_1$, and so on. Consider the path π_0 consisting of starting in some state S_0 and always selecting the next state such that the sequence of coset leaders selected is $(\underline{\lambda}_0, \underline{\lambda}_0, \cdots)$. There also exists a path π_1, starting at some state S_1, such that the sequence of coset leaders selected is also $(\underline{\lambda}_0, \underline{\lambda}_0, \cdots)$. From rule 3, the paths π_1 and π_0 never merge, which implies that the code is catastrophic.

Hence a coset-selecting code with b input bits must use a set of coset leaders of more than 2^b elements. Since the number of coset leaders is an odd power of 2, a coset-selecting code must have at least 2^{b+1} available coset leaders if b is even, and 2^{b+2} if b is odd. Putting it the other way around, if the RM code has length 2^m (m even), the maximal number of inputs to the coset-selecting trellis code is $b_{\max} = m - 2$. This statement holds for an arbitrary good coset-selecting trellis code (i.e. for any coset-selecting trellis code satisfying the construction rules of section 3.2.2). Note that the value of $b_{\max}$ is significantly larger than that of the TRM1 codes considered earlier.

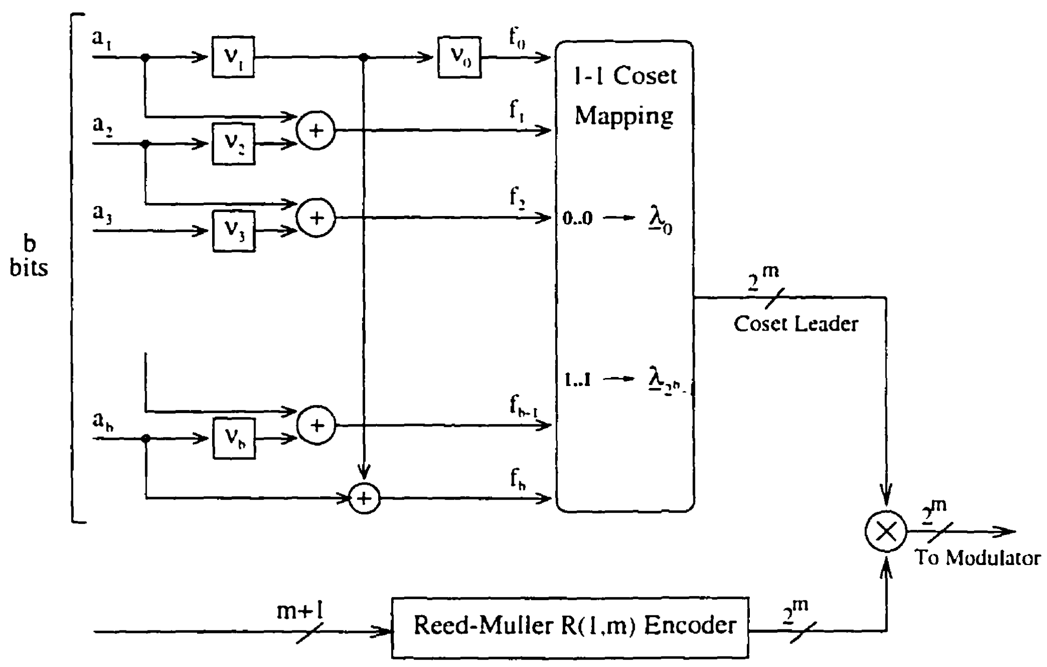


Figure 3.10: Encoder structure of TRM2 codes.

Figure 3.10 shows a feedback free implementation of a TRM2 code. It consists of a convolutional code with $b+1$ delay elements, and $b+1$ output bits which are the indices of the corresponding coset leaders. The state of the convolutional encoder is given by the vector $\underline{\nu} = (\nu_0, \nu_1, \cdots, \nu_b)$, and its b input bits are given by the vector $\underline{a} = (a_1, \cdots, a_b)$. Note that in Figure 3.10, $\underline{a} \in \{0,1\}^b$, $\underline{\nu} \in \{0,1\}^{b+1}$, and $F = \{\underline{f}\} \in \{0,1\}^{b+1}$. On the other hand, the coset leaders and the output of the RM encoder belong to $\{1,-1\}^{2^m}$. The input-output relation of the convolutional encoder is given by

$$\underline{f}(\underline{a}, \underline{\nu}) = (\nu_0, a_1 + \nu_2, \cdots, a_i + \nu_{i+1}, \cdots, a_{b-1} + \nu_b, a_b + \nu_1) \tag{3.49}$$

where addition is understood to be modulo 2. We shall now proceed to show that the coset-selecting code of Figure 3.10 code satisfies all the construction rules of section 3.2.2.

1. Rule 1 is verified trivially because of the one-to-one mapping from the output of the convolutional encoder to the set of coset leaders.

2. Rule 2 is verified since each input bit is fed into a memory element which affects the state of the encoder.

3. Suppose the encoder is in a certain state $\underline{\nu}$, and consider two distinct input vectors $\underline{a}$ and $\underline{a}'$. From (3.49), we have

   $$\underline{f}(\underline{a}, \underline{\nu}) + \underline{f}(\underline{a}', \underline{\nu}) = (0, a_1 + a_1', \cdots, a_i + a_i', \cdots, a_{b-1} + a_{b-1}', a_b + a_b') \tag{3.50}$$

   Hence,

   $$\underline{f}(\underline{a}, \underline{\nu}) = \underline{f}(\underline{a}', \underline{\nu}) \Rightarrow \underline{a} = \underline{a}' \tag{3.51}$$

   Thus any two distinct branches emanating from the same node are mapped into distinct coset leaders.

   Consider now the case of branches merging into the same node. Note that this requires that the corresponding input words be identical (it should be clear from Figure 3.10 that two distinct input words must yield distinct "next states"). Suppose we have two distinct states $\underline{\nu}$ and $\underline{\nu}'$, and some input vector $\underline{a}$. Then

   $$\underline{f}(\underline{a}, \underline{\nu}) + \underline{f}(\underline{a}, \underline{\nu}') = (\nu_0 + \nu_0', \nu_2' + \nu_2, \cdots, \nu_i + \nu_i', \cdots, \nu_b + \nu_b', \nu_1 + \nu_1') \tag{3.52}$$

   Hence,

   $$\underline{f}(\underline{a}, \underline{\nu}') = \underline{f}(\underline{a}, \underline{\nu}) \Rightarrow \underline{\nu}' = \underline{\nu} \tag{3.53}$$

   Thus any two distinct branches merging into the same state are mapped into distinct coset leaders.

4. A convolutional code is non-catastrophic provided the state diagram contains no loop of zero weight other than the self-loop around the zero state [33, pp. 308]. Equivalently, the coset-selecting code of Figure 3.10 is non-catastrophic provided that there exists no infinite-length sequence of inputs $(\underline{a}^{(0)}, \underline{a}^{(1)}, \cdots)$, with a non-zero Hamming weight, which generates an output sequence $(\underline{f}^{(0)}, \underline{f}^{(1)}, \cdots)$ of zero Hamming weight.

   Let us try to construct an arbitrary long sequence of input vectors, $(\underline{a}^{(0)}, \underline{a}^{(1)}, \cdots)$, which generates consistently zero output vectors. Suppose we start off in some state $\underline{\nu}$. We must have $\nu_0 = \nu_1 = 0$; otherwise, a non-zero output vector would be generated after at most 1 delay. Similarly, we must have $a_1^{(i)} = 0$ for all i. If $a_1^{(i)} = 0$, then ν_2 must also always be zero, which implies that $a_2^{(i)} = 0$ for all i. By repeating the same reasoning, we end up with the following condition: in order to have a continuous stream of zero output vectors, the initial state must be zero, and all the input vectors must be zero. Hence the coset-selecting code of Figure 3.10 is non-catastrophic.

Note that the modified free distance has not increased from TRM1 to TRM2 codes. The role of the additional delay element is to provide more states, relaxing the condition that distinct states transitions bear distinct coset leaders, thereby allowing larger values of b for the same m.

3.4.2 Evaluation of the Transfer Function

In order to apply the performance analysis tools that were used for TRM1 codes, we need to derive the modified generalized transfer function $T(D, L, I)$, or at least a sufficient number of coefficients in the long-division expansion of $T(D, L, I)$. We denote by S_i the state specified by the state vector $\underline{\nu}$ such that the base-2 representation of

i is $(\nu_0\nu_1\cdots\nu_b)_2$. Let $S_i^{(j)} \equiv S_{j2^{b-1}+i}$. We define the following vectors of states

$$\begin{aligned}
\underline{U}_0 &= (S_1 \;\cdots\; S_{2^{b-1}-1})^T &&= (S_1^{(0)} \;\cdots\; S^{(0)}_{2^{b-1}-1})^T\\
\underline{U}_1 &= (S_{2^{b-1}} \;\cdots\; S_{2^b-1})^T &&= (S_0^{(1)}\; S_1^{(1)} \;\cdots\; S^{(1)}_{2^{b-1}-1})^T\\
\underline{U}_2 &= (S_{2^b} \;\cdots\; S_{2^b+2^{b-1}-1})^T &&= (S_0^{(2)}\; S_1^{(2)} \;\cdots\; S^{(2)}_{2^{b-1}-1})^T\\
\underline{U}_3 &= (S_{2^b+2^{b-1}} \;\cdots\; S_{2^{b+1}-1})^T &&= (S_0^{(3)}\; S_1^{(3)} \;\cdots\; S^{(3)}_{2^{b-1}-1})^T
\end{aligned} \tag{3.54}$$

We shall denote by U_i the set of states to which the elements of $\underline{U}_i$ belong. Hence the sets $\{S_0\}, U_0, U_1, U_2, U_3$ induce a partition of the set of states. We shall refer to the U_is as superstates. This allows to draw the vectorized state diagram of Figure 3.11.

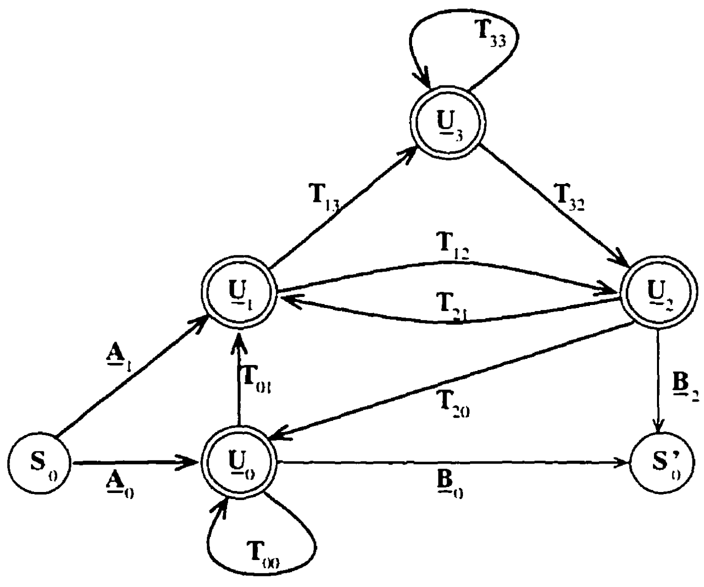


Figure 3.11: Vectorized state diagram for the coset selecting code of a TRM2 code.

In Figure 3.11, the superstates are shown with double circles. When more than one branch links two superstates, this is indicated by a thick arrowed line. The transfer function between two superstates is represented by a matrix.

The state equations in matrix form are then

$$\underline{U}_0 = \underline{A}_0\, S_0 + T_{00}\, \underline{U}_0 + T_{20}\, \underline{U}_2 \tag{3.55}$$

$$\underline{U}_1 = \underline{A}_1\, S_0 + T_{01}\, \underline{U}_0 + T_{21}\, \underline{U}_2 \tag{3.56}$$

$$\underline{U}_2 = T_{12}\, \underline{U}_1 + T_{32}\, \underline{U}_3 \tag{3.57}$$

$$\underline{U}_3 = T_{13}\, \underline{U}_1 + T_{33}\, \underline{U}_3 \tag{3.58}$$

$$S'_0 = \langle \underline{B}_2 \mid \underline{U}_2 \rangle + \langle \underline{B}_0 \mid \underline{U}_0 \rangle \tag{3.59}$$

The vectors $\underline{A}_0$ and $\underline{B}_0$ and the matrices T_{00}, T_{20}, and T_{01} are given explicitly in Appendix B.4.

The solution of the state equations is not trivial, and is practical only for small values of b, typically $b \leq 5$. For larger values of b, it is difficult to compute the transfer function. However, it is still possible to obtain iteratively the path enumerating function $g(j, d, l)$, which gives the number of paths of length l and modified weight d which are generated by an input sequence of Hamming weight j. The improved upper bound on the performance of TRM1 codes used this approach, not because of the unavailability of the transfer function, but because of the greater accuracy that was obtained. The unavailability of the transfer function deprives us only of a looser upper bound, which would have the sole advantage of being in closed form.

The state equations (3.55–3.59) can be written as

$$\underline{x} = \mathcal{A}\,\underline{x} + \underline{\mathcal{B}}\,S_0 \tag{3.60}$$

where $\underline{x}^T = (S_0', \underline{U}_0^T, \underline{U}_1^T, \underline{U}_2^T, \underline{U}_3)^T$, and the matrices $\mathcal{A}$ and $\mathcal{B}$ are in accordance with (3.55–3.59). As shown in [40, Section 6.4], the formal solution $\underline{x} = (I - \mathcal{A})^{-1} S_0$ leads to the conventional transfer function, while the equivalent matrix series solution

$$\underline{x} \;=\; (1 + \mathcal{A} + \mathcal{A}^2 + \mathcal{A}^3 + \cdots)S_0 \tag{3.61}$$

provides an algorithm for the computation of the transfer function. Knowledge of the relation between $(1 + \mathcal{A} + \cdots \mathcal{A}^n)$ and $(1 + \mathcal{A} + \cdots \mathcal{A}^{n+1})$ yields an iterative method to compute the transfer function.

The iterative computation of the path enumerating function is better understood by looking at the trellis diagram of TRM2 coset-selecting codes, illustrated in Figure 3.12. Let $\underline{\mathcal{T}}_v^{(n)}$ denote the vector of transfer functions between $S_0^{(0)}$ and $\underline{U}_v^{(n)}$, and let $\mathcal{T}_s^{(n)}$ denote the transfer function between $S_0^{(0)}$ and $S_0^{(n)}$. We then have

$$\underline{\mathcal{T}}_3^{(n+1)} \;=\; T_{33}\underline{\mathcal{T}}_3^{(n)} + T_{13}\underline{\mathcal{T}}_1^{(n)} \tag{3.62}$$

$$\underline{\mathcal{T}}_2^{(n+1)} \;=\; T_{32}\underline{\mathcal{T}}_3^{(n)} + T_{12}\underline{\mathcal{T}}_1^{(n)} \tag{3.63}$$

$$\underline{\mathcal{T}}_1^{(n+1)} \;=\; T_{21}\underline{\mathcal{T}}_2^{(n)} + T_{01}\underline{\mathcal{T}}_0^{(n)} \tag{3.64}$$

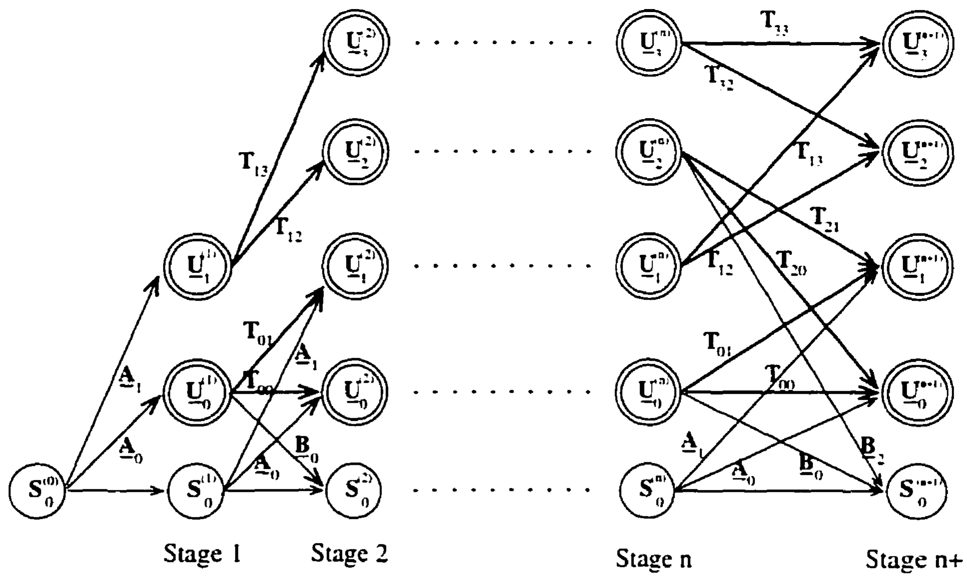


Figure 3.12: Trellis diagram of a TRM2 coset-selecting trellis code.

$$\mathcal{T}_0^{(n+1)} = T_{20}\mathcal{T}_2^{(n)} + T_{00}\mathcal{T}_0^{(n)} \tag{3.65}$$

$$\mathcal{T}_s^{(n+1)} = \langle \underline{B}_2 \mid \mathcal{T}_2^{(n+1)} \rangle + \langle \underline{B}_0 \mid \mathcal{T}_0^{(n+1)} \rangle \tag{3.66}$$

which is the iterative relation to compute the transfer function. $T(D, L, I) = \lim_{n\to\infty} \mathcal{T}_s^{(n)}$.

For finite n, $\mathcal{T}_s^{(n)}$ is a polynomial in D, I, and L, where the exponent of L is always n (by inspection from the trellis). Moreover, the coefficient of each term in $D^d I^j L^n$ is the number of trellis paths with a modified weight d, length n and generated by an input sequence of weight j. Mathematically

$$\mathcal{T}_s^{(n)}(D, L, I) = \sum_{j=0}^{bn}\sum_{d=2}^{n} g(j, d, n) D^d I^j L^n \tag{3.67}$$

$$\Rightarrow \mathcal{T}_s^{(n)}(D, L, 1) = \sum_{d=2}^{n} \underbrace{\sum_{j=0}^{bn} g(j, d, n)}_{c(d,n)} D^d L^n \tag{3.68}$$

and

$$\left.\frac{\partial \mathcal{T}_s^{(n)}(D, L, I)}{\partial I}\right|_{I=1} = \sum_{d=2}^{n} \underbrace{\left(\sum_{j=0}^{bn} j g(j, d, n)\right)}_{h(d,n)} D^d L^n \tag{3.69}$$

Hence, in principle, one could obtain $h(d, n)$ and $c(d, n)$ for an arbitrary range of values of $n = 2 \cdots N$. In practice, a relatively small N provides a sufficient accuracy in computing the upper bound on the bit error probability. This is because for large values of n, good codes have $c(d,n) = h(d,n) = 0$ for $d \ll n$, which means that longer diverging paths have a larger modified distance and hence a smaller probability of being chosen. If the signal to noise ratio is not too small, the probability of choosing erroneous long paths in the trellis is negligible when compared with the probability of making an error on shorter paths, despite the fact that the number of paths of a given length n increases rapidly with n. The computation of $\mathcal{T}_s^{(n)}(D,L,I)$, of its derivative, and of $c(d,n)$ and $h(d,n)$ for all $n \leq N$ is easily done on a computer.

3.4.3 Performance Evaluation

In this section, we evaluate the performance of TRM2 codes as a function of signal to noise ratio. We first argue that the closed form upper bounds are not tight enough for the signal to noise ratio range of interest. We then proceed to use the truncated path enumeration bound to estimate the bit error probability of TRM2 codes.

Closed Form Upper Bound

The closed form upper bound of (3.36) is meaningful provided the infinite sums of (B.12–B.16) converge, which happens at sufficiently large signal-to-noise ratio. This is insured if the variable D, evaluated at $D = e^{-w_2 E_s/N_o}$, is smaller than the smallest root of the denominator of $T(D,L,I)|_{D=e^{-w_2 E_s/N_o},\ I=1,L=2^{m-1}}$. Since $T(D,L,I)$ is a ratio of polynomials in D,L,I, its poles are also the poles of its partial derivatives with respect to D, L, or I. For example, with $b = 2$ and $m = 6$, numerical evaluation shows that the smallest pole of $T(D,L,I)|_{I=1,L=128} = 4.731\ 10^{-7}$. This means for a $b = 2$, $m = 6$ TRM2 code, the upper bound of (3.36) is useful for $E_b/N_o > 5.7$dB, which is well outside the range of interest. As m increases, the minimum required E_b/N_o to ensure the convergence of (3.36) increases too. This is a consequence of the large number of parallel branches in the full TRM trellis. It also enforces the observation that closed form upper bounds based on the transfer function of the coset selecting code does not lead to a useful measure of the performance of TRM codes.

Truncated Path Enumeration Approximation

This method was used in the evaluation of an upper bound on the bit error probability of TRM1 codes. By evaluating (3.33–3.35) for a sufficiently large of terms in the infinite sums. an approximate upper bound which is tighter than the transfer function bound is obtained. Therefore. the truncated path-enumeration approximation is the only indicator we use in the performance evaluation of TRM2 codes. and we will somewhat loosely refer to it as an upper bound. Figure 3.13 shows the upper and lower bounds for all TRM2 codes with $m = 6, 8, 10, 12$. The tightness of the bounds increases with signal to noise ratio (as expected since they are asymptotically tight). but decreases with the number of input bits to the coset-selecting code. since larger values of b implies a large number of paths. The looseness of (3.48) at low SNR is due to the large number of paths in the full TRM trellis. since we are in fact using a union bound over all paths of the full TRM trellis. Tables 3.2-3.4 summarize the

TRM Code: (m, b)	**(6,4)**	**(6,3)**	**(6, 2)**	**(8,6)**	**(8,5)**	**(8,4)**	**(8,3)**	**(8,2)**
Bandwidth Expansion	5.8	6.4	7.11	17.1	18.3	19.7	21.3	23.3
$\frac{E_b}{N_0}\Big\vert_{dB}$ $@P_b = 10^{-4}$	2.7-2.95	3.2	3.675	1.8-2.2	2.12-2.35	2.5-2.65	2.875	3.3
$\frac{E_b}{N_0}\Big\vert_{dB}$ $@P_b = 10^{-5}$	3.6	4.15	4.52	2.6-2.7	2.97	3.3	3.67	4.07
$\frac{E_b}{N_0}\Big\vert_{dB}$ $@P_b = 10^{-6}$	4.3	4.75	5.27	3.3	3.63	3.96	4.33	4.7
ACG (dB)	5.5	5.0	4.5	7.5	7.0	6.5	6.0	5.5

Table 3.2: Required E_b/N_0 for TRM2 codes at different bit error probabilities. and asymptotic coding gain. $(m = 6, 8)$

conclusions that can be drawn from Figure 3.13. by displaying the required E_b/N_0 to achieve bit error probabilities of 10^{-4}. 10^{-5}. and 10^{-6}. Whenever the difference between the upper and lower bound of Figure 3.13 is less than 0.1 dB. we consider the average of these two values as a reliable estimate. thus ensuring that the maximal error is never greater than 0.05 dB. Otherwise. we list the corresponding E_b/N_0 range (lower and upper bound). Bit error probabilities of 10^{-3} and larger are not considered because the bounds are not tight enough in this region.

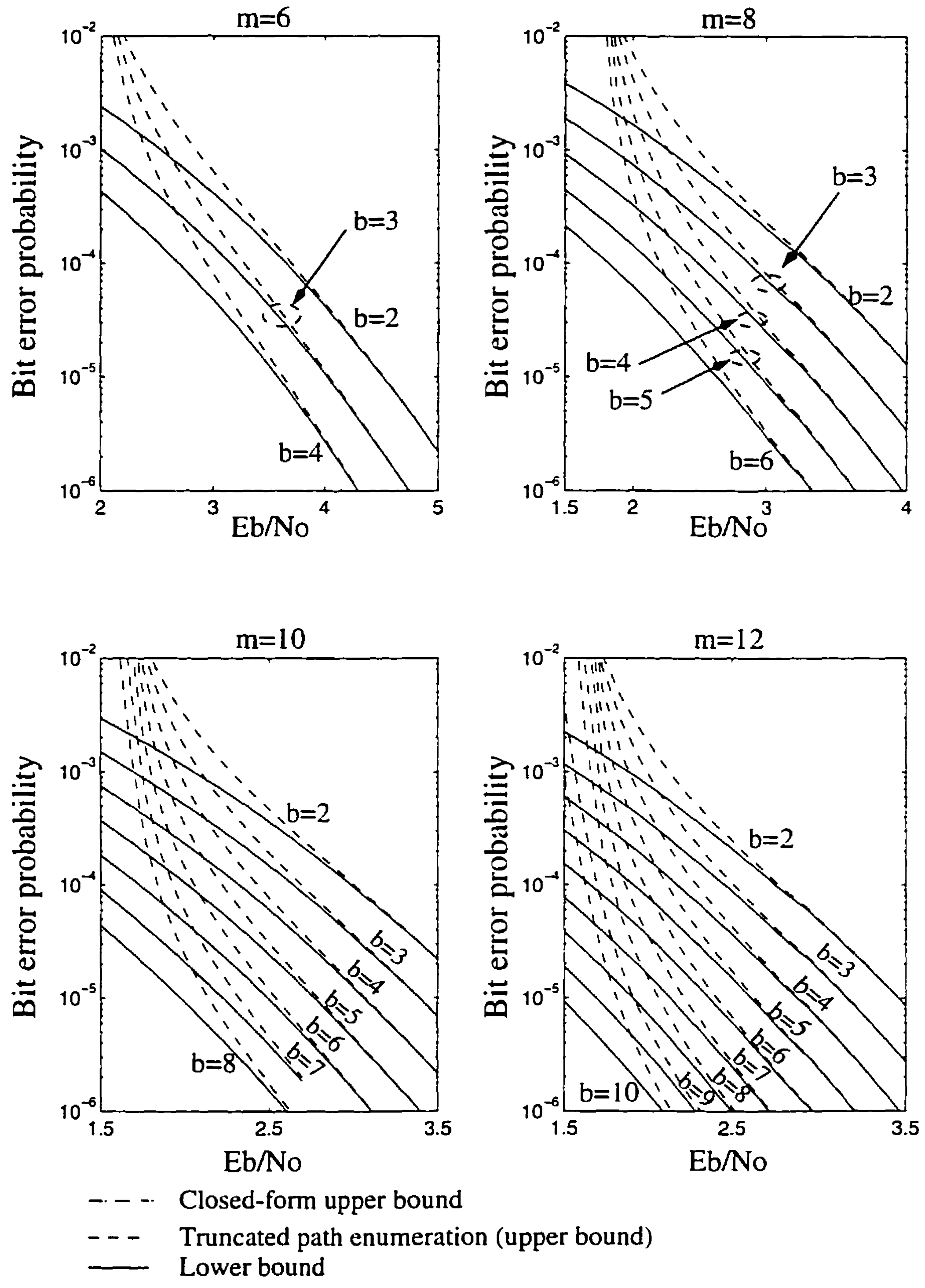


Figure 3.13: Upper and lower bounds on TRM2 codes.

TRM Code: $m = 10, b$:	**8**	**7**	**6**	**5**	**4**	**3**	**2**
Bandwidth Expansion	53.9	56.9	60.2	64	68.3	73.1	78.8
$\frac{E_b}{N_o}\Big\vert_{dB}$ $@P_b = 10^{-4}$	1.2-1.84	1.46-1.93	1.74-2.06	2.02-2.22	2.38	2.67	3.0
$\frac{E_b}{N_o}\Big\vert_{dB}$ $@P_b = 10^{-5}$	1.97-2.14	2.28	2.52	2.78	3.08	3.39	3.74
$\frac{E_b}{N_o}\Big\vert_{dB}$ $@P_b = 10^{-6}$	2.61-2.63	2.8	3.12	3.4	3.7	4.0	4.35
ACG (dB)	9.5	9.0	8.5	8.0	7.5	7.0	6.5

Table 3.3: Required E_b/N_0 for TRM2 codes at different bit error probabilities. and asymptotic coding gain. ($m = 10$)

$m = 12, b$:	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**
BW Expansion	178.1	186.2	195	204.8	215.6	227.6	240.9	256	273.1
$\frac{E_b}{N_o}\Big\vert_{dB}$ $@P_b = 10^{-4}$	0.75-1.6	0.96-1.7	1.2-1.8	1.4-1.86	1.65-1.97	1.9-2.1	2.23	2.49	2.77
$\frac{E_b}{N_o}\Big\vert_{dB}$ $@P_b = 10^{-5}$	1.5-1.8	1.7-1.9	1.9-2.06	2.18	2.39	2.63	2.89	3.16	3.46
$\frac{E_b}{N_o}\Big\vert_{dB}$ $@P_b = 10^{-6}$	2.11	2.3	2.5	2.73	2.96	3.21	3.47	3.74	4.04
ACG (dB)	11.5	11.0	10.5	10.0	9.5	9.0	8.5	8.0	7.5

Table 3.4: Required E_b/N_0 for TRM2 codes at different bit error probabilities. and asymptotic coding gain. ($m = 12$)

The required E_b/N_0 estimates for most of the codes of Figure 3.13 are accurate at bit error probabilities of 10^{-5} or lower. For some codes, the estimates are accurate at even higher bit error probabilities. However, for probabilities of error of the order of 10^{-3} or higher, we need to resort to simulations to estimate with precision the corresponding E_b/N_0.

3.5 Conclusions

In this chapter, we have introduced Trellis/Reed-Muller codes, a new class of very low rate codes which is based on the combination of trellis codes and first-order Reed-Muller codes. We derived a set of rules on the construction of the trellis code. These rules lead to the development of two main families of Trellis/Reed-Muller codes: the TRM1 codes which use very simple trellis codes, and the TRM2 codes which use a more complex trellis code in exchange for a higher achievable coding rate, and a better performance.

We have also explored in this chapter analytical methods of estimating the bit error probability of TRM codes as a function of the bit energy to noise power spectral density ratio (E_b/N_0). We used these methods in the estimation of the bit error probability of both the TRM1 and TRM2 codes. In the next chapter, we will thoroughly compare the TRM codes with the other known very low rate codes mentioned in the early sections of the current chapter.

Chapter 4

Comparison of Very Low Rate Codes

This chapter presents a thorough comparison of TRM codes with the other very low rate codes examined in this thesis. The comparisons are made on the basis of performance (required signal to noise ratio to achieve a given bit error probability), coding rate, and decoding complexity.

4.1 Analytical Methods

In this section, we look at some analytical methods of assessing the performance of all the very low rate codes considered in this thesis. For most codes, no exact expression of the bit error probability is known. However, one can derive tight bounds which can accurately approximate the bit error probability of the codes, especially at high signal to noise ratio.

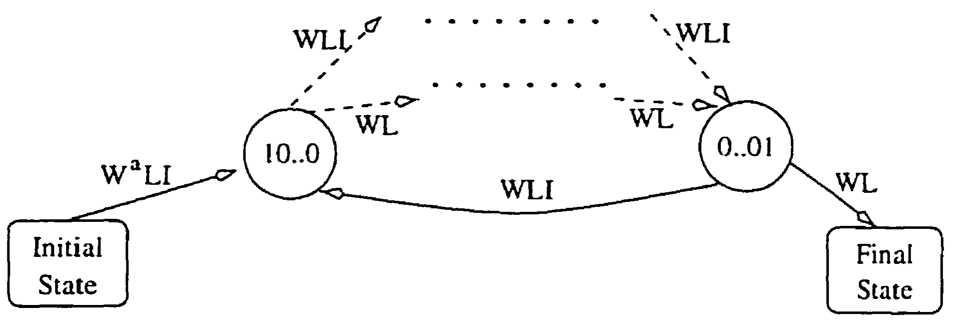
The construction of the orthogonal and superorthogonal convolutional codes are attributed to Viterbi [12, 27], whereas that of the biorthogonal convolutional codes is attributed to Rikkinen [26]. A performance analysis of the orthogonal and superorthogonal convolutional codes is given in [19] and [27] respectively. The author relies on the classical upper bounding technique which uses the generating function of the code and the approximation of the complementary error function by an exponential. The resulting bound is known to be asymptotically tight.

The derivation of the generating function of orthogonal, and superorthogonal convolutional codes can be found in [27]. We assume that all the codes have a rate $1/2^K$, and adapt the results of [27] to obtain

$$T_{\text{Ortho}}(W,I) = \frac{IW^K(1-W)}{1-(1+I)W+IW^K} \tag{4.1}$$

$$T_{\text{Suportho}}(W,I) = \frac{IW^{K+4}(1-W)}{1-(1+I)W-IW^K+2IW^{K+1}} \tag{4.2}$$

where $W = D^{2^{K-1}}$. Recall that the exponent of D in each term of the series expansion of the generating function gives the Hamming weight of the corresponding path. In



Biorthogonal Convolutional Code: a=2, $W=D^{2^{K-1}}$ Rate = $1/2^{K-1}$
Orthogonal Convolutional Code: a=1, $W=D^{2^K}$ Rate = $1/2^K$

Figure 4.1: State diagram of an orthogonal and biorthogonal convolutional code.

order to obtain the generating function of the biorthogonal convolutional code, we use the great similarity between orthogonal and biorthogonal convolutional codes. Consider the state diagram of a biorthogonal convolutional code of rate $1/2^K$, depicted in Figure 4.1. As emphasized in Figure 4.1, this diagram is also the state diagram of an orthogonal convolutional code of rate $1/2^{K+1}$. The only difference lies in the labeling of the first branch, and in the expression of W in terms of D. Therefore,

$$T^{(K)}_{\text{Biortho}}(W,I) = WT^{(K+1)}_{\text{Ortho}}(W,I) \tag{4.3}$$

$$= \frac{IW^{K+2}(1-W)}{1-(1+I)W+IW^{K+1}} \tag{4.4}$$

where $W = D^{2^{K-1}}$.

The closed form upper bound on the bit error probabilities of a convolutional code is given by [19, chap. 4]

$$P_b < Q\left(\sqrt{\frac{2d_{\text{free}}E_s}{N_0}}\right) e^{\frac{d_{\text{free}}E_s}{N_0}} \left.\frac{\partial T(D,I)}{\partial I}\right|_{I=1,D=e^{-\frac{E_s}{N_0}}} \tag{4.5}$$

For the rate $1/2^K$ codes under analysis, $\frac{E_s}{N_0} = \frac{E_b}{2^K N_0}$, and we evaluate the transfer function at $W = D^{2^{K-1}} = e^{-\frac{E_b}{2N_0}}$. Moreover, the product $\frac{d_{\text{free}}E_s}{N_0}$ is equal to $\frac{KE_b}{2N_0}$ for orthogonal convolutional codes, $\frac{(K+2)E_b}{2N_0}$ for biorthogonal convolutional codes, and $\frac{(K+4)E_b}{2N_0}$ for superorthogonal convolutional codes. Thus (4.5) becomes

$$P_b < Q\left(\sqrt{\frac{2d_{\text{free}}E_s}{N_0}}\right) e^{\frac{d_{\text{free}}E_s}{N_0}} \left.\frac{\partial T(W,I)}{\partial I}\right|_{I=1,W=e^{-\frac{E_b}{2N_0}}} \tag{4.6}$$

We can now use the generating functions to obtain

$$P_{b,\text{Ortho}} < Q\left(\sqrt{\frac{KE_b}{N_0}}\right) e^{\frac{KE_b}{2N_0}} \left.\frac{W^K(1-W)^2}{(1-2W+W^K)^2}\right|_{W=e^{-\frac{E_b}{2N_0}}} \tag{4.7}$$

$$P_{b,\text{Biortho}} < Q\left(\sqrt{\frac{(K+2)E_b}{N_0}}\right) e^{\frac{(K+2)E_b}{2N_0}} \left.\frac{W^{K+2}(1-W)^2}{(1-2W+W^{K+1})^2}\right|_{W=e^{-\frac{E_b}{2N_0}}} \tag{4.8}$$

$$P_{b,\text{Suportho}} < Q\left(\sqrt{\frac{(K+4)E_b}{N_0}}\right) e^{\frac{(K+4)E_b}{2N_0}} \left.\frac{W^{K+4}(1-W)^2}{(1-2W-W^K+2W^{K+1})^2}\right|_{W=e^{-\frac{E_b}{2N_0}}} \tag{4.9}$$

At moderate to high signal to noise ratio, i.e. whenever $W^K \ll W$, the denominators of all three bounds are the same. In this situation, a superorthogonal convolutional code of rate $1/2^K$ has the same performance as a biorthogonal convolutional code of rate $1/2^{K+2}$ and of an orthogonal convolutional code of rate $1/2^{K+4}$. Naturally, this observation fails at low signal to noise ratio, but it is much stronger than a comparison based on asymptotic coding gain. Actually, the series expansion of all three bounds yield the same first K coefficients, which shows the great similarity in the distance profiles.

As mentioned in [26], the number of non-zero data bits on a diverging path at a Hamming distance $d_{\text{free}} + n2^{K-1}$ from the zero path is given by

$$\beta_{d_{\text{free}}+n2^{K-1}} = 2^n + (n-1)2^{(n-2)} \qquad n = 1, \cdots, K-1 \tag{4.10}$$

and $\beta_0 = 1$. Although (4.10) is given for orthogonal and biorthogonal convolutional codes, it also holds for superorthogonal convolutional codes. This can be checked, for example, by performing a long division of the right hand sides of (4.7), (4.8), and (4.9). These coefficients are extremely helpful in computing a much tighter upper bound on the bit error probability. Indeed, given the coefficients β_n, the union bound on the bit error probability of a convolutional code is given by

$$P_b \quad < \quad \sum_{n=d_{\text{free}}}^{\infty} \beta_n Q\left(\sqrt{\frac{nE_s}{N_0}}\right) \qquad (4.11)$$

The closed form upper bound is obtained by upper bounding the $Q(\cdot)$ function by an exponential, such that (4.11) is recognized as the series expansion of the derivative of the generating function, evaluated at $D = e^{-E_s/N_0}$. However, as seen earlier with TRM codes, a tighter approximate bound is obtained by simply truncating the sum in (4.11) to a sufficiently large number of terms. If the β_n's are known, the computation is straightforward. In our case, the K first non-zero values of β_n are given by (4.10).

Finally, a simple lower bound on the bit error probability is given by the probability that the minimum distance path is decoded. Since for the codes under analysis, the minimum distance path is unique, and is generated by a single bit, we have

$$P_b \quad > \quad Q\left(\sqrt{2\frac{d_{\text{free}}E_s}{N_0}}\right) \qquad (4.12)$$

Figure 4.2 systematically displays the above-mentioned bounds for OBS codes of rates 1/8 to 1/2048. These curves show that it is difficult to get a reliable estimate on the bit error probability for low signal to noise ratio. The tightness of the bounds increases with signal to noise ratio and with the coding rate. Hence for low coding rates, these bounds are tight only at very low bit error probabilities. However, it is seen from Figure 4.2 that we can reliably estimate the bit error probability of some codes starting at a given signal to noise ratio threshold.

It is shown in [26] that the path-enumerating bound is very tight for orthogonal and biorthogonal convolutional codes of rate greater or equal to 1/128 and bit error probabilities less or equal to 10^{-3}. However, no such guarantees are given for lower coding rates, higher bit error probabilities, or for the superorthogonal convolutional codes. In order to assess when and which of the bounds of Figure 4.2 can be used to reliably estimate the code performance, we need to resort to simulations.

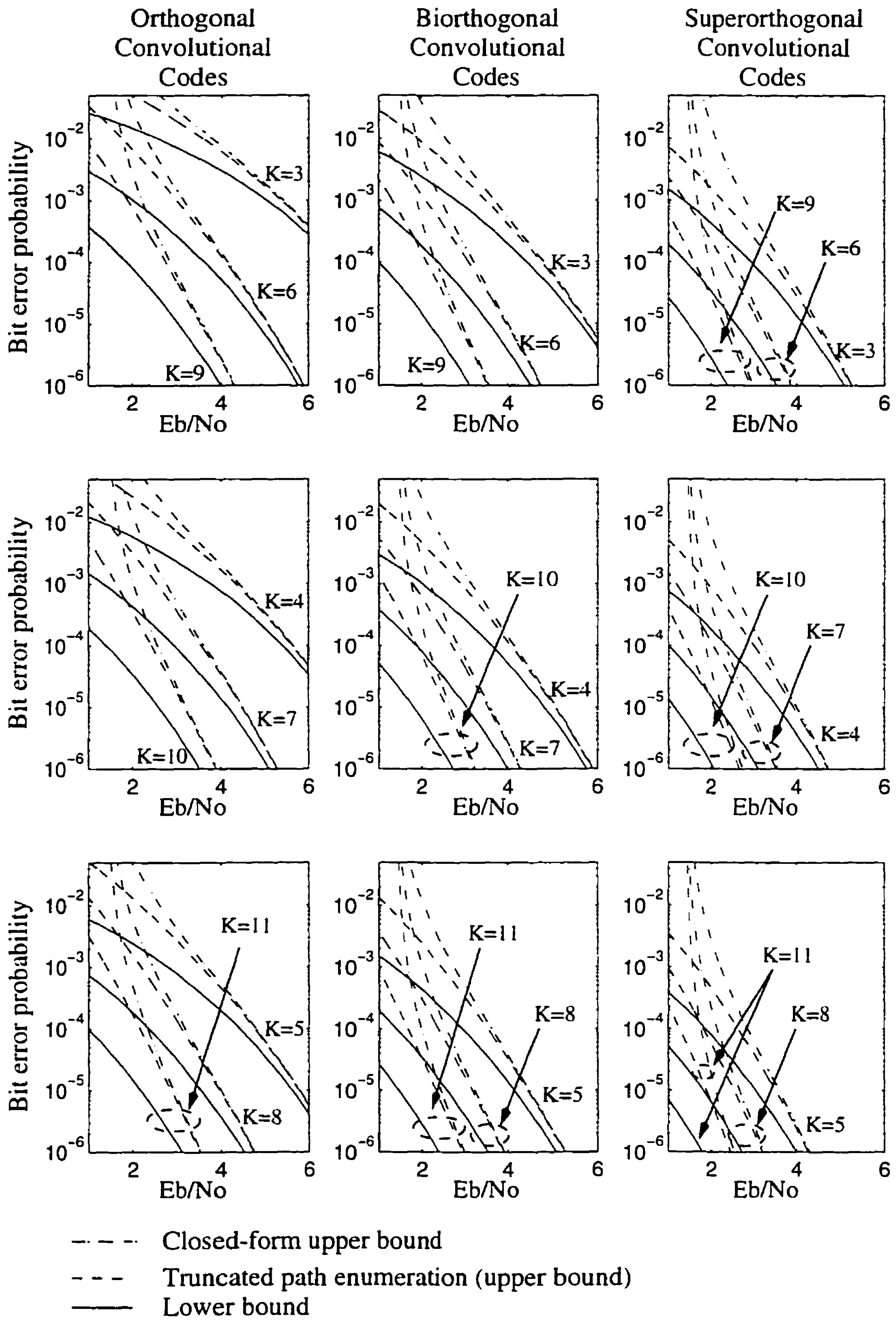


Figure 4.2: Bounds on the bit error probability of orthogonal, biorthogonal, and superorthogonal convolutional (OBS) codes of rate $1/2^K$.

Another conclusion that can be drawn from Figure 4.2 is that the closed-form upper bound can be much looser than the path-enumerating bound of (4.11). For example, the two bounds differ by almost 1 dB at 10^{-3} for the superorthogonal convolutional code of rate 1/256. For this reason, we will only use the upper bound based on (4.11) in all future references to upper bounds on the bit error probability of orthogonal, biorthogonal, and superorthogonal convolutional codes.

The last code under consideration that was not yet mentioned in this section is the IS-95 uplink code. The natural method of evaluating analytically its performance is to use a transfer function method (or better, a path-enumerating method), which is modified to take into account the inner Hadamard code. Basically, we consider the whole code as a 2-bit input, 64-symbol output convolutional code of constraint length 9. However, due to the cumbersome expressions that result from trying to evaluate the transfer function of a constraint length 9 convolutional code, we will limit ourselves to simulations to assess the performance of the IS-95 uplink code. Since there is only one such code (as opposed to a family of codes), we can afford the extra computational effort to reliably estimate the code performance even at relatively low bit error probabilities.

4.2 Simulations Results over the AWGN Channel

All the very low rate codes under consideration were simulated on an AWGN channel. The simulator was written in C, and its output format is compatible with Matlab. In this section, we display the simulation results as the bit error probability of a given code versus the corresponding E_b/N_0.

4.2.1 Procedure

We estimate the bit error probability at a given E_b/N_0 by running the simulator for a certain number of input bits, and taking the ratio of the number of bit errors over the number of transmitted bits. By repeating the experience with different seeds to the random number generator, we can obtain several estimates of the same bit error probability. In all the curves presented in this section. the data points are obtained by averaging ten simulation points. This allows us to compute the standard deviation of the estimate. thereby determining a confidence interval. Note that the number of transmitted bits may vary from one data point to another. since lower bit error rates require longer data streams for an accurate estimate. However. the ten simulation points that are used to obtain one data point must be obtained from the same number of transmitted data bits: otherwise. the derived statistics. namely the standard deviation. are meaningless.

The curves of bit error probability estimates are plotted with Matlab on a logarithmic scale. The data points are clearly identified. together with the corresponding standard deviation. The curves are obtained from a cubic spline data interpolation.

4.2.2 IS-95 Uplink Code

In the IS-95 standard. the error control code used in the uplink consists of a rate 1/3. constraint length 9. convolutional code. followed by a Hadamard (64. 6) encoder. In other words. each two output symbols of the convolutional encoder are encoded into a 64-bit orthogonal codeword by the Hadamard encoder. The resulting codeword is further spread four-fold by a PN sequence. Hence this can be viewed as a rate 1/3 code. followed by a rate 6/64 code. followed by a rate 1/4 repetition code (the latter providing no coding gain). As shown earlier. such a code has an asymptotic coding gain of 6 dB.

The rate 1/3 constraint length 9 convolutional code used in the IS-95 standard has generating polynomials (557. 633. 711) in octal notation. Its free distance is 18 [33] which yields an asymptotic coding gain of approximately 7.8 dB. Although the convolutional code is almost 1 dB better than the IS-95 code at very large signal to noise ratio, Figure 4.3 shows that for probability of errors of interest. the IS-95 code offers the best performance. Actually. the two curves meet for $E_b/N_0 \approx 4.2$ dB for a bit error probability of about $3\,10^{-7}$.

4.2.3 OBS Codes

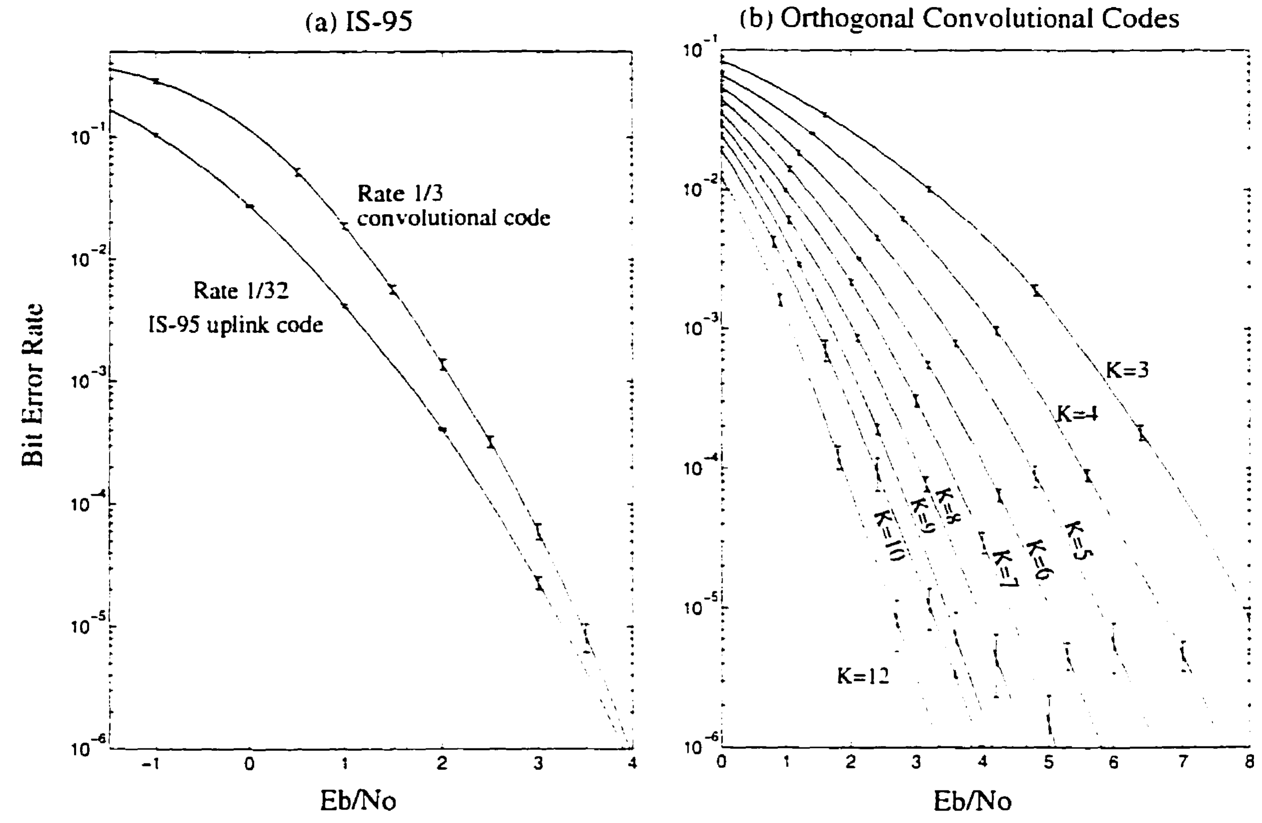


Figure 4.3: (a) Bit error probability of the IS-95 uplink code and its rate 1/3, constraint length 9, convolutional code. (b) Bit error probability of orthogonal convolutional codes of rate $1/2^K$, $K = 3 \cdots 10, 12$

Figures 4.3, and 4.4 show the simulation results for all the OBS codes of interest. The coding rates under examination range from 1/4 to 1/4096. It can be verified that the bit error probability of different codes of the same rate are for all practical purposes shifted versions of each other. This observation confirms the earlier prediction based on the generating functions. There is not much more about these curves that needs to be further emphasized. However, these are the performance curves that will be used later for a comparison between different coding methods.

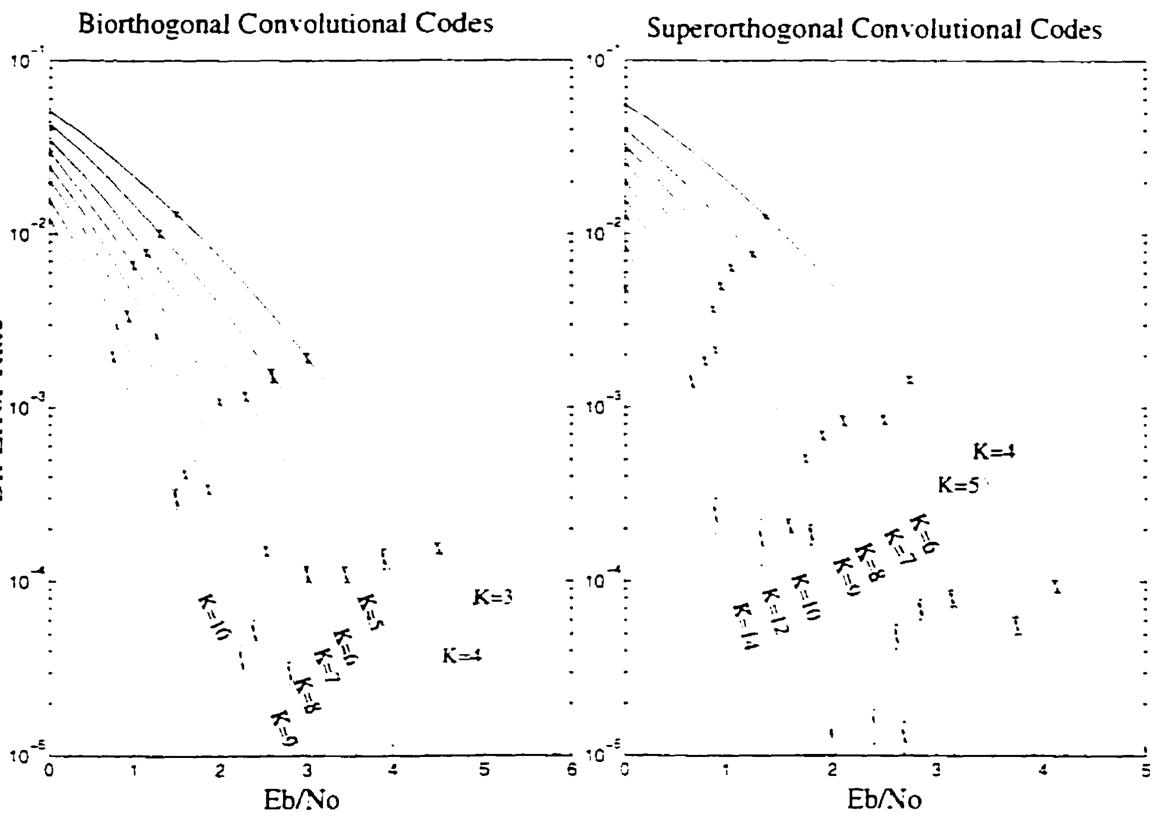


Figure 4.4: Bit error probability of biorthogonal and superorthogonal convolutional codes of rate $1/2^K$. for various values of K. over the AWGN channel.

4.2.4 TRM Codes

The estimated bit error probability of TRM1 codes. obtained from simulations. are shown in Figure 4.5 (a). for $m = 6, 8, 10$ and 12. This corresponds to coding rates of 9/64. 12/256. 15/1024. and 18/4096 respectively. Equivalently. these TRM1 codes spread the bandwidth by factors of about 7.1. 21.3. 68.3. and 227.6 respectively.

Figure 4.5 (b) shows the estimated bit error probability for TRM2 codes with $m = 6$ and $m = 8$. The corresponding coding rates are 9/64. 10/64. 11/64. 11/256. 12/256. 13/256. 14/256. and 15/256. It is seen that the larger m and b are. the better is the performance. Unlike other families of low rate codes. it is not necessary to decrease the coding rate to improve the performance. since the rate is an increasing function of b.

The estimated bit error probability for TRM2 codes with $m = 10$ and $m = 12$ is

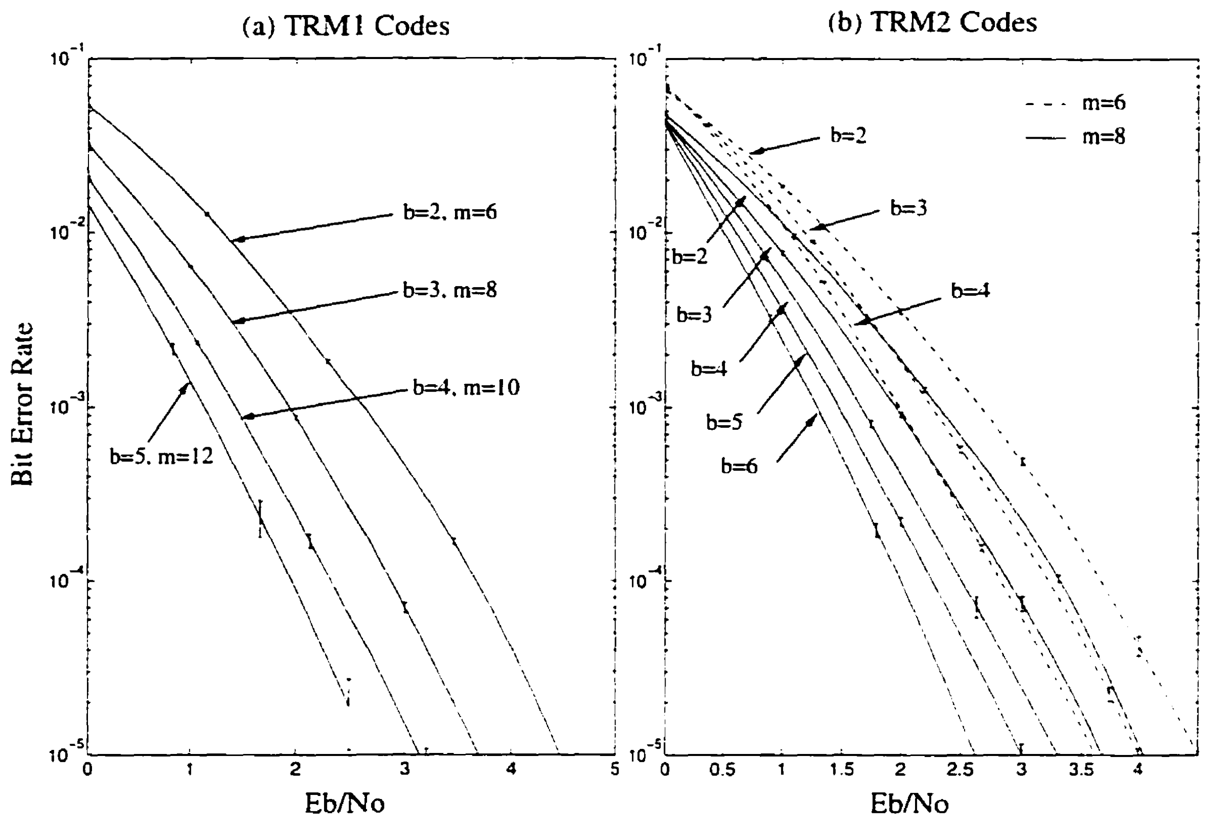


Figure 4.5: (a) Bit error probability of TRM1 codes with $m = 6, 8, 10, 12$. (b) Bit error probability of TRM2 codes with $m = 6$ and 8.

shown in Figure 4.6. For $m = 10$, the coding rates are 13/1024, 12/1024, .. 19/1024. For $m = 12$, the coding rates considered are 15/4096, 16/4096, up to 22/4096. We can verify again the performance improvement with increasing m or b.

We also use the simulation results to verify that the average bit error rate estimate is dominated by the bit error rate of the $m + 1$ bits fed to the first-order Reed-Muller encoder, and that the b input bits to the coset-selecting trellis code are better protected than the remaining $m+1$. This is a confirmation of the discussion of section 3.2.4. In Appendix C, we give the bit error rates for both "types" of input bits, and the corresponding average bit error rate data that is used in some curves of Figures 4.5 and 4.6.

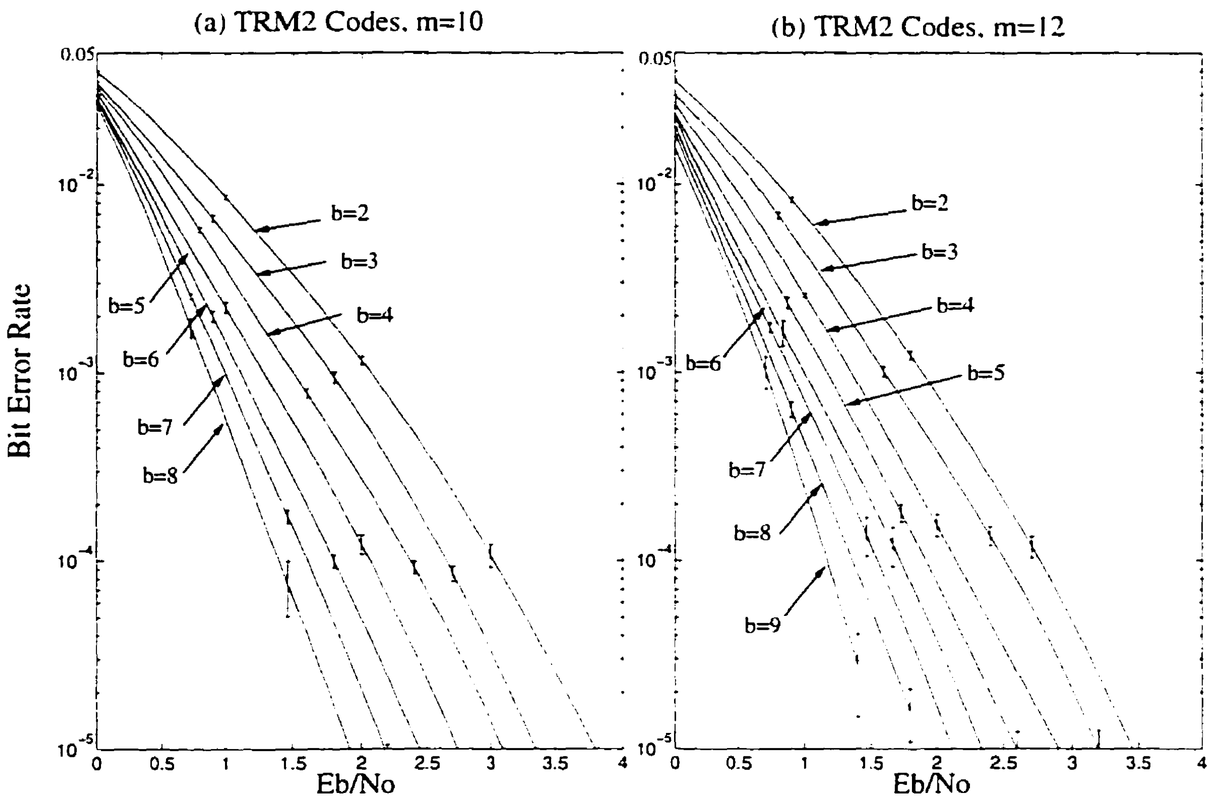


Figure 4.6: Bit error probability of TRM2 codes with $m = 10$ and 12, over the AWGN channel.

4.3 Performance Comparison of Different Coding Scheme

In this section, we collect the fruits of our labor. We use the performance analyses and simulation results to compare the different coding schemes. This is not a simple task. Indeed, a fair comparison must compare codes of the same rate and comparable complexity. However, because of their different structures, it is not always possible to have codes of different families with exactly the same rate. Moreover, the definition of complexity opens up a new discussion, and usually leads to more questions than it provides answers.

We will ignore the issue of complexity for the purpose of this section, and will strive to compare in a fair manner the different codes based on the bit error probability.

In Figure 4.7, we compare all the codes under study on the basis of the required

E_b/N_0 to achieve a bit error probability of 10^{-3}. Each code is represented by a point: the abscissa is the required E_b/N_0, and the ordinate is the code rate. For the same rate, it is the code with the smallest required E_b/N_0 value that is the best, whereas for the same required E_b/N_0 value, it is the code with the largest rate that is best. Hence the better codes are those in the top left corner of the diagram, since they combine relatively large coding rates with good performance.

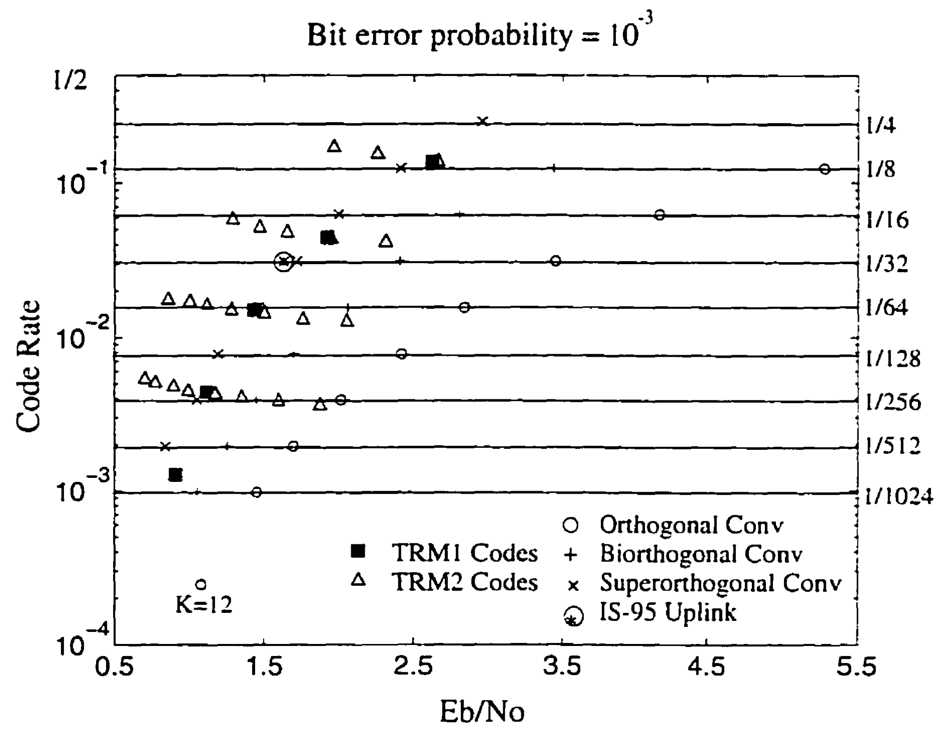


Figure 4.7: Comparison based on coding rate and required E_b/N_0 to achieve a bit error probability of 10^{-3}.

In Figure 4.7, the performance hierarchy of OBS codes is very clear, since the rates of these codes match exactly. Hence, as far as rate and performance is concerned, the superorthogonal convolutional code constantly and significantly outperforms the orthogonal and biorthogonal convolutional codes. We also notice that the IS-95 uplink code is only slightly better than the superorthogonal convolutional code of the same rate.

The performance of TRM1 codes on the other hand is comparable to that of the superorthogonal convolutional codes except at very low rate (less than 1/512) and

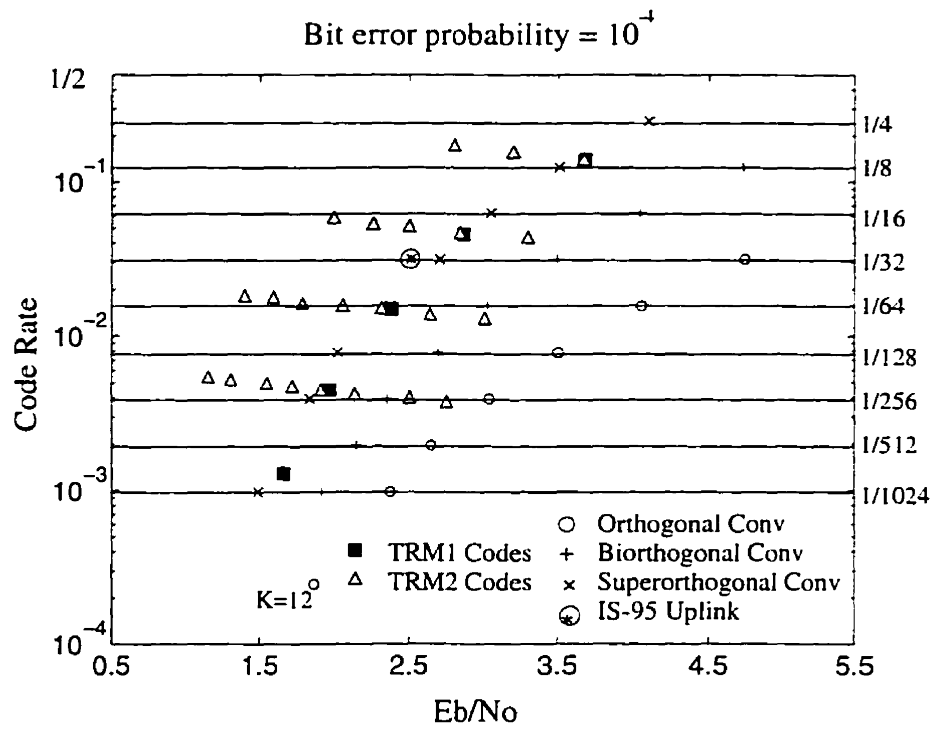


Figure 4.8: Comparison based on coding rate and required E_b/N_0 to achieve a bit error probability of 10^{-4}.

at relatively large rate (greater than 1/10), where the superorthogonal convolutional codes are slightly better.

The important observation that follows from Figure 4.7 is the superior performance of TRM2 codes. Indeed, for all rates of interest, there exists a TRM2 code which outperforms all other codes under study. In Figure 4.7, for every point representing a code of rate less than 1/5, there exists a TRM2 code which is both left and above of that point. In other words, for any non-TRM2 code, there exists a TRM2 code of higher rate and better performance (where performance is here measured by the required E_b/N_0 to achieve a bit error probability of 10^{-3})

In Figures 4.8 and 4.9, we repeat the comparison of Figure 4.7 for bit error probabilities of 10^{-4} and 10^{-5}. A careful examination of these comparisons shows that the general pattern of Figure 4.7 is repeated in Figures 4.8 and 4.9. Hence the relative performances of the codes under study are consistant over the range of error rates of interest. This consolidates the superiority of TRM2 codes in terms of providing the

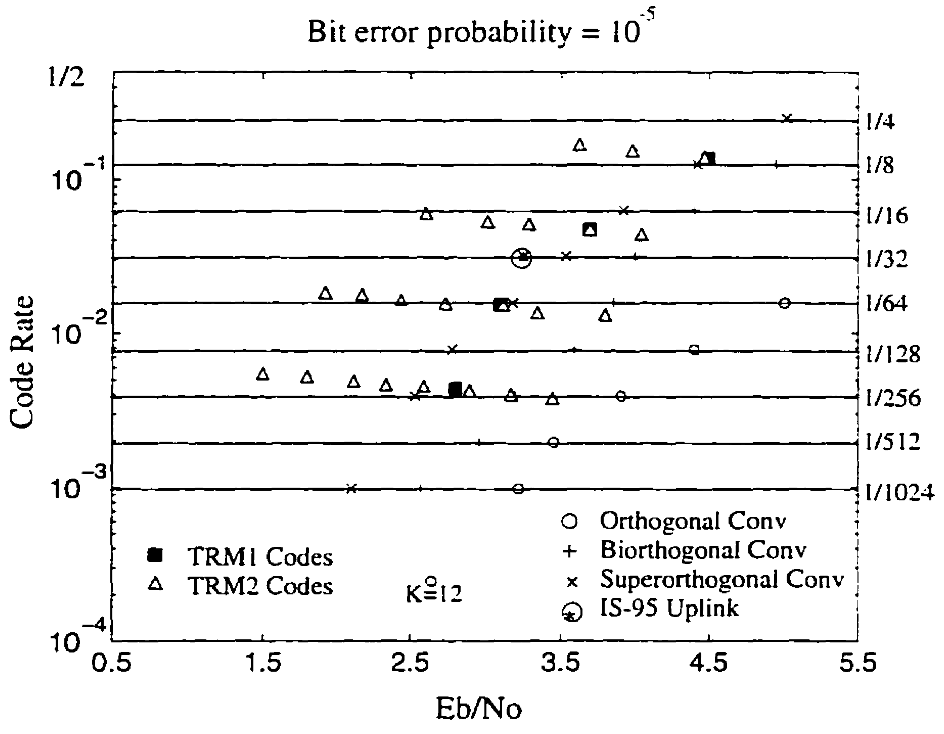


Figure 4.9: Comparison based on coding rate and required E_b/N_0 to achieve a bit error probability of 10^{-5}.

best coding gains for a given coding rate, amongst all the low rate codes considered.

4.4 Effects of Coding on System Capacity

In this section, we try to quantify the increases in capacity that the different coding schemes can provide to an asynchronous DS-CDMA system. The STIR for an uncoded such system has been computed in section 2.1 and given by (2.11). With perfect power control conditions, we have

$$\mathrm{STIR}_{\mathrm{uncoded}} = \frac{2}{\frac{N_0}{E_b} + \frac{\chi_\rho}{l}(M-1)} \tag{4.13}$$

Suppose now that we are using coding on top of PN sequence spreading. Let R_c denote the coding rate, E_s the energy per coded symbol, and let l be the PN sequence spreading factor (on top of the coding, such that the total spreading factor is l/R_c).

The equivalent bit energy to interference ratio is

$$\frac{E_b}{I} = \frac{\mathrm{STIR}_{\mathrm{uncoded}}}{R_c} = \frac{2}{R_c\frac{N_0}{E_s} + \frac{R_c\chi_p}{l}(M-1)} = \frac{2}{\frac{N_0}{E_b} + \frac{B_pT_p}{B_e}(M-1)} \quad (4.14)$$

where B_e is the effective bandwidth expansion factor given by $B_pT_cl/R_c = B_pT_b$, which takes into account the normalized bandwidth of the chip pulse shape. As argued in section 2.2.2, the best pulse shapes minimize the time-bandwidth product T_pB_p. Indeed, (4.14) shows that for a given effective bandwidth expansion factor, the lower T_pB_p is the higher is the effective bit energy to interference ratio. We restrict ourselves, for the remaining part of this section, to the IS-95 chip pulse shape, which was shown to be close to optimal for a 95% energy containment bandwidth definition. Let p be the desired bit error probability which ensures a satisfactory grade of service to the user. We evaluate the impact of coding on the system capacity in the following way. For a given code, we find out the required E_b/I necessary to obtain a bit error probability of p, based on the curves of section 4.2. We then solve for M in (4.14) to obtain

$$M = 1 + \frac{B_e}{B_pT_p}\left(\frac{2}{E_b/I} - \frac{1}{E_b/N_o}\right) \quad (4.15)$$

From (4.15), the relationship between $M-1$ and B_e is linear. Since $B_e = B_pT_cl/R_c$, it follows that PN sequence spreading provides a linear increase in system capacity. In the following, we disregard the TRM1 codes and the orthogonal and biorthogonal convolutional codes, because the TRM2 codes and the superorthogonal convolutional codes (respectively) provide a better performance for similar or larger coding rates.

We will now compare the IS-95 uplink code, the superorthogonal convolutional codes, and the TRM2 codes, based on the number of users per unit bandwidth that a system can support with these different coding schemes. The number of users per unit bandwidth is a function of

- The desired grade of service (for example, a bit error rate of 10^{-4})
- The amount of thermal noise, i.e. the ratio E_b/N_0
- The FOM of the chip shape. We consider here the IS-95 chip shape with a 95% energy containment bandwidth.

- The coding scheme
- The bandwidth expansion factor B_e, or the ratio l/T_c since we restrict ourselves to the IS-95 chip shape with a 95% energy containment bandwidth

Once the above-listed information is known, one can solve for M in (4.15).

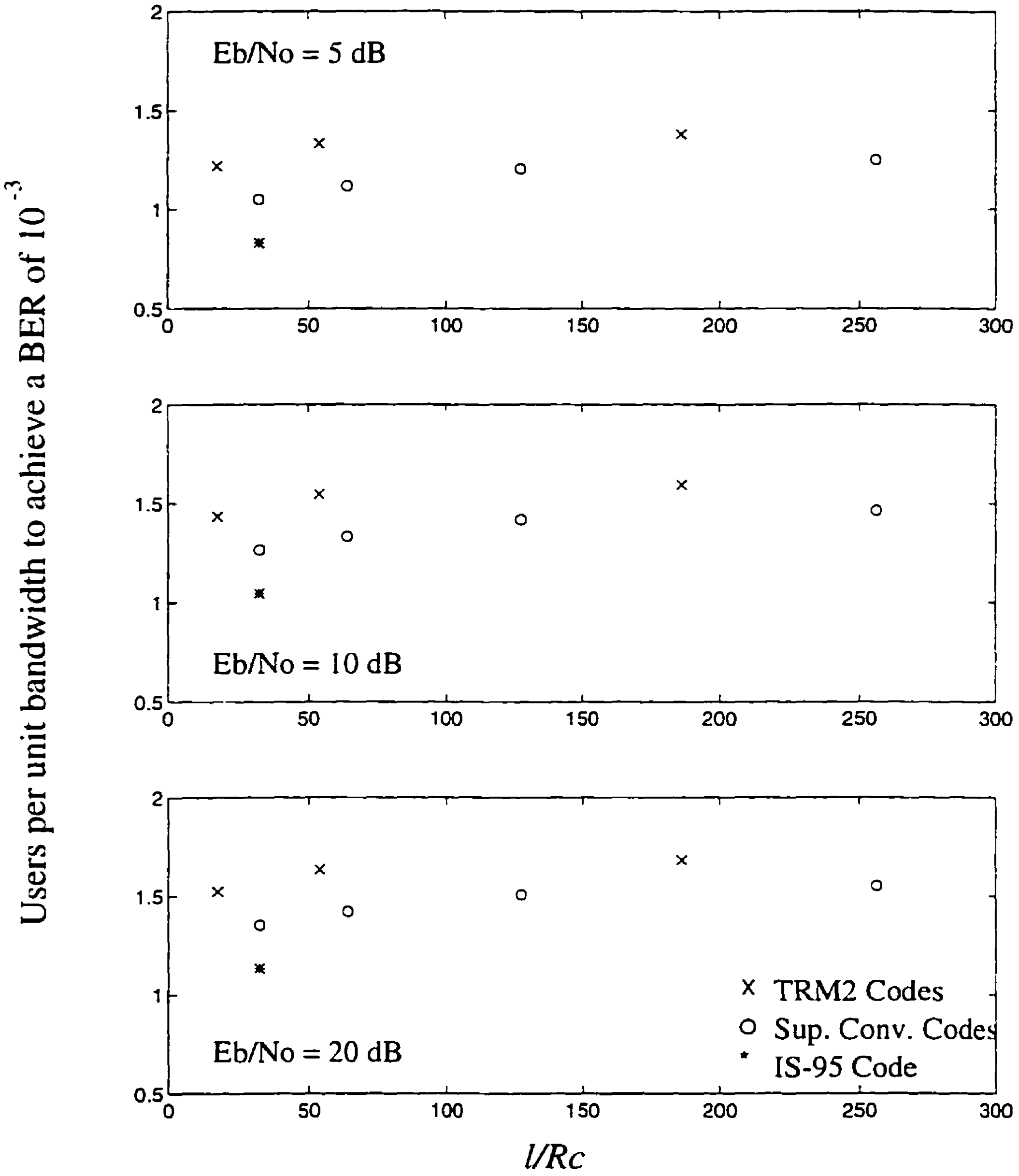


Figure 4.10: Number of users per unit bandwidth for the best TRM2 and super-orthogonal convolutional codes, and the IS-95 code, as a function of l/R_c, and with $p_b = 10^{-3}$.

Since we constrain ourselves to the IS-95 pulse shape, the bandwidth of the system is proportional to l/R_c. Hence we slightly abuse the language and refer to $M/(l/R_c)$ as the capacity per unit bandwidth or the system efficiency. In the Figures that follow, we show the capacity per unit bandwidth versus l/R_c for the IS-95 code, and the best superorthogonal convolutional and TRM2 codes. For each family of codes, we only show a code if it yields a greater efficiency than all codes of the same family with a lower l/R_c. Note that since M is almost proportional to l, the number of users per unit bandwidth is independent of l. Therefore the bandwidth expansion through PN sequence does not increase the system efficiency, emphasizing that spreading through PN sequences is an inefficient spectrum spreading method (in the abscissa of our plots, l/R_c is actually equal to $1/R_c$).

Figure 4.10 shows the capacity per unit bandwidth versus l/R_c for a bit error probability of 10^{-3} grade of service, and three values of E_b/N_0. A 5 dB value for E_b/N_0 represents a relatively noisy environment, whereas a 20 dB value reflects an interference-limited system. The codes shown which yield the largest number of users per unit bandwidth are the rate 1/32, 1/64, 1/128, and 1/256 superorthogonal convolutional codes, and the $(M = 8, b = 6)$, $(m = 10, b = 8)$, and $(m = 12, b = 9)$ TRM2 codes. Note that the $(m = 12, b = 10)$ TRM2 code would yield an even greater capacity per unit bandwidth, but we have not simulated its performance.

It is seen that the best TRM2 codes provide consistently a greater number of users per unit bandwidth than the best superorthogonal convolutional codes and the IS-95 code. Also, the superorthogonal convolutional codes are seen to be superior to the IS-95 coding scheme.

Figure 4.11 shows again the capacity per unit bandwidth for a bit error probabilities of 10^{-4} criterion, and confirms the superiority of TRM2 codes over the other coding schemes considered. Also, the advantage of superorthogonal convolutional codes over the IS-95 code is less pronounced. As a matter of fact, for $l/R_c < 64$, the capacity per unit bandwidth induced by the IS-95 code is slightly greater than that of the corresponding rate 1/32 superorthogonal convolutional code.

In Figure 4.12, the system efficiency is given for a bit error probabilities of 10^{-5}

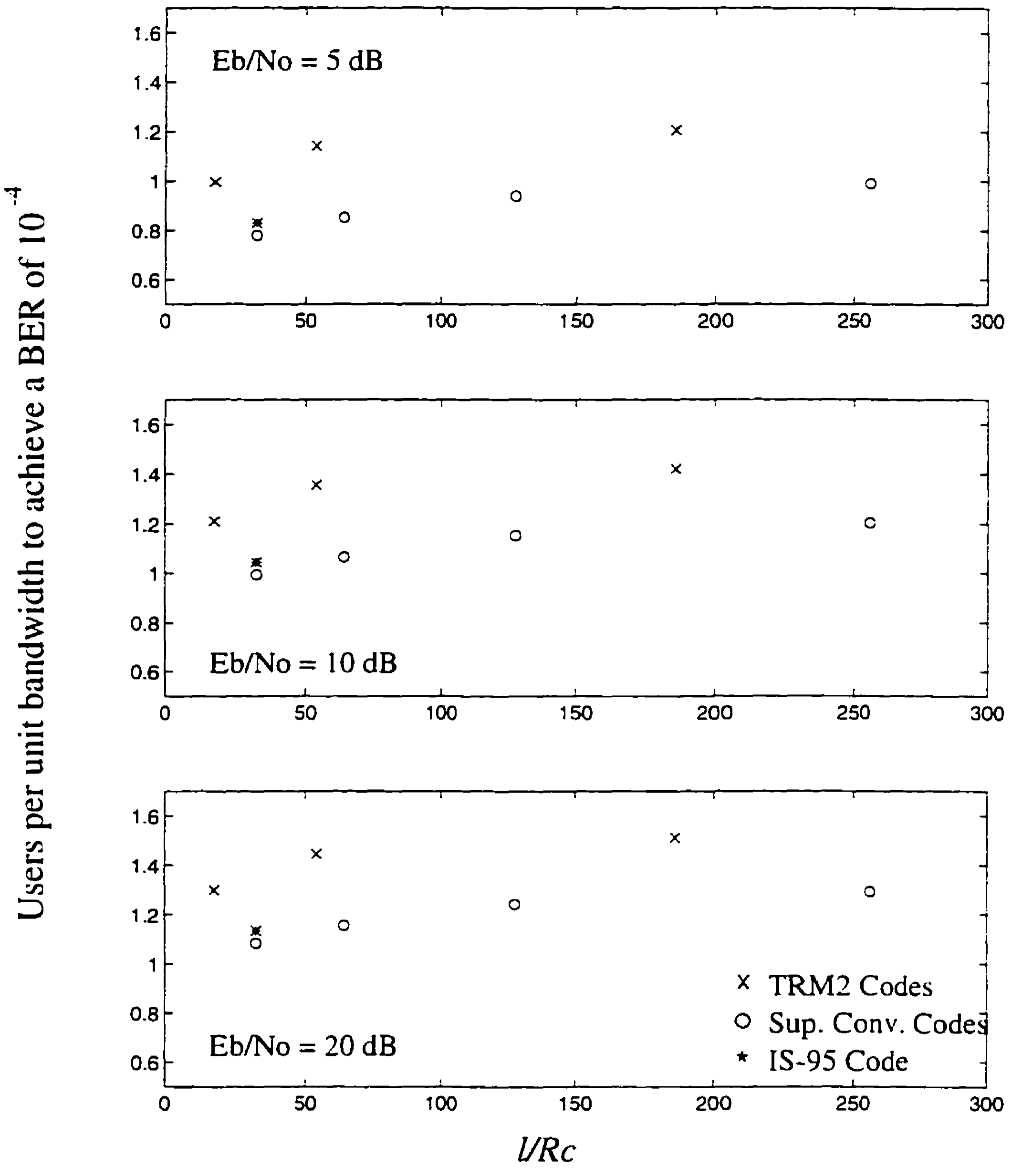


Figure 4.11: Capacity per unit bandwidth for the best TRM2 and superorthogonal convolutional codes, and the IS-95 code, as a function of l/R_c, and with $p_b = 10^{-4}$.

criterion. Unlike the case with less stringent grades of service criteria, the IS-95 code is seen to provide a greater capacity per unit bandwidth than the superorthogonal convolutional codes. The TRM2 codes are still superior to the IS-95 code, albeit less markedly. Indeed, for $l/Rc < 1024/19 \approx 53.9$, the IS-95 code and the best TRM2 code ($m = 8, b = 6$) provide the same number of users per unit bandwidth. However, the TRM2 code requires almost half the bandwidth expansion necessary to the IS-95

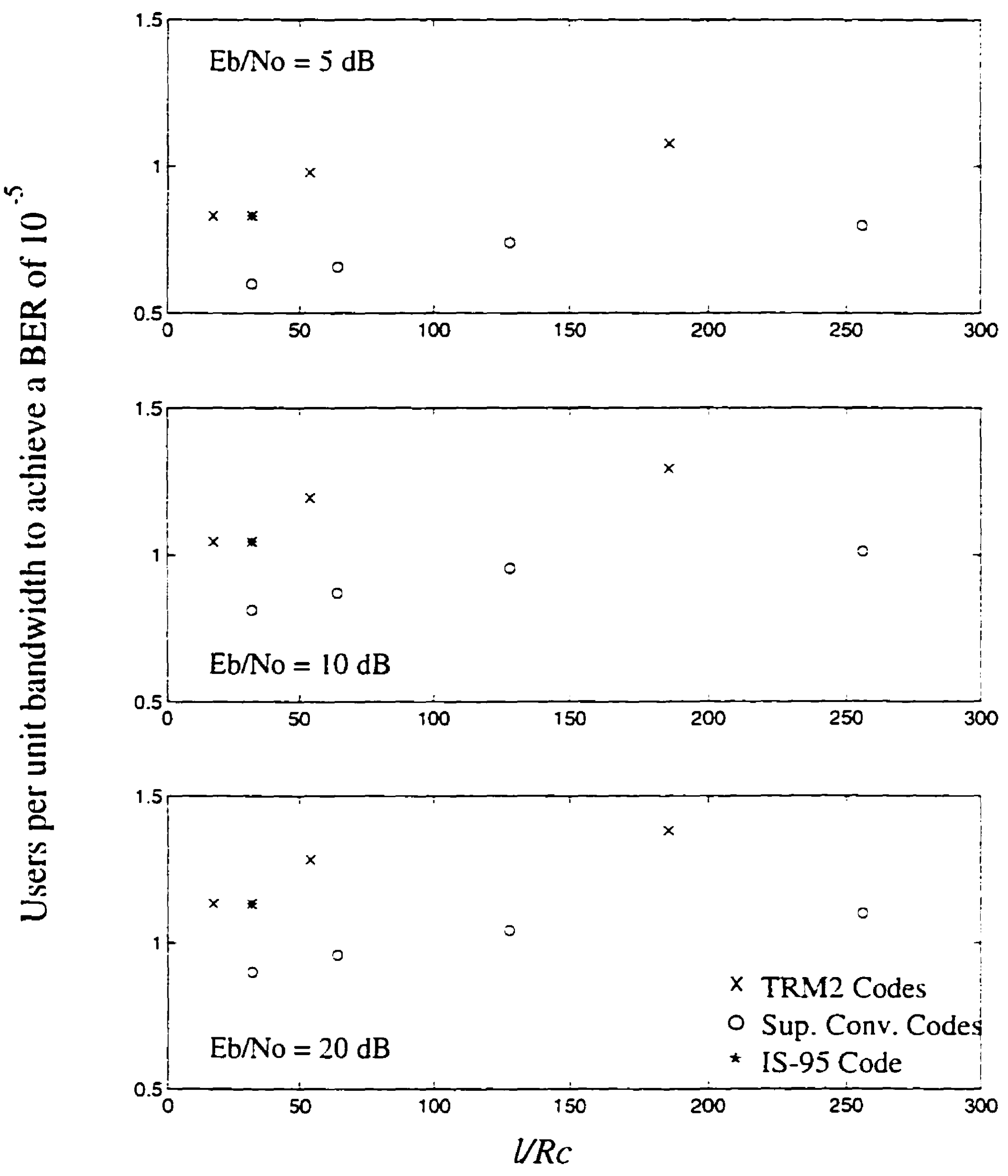


Figure 4.12: Capacity per unit bandwidth for the best TRM2 and superorthogonal convolutional codes, and the IS-95 code, as a function of l/R_c, and with $p_b = 10^{-5}$.

code to achieve the same efficiency. For $l/R_c \geq 53.9$, the TRM2 codes provide a net improvement over the IS-95 code.

In summary, TRM2 codes provide consistently a superior capacity per unit bandwidth over the whole range of bit error probabilities of interest. The improvement over other low rate codes is substantial, ranges from 15 to 40%. Although our definition of capacity employs simplifying assumptions, similar improvements can be expected when system capacity is measured with more practical grades of service criteria (such as blocked and dropped call rates). and more realistic assumptions.

4.5 Complexity Issues

We now address the crucial issue of decoding complexity which was put aside earlier. Indeed. decoding complexity is a prime concern for any implementation of error control coding. We will start by assessing the decoding complexity of the OBS codes. and the IS-95 uplink code. We will then derive the decoding complexity of TRM codes. In doing so. we will show that the combination of several FHTs sharing intermediate results provides some complexity reduction without sacrificing the performance.

4.5.1 OBS Codes

We remind the reader that the maximum-likelihood decoding algorithms of each of the OBS codes are almost identical. They all involve a VA. with the metrics computation being performed by a FHT. and the standard add-compare-select operation. The decoding complexity is thus determined by

1. The FHT. which. for codeword of lengths 2^K requires $K2^K$ additions and subtractions.

2. The number of metrics adjustments (the "add" part of "add-compare-select" operation). Since there are two outgoing branches per state. the number of metric adjustments per decoding stage is twice the number of states. Each metric adjustment requires a single addition or subtraction.

3. The number of "compare" operations. which. for each decoding stage and for each state. equals the number of incoming branches minus one. This is the number of comparisons required to determine the surviving path. We assume that a comparison is of the same complexity as an addition.

4. The number of "select" operations, which delete the non-surviving paths. Their complexity should not exceed that of an addition for each deleted path, at least from an algorithmic point of view. Hence we will assume that the cost of the "select" is the same as that of the "compare" operation.

The maximum-likelihood decoding of an orthogonal convolutional code of rate $1/2^K$ requires a VA with 2^{K-1} states. Each stage of the VA requires one 2^K-FHT for the computation of the metrics, 2^K additions, 2^{K-1} comparisons and selections. The decoding complexity of orthogonal convolutional codes of rate $1/2^K$ per information bit is then

$$\chi_{\text{Ortho}} = K2^K - 2^K - 2 \times 2^{K-1} = 2^K(K-2) \text{ additions/bit} \qquad (4.16)$$

The maximum-likelihood decoding of a biorthogonal convolutional code of rate $1/2^K$ requires a VA with 2^K states. Each stage of the VA requires one 2^K-FHT for the computation of the metrics, 2^{K-1} additions and subtractions, 2^K comparisons and selections. The decoding complexity of biorthogonal convolutional codes of rate $1/2^K$ per information bit is then

$$\chi_{\text{Biortho}} = K2^K - 2^{K-1} - 2 \times 2^K = 2^K(K-4) \text{ additions/bit} \qquad (4.17)$$

Finally, the maximum-likelihood decoding of a superorthogonal convolutional code of rate $1/2^K$ requires a VA with 2^{K-1} states. Each stage of the VA requires one 2^K-FHT for the computation of the metrics, 2^{K-2} additions and subtractions, 2^{K-1} comparisons and selections. The decoding complexity of superorthogonal convolutional codes of rate $1/2^K$ per information bit is then

$$\chi_{\text{Suportho}} = K2^K - 2^{K-2} - 2 \times 2^{K-1} = 2^K(K-8) \text{ additions/bit} \qquad (4.18)$$

The decoding complexity of the superorthogonal convolutional code is slightly greater than that of the biorthogonal convolutional code, which is in turn slightly greater than that of the orthogonal convolutional code. However the difference is rather small while the performances vary considerably. We emphasize, based on (4.16)–(4.18) that the increase in constraint length from the orthogonal to the biorthogonal and then the superorthogonal convolutional code (all of the same rate) does not induce a corresponding doubling in decoding complexity. For example. for the rate 1/256 codes (K=8). we have $\chi_{\mathrm{Ortho}} = 2560$. $\chi_{\mathrm{Biortho}} = 3072$. and $\chi_{\mathrm{Suportho}} = 4096$.

4.5.2 IS-95 uplink code

The IS-95 uplink code has 256 states. Each decoding stage uses one received Reed-Muller codeword. which encodes two data bits. The cost of performing a FHT of length 64 is $6 \times 64 = 384$ additions and subtractions. Moreover. each state has four outgoing branches. which means that the computation of the cumulative metric requires $4 \times 256 = 1024$ additions. Finally. the "compare-select" operation involves four incoming branches per state. In order to find the maximum of four numbers. three comparisons are necessary. The select operation chooses one surviving path out of four. for each state. Hence the complexity associated with the "compare-select" operation is $3 \times 256 \times 2 = 1536$. Putting it all together. the decoding complexity per information bit of the IS-95 uplink code is

$$\chi_{\mathrm{IS-95}} = \frac{384 + 1024 + 1536}{2} = 1472 \text{ additions/bit} \tag{4.19}$$

4.5.3 TRM Codes: Straightforward Metrics Computation

We now turn to the estimation of the maximum-likelihood decoding complexity of Trellis/Reed-Muller Codes. Consider a TRM code with a b-input coset-selecting trellis code. and Reed-Muller code of length 2^m. We denote the number of states of the trellis by σ. and the number of additions required to compute all branch metrics for each decoding stage by $\mathcal{M}$. We need 2^m comparisons per state in order to select the correct primary branch metric. Since each stage has 2^b incoming and outgoing primary branches. we also require $\sigma 2^b$ additions to update the cumulative path metrics. and $2 \times \sigma 2^b$ additions for the "compare-select" operation. per decoding stage. The decoding complexity per bit is then

$$\chi_{\mathrm{TRM}} = \frac{\mathcal{M} + 3\sigma(2^b + 2^m)}{m + b + 1} \text{ additions/bit} \tag{4.20}$$

For TRM1 codes. the number of states is $\sigma_1 = 2^b$. whereas for TRM2 codes. it is $\sigma_2 = 2^{b-1}$.

In a straightforward implementation of the metrics computer. the decoder performs a number of FHTs. one for each coset leader that is used. The FHT complexity is of $m2^m$ additions. For TRM1 codes. 2^{2b} coset leaders are employed. whereas TRM2 codes use 2^{b+1} coset leaders. Hence we can write

$$\chi_{TRM1} = \frac{2^{2b} \times m2^m + 3\sigma(2^b + 2^m)}{m+b+1} = \frac{2^{2b}}{m+b+1}((m+3)2^m + 3) \qquad (4.21)$$

$$\chi_{TRM2} = \frac{2^{b-1} \times m2^m + 3\sigma(2^b + 2^m)}{m+b+1} = \frac{2^{2b-1}}{m+b+1}\left((m+3)2^{m-b} + 3\right) \qquad (4.22)$$

From the above expressions. it follows that the decoding complexity is heavily dominated by the computation of the branch metrics through the series of FHTs. This is particularly true for TRM1 codes and TRM codes with relatively small values of b. However. even when $b = b_{\max}$ for TRM2 codes. the decoding complexity $\frac{2^{2m-3}(4m-15)}{2m-1}$ is dominated by the metrics computation since $4m > 15$.

Another important observation is that. for TRM1 and TRM2 codes with the same m and b. we have

$$\frac{\chi_{TRM1}}{\chi_{TRM2}} = \frac{(m+3)2^m + 3}{(m+3)2^{m-b+1} + 6} > \frac{(m+3)2^m + 3}{(m+3)2^{m-1} + 6} = 2 - \frac{9}{(m+3)2^{m-1} + 6} \qquad (4.23)$$

This shows that TRM2 codes are less complex than TRM1 codes of the same parameters. The inherent reason for this is that. although the TRM2 code has more states. it employs much fewer distinct coset leaders than the TRM1 code of the same parameters.

4.5.4 TRM Codes: Reduced Complexity Metrics Computation

Since the decoding complexity of TRM codes is dominated by the branch metrics computation. it is very important to compute these metrics in the most efficient way. In this section. we will show that different FHTs can share intermediate results and thereby decrease the complexity of the metrics computation.

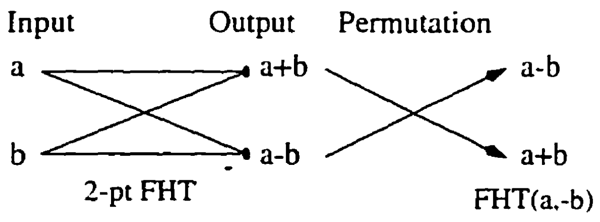


Figure 4.13: FHT of $(a, -b)$ obtained from the FHT of (a, b).

Every stage of the FHT is composed of several two-point FHTs, or butterfly structures. Suppose that we have computed the two-point FHT of $\underline{u} = (a, b)$. Then, as shown in Figure 4.13, the FHT of $(a, -b)$, is obtained by a simple permutation of the FHT of $\underline{u}$. Naturally, FHT$(-a, b)$ = −FHT$(a, -b)$ and FHT$(-a, -b)$ = −FHT(a, b). This shows that the FHT of $(\pm a, \pm b)$ is obtained from the FHT of (a, b) without any further computation, since the permutation can be hardwired and the possible minus signs incorporated in subsequent stages of the larger FHT. Therefore, in the branch metrics computation of TRM decodes, all simultaneous FHTs can share the same first stage.

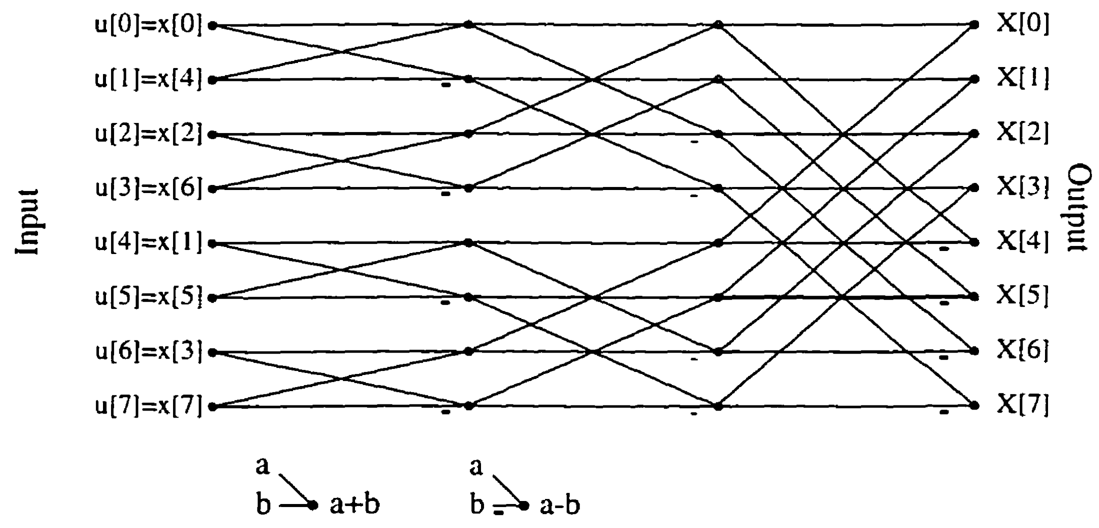


Figure 4.14: 8-point Fast Hadamard Transform.

Moreover, some intermediate results which are at deeper stages may be shared among FHTs. The number of such occurrences depends on the different coset leaders. In order to determine exactly how many intermediate results can be shared or, equivalently, how many 2-pt FHTs are required to compute all branch metrics, we first rearrange the usual structure of the FHT to that of Figure 4.14.

Take a vector $\underline{u}$ and a coset leader $\underline{\lambda}$ of length 2^m, which is already sorted for the FHT implementation of Figure 4.14. Suppose the FHT computation of $\underline{u}$ is available (i.e. the final result and all intermediate values). We wish to determine the number of unnecessary (redundant) intermediate results that would be computed in the FHT evaluation of $\underline{u}\otimes\underline{\lambda}$. A butterfly structure is redundant if the corresponding butterfly in the computation of the FHT of $\underline{u}$ computes the same two functions up to sign and permutation. We have shown that all first-stage butterfly structures are redundant, since all simultaneous FHTs can share the same first stage. Moreover, as we established above, for two 2-point FHTs to compute the same functions up to sign and permutation, they must have the same inputs up to sign and permutation. Indeed, the two outputs of a depth k butterfly structure in the computation of a FHT are of the form

$$\zeta_0 = \sum_{i=\text{offset}}^{2^{k-1}-1+\text{offset}} c_i\, u_i + \sum_{i=2^{k-1}+\text{offset}}^{2^k+\text{offset}-1} c_i\, u_i \tag{4.24}$$

$$\zeta_1 = \sum_{i=\text{offset}}^{2^{k-1}-1+\text{offset}} c_i\, u_i - \sum_{i=2^{k-1}+\text{offset}}^{2^k+\text{offset}-1} c_i\, u_i \tag{4.25}$$

where the c_i's are the ±1 coefficients, and "offset" is a power of two which specifies which butterfly structure of depth k is referred to. The corresponding butterfly structure in the computation of the FHT of $\underline{u}\otimes\underline{\lambda}$ yields

$$\zeta_0' = \sum_{i=\text{offset}}^{2^{k-1}-1+\text{offset}} c_i\, \lambda_i\, u_i + \sum_{i=2^{k-1}+\text{offset}}^{2^k+\text{offset}-1} c_i\, \lambda_i\, u_i \tag{4.26}$$

$$\zeta_1' = \sum_{i=\text{offset}}^{2^{k-1}-1+\text{offset}} c_i\, \lambda_i\, u_i - \sum_{i=2^{k-1}+\text{offset}}^{2^k+\text{offset}-1} c_i\, \lambda_i\, u_i \tag{4.27}$$

For these two butterfly structures to yield the same results up to sign and permutation for any $\underline{u}$, we require

$$\begin{aligned} \lambda_{\text{offset}} &= \lambda_{\text{offset}+1} = \cdots = \lambda_{\text{offset}+2^{k-1}-1} \\ \text{and}\quad \lambda_{\text{offset}+2^{k-1}} &= \lambda_{\text{offset}+2^{k-1}+1} = \cdots = \lambda_{\text{offset}+2^k-1} \end{aligned} \tag{4.28}$$

The above conditions show one possible way to reduce the complexity of the branch metrics computation, by avoiding redundant computations. The number of such redundant operations is estimated by looking at the set of coset leaders, and counting the number of times where (4.28) is verified. We can design an algorithm for this purpose, which works as follows

- Create a list of examined coset leaders which contains only the first one
- for each coset leader $\underline{\lambda}'$ not in the list do
  - for each coset leader λ'' in the list, compute $\underline{\lambda} = \underline{\lambda}' \otimes \underline{\lambda}''$.
  - check whether (4.28) is satisfied for some "offset" values, and some k
  - update accordingly the complexity counter

Such an analysis shows that about 30% reduction in complexity is possible, compared to the straightforward method. However, it is not optimal in general. Indeed, it is shown through an empirical method in [41] that a 40% reduction in complexity is possible for the case $m = 4$. We conjecture therefore that some additional reduction may be achievable.

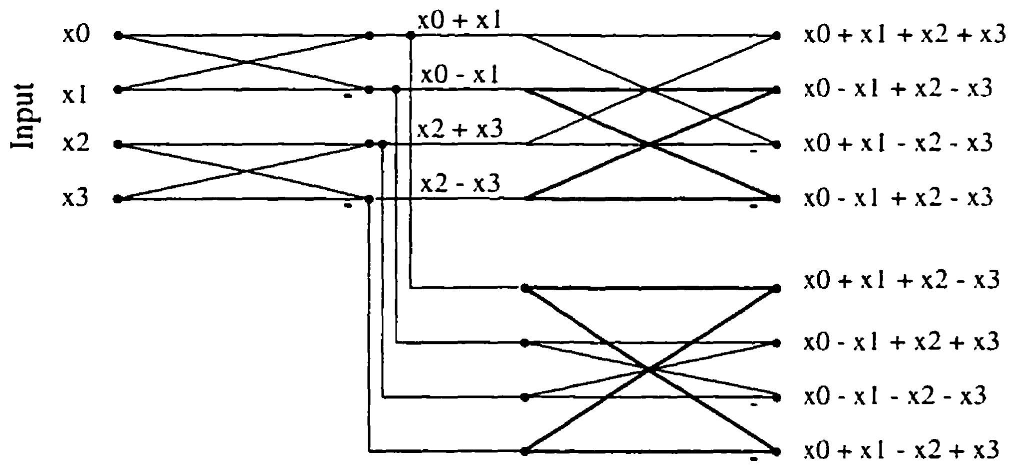


Figure 4.15: Computation of all binary functions of four variables.

The key to a further reduction in complexity is the computation of intermediate results which may not be necessary to any particular FHT stage, but is useful to many later stages. To illustrate this concept, consider the structure of Figure 4.15. Such a structure, which involves 6 butterfly operators, computes all possible functions of four variables of the form

$$f(x_0, x_1, x_2, x_3) = x_0 + \sum_{i=1}^{3} c_i x_i \tag{4.29}$$

with $c_i = \pm 1$. Therefore, such a structure can provide intermediate results to all 4-point FHTs involved in the metrics computation. The structure can also be generalized for an arbitrary number of inputs 2^k. Let ζ_k be the number of butterfly structures required to compute all possible functions of the form

$$f(x_0, \cdots, x_{2^k-1}) = x_0 + \sum_{i=1}^{2^k-1} c_i x_i \tag{4.30}$$

with $c_i = \pm 1$. We will refer to such a form as binary function. Then ζ_{k+1} can be obtained from the following observation: we first need all possible binary functions of $(x_0, \cdots, x_{2^k-1})$ and $(x_{2^k}, \cdots, x_{2^{k+1}-1})$, and then all possible combinations of these. The number of such combinations is given by the number of possible binary functions of $(x_0, \cdots, x_{2^{k+1}-1})$, which is $2^{2^{k+1}-1}$. Since each butterfly structure yields two such combinations, we have

$$\zeta_{k+1} = 2\zeta_k + 2^{2^{k+1}-2} \tag{4.31}$$

and $\zeta_1 = 1$. This is a fast growing function of k, but for $k <= 4$, the values of ζ_k are reasonable. Indeed, $\zeta_2 = 6$, $\zeta_3 = 76$, and $\zeta_4 = 16536$.

We now combine the two methods to try to minimize the complexity of the branch metrics computation. Each branch metric computation starts with the structure of Figure 4.15 of order k. It is then followed by the last $m-k$ stages of the FHT for each coset leader. Moreover, FHTs with different coset leaders share when possible their intermediate results of depth greater than k: this is determined by the first method, where the condition of (4.28) is tested for lengths greater than 2^k. Table 4.1 shows the resulting complexity reduction coefficients (number of required operations/number of operations with a straightforward implementation). These reductions in complexity are not dramatic. However, even the smallest reductions (highest coefficients) are still

TRM1 Codes		TRM2 Codes				
m, b		$b\backslash m$	6	8	10	12
m=6, b=2	62.6 %	2	73.7%	80.2 %	84.2 %	86.6 %
m=8, b=3	64.3 %	3	62.6%	72.0 %	77.7 %	81.3 %
m=10, b=4	68.2 %	4	56.3%	67.0 %	73.7 %	77.9 %
m=12, b=5	69.4 %	5		64.3 %	71.3 &	76.0 %
m=14, b=6	69.3 %	6		62.6 %	69.7 %	74.5 %
		7			68.2 %	73.4 %
		8			66.2 %	71.7 %
		9				69.4 %
		10				66.8 %

Table 4.1: Complexity reduction coefficients achievable for various TRM1 and TRM2 codes

significant. For the case $m = 6$, $b = 4$ TRM2 code, the decoding complexity is reduced almost by half. Whenever the number of coset leaders required by a code was less than the maximum possible ($b < b_{\text{max}}$), we just picked the first ones. It is very likely that an additional complexity reduction is possible by carefully selecting the coset leaders in order for the corresponding FHTs to share as much intermediate results as possible.

Table 4.1 shows that it is possible to obtain at least the complexity reduction coefficients shown. Furthermore, since the complexity reduction coefficients of the TRM2 codes with the maximum number of coset leaders approach 50%, we conjecture that it is unlikely to obtain reductions in complexity significantly better than this figure. For small values of m, it is possible to exhaustively examine all combining possibilities to minimize the number of required computations. In [41], it is argued that 304 additions is the minimum number of operations (additions) needed to compute all 8 FHTs, for $m = 4$. This is a reduction coefficient of 59.4 % over the 512 additions required by a straightforward implementation of the 8 FHTs.

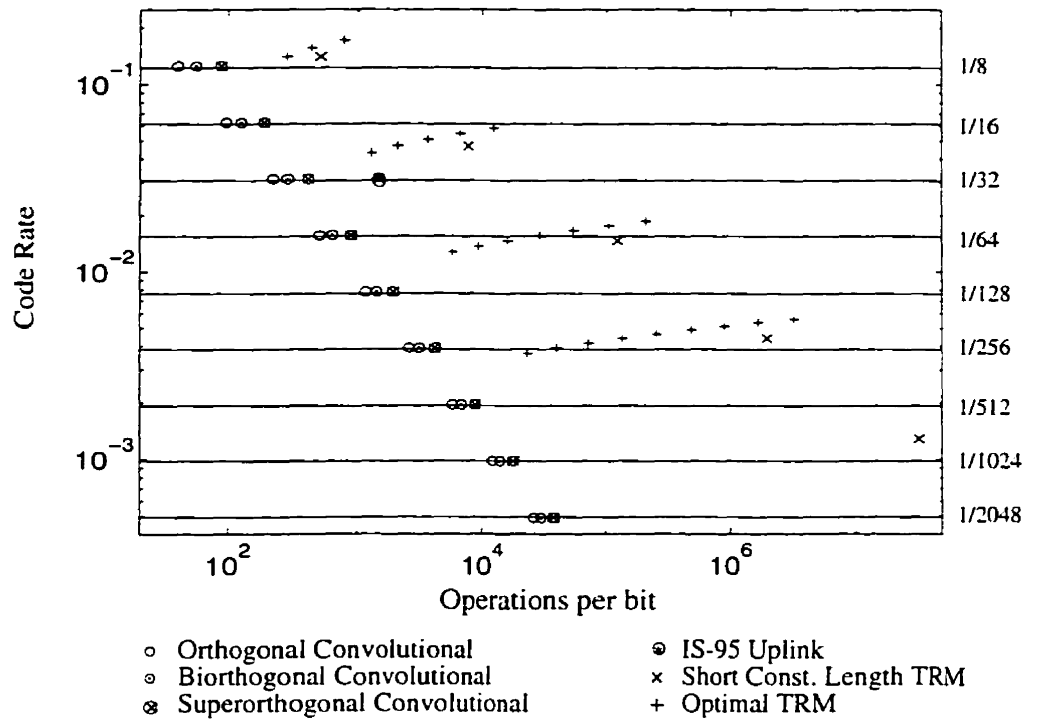


Figure 4.16: Code comparison based on the decoding complexity per bit.

Figure 4.16 shows the complexity per bit and the coding rate for all the codes that were studied so far. It is seen that TRM2 codes require more computations than other codes of comparable rate. These results must be used with caution since the criterion used does not take into account many practical considerations. Indeed, measuring complexity in a fair manner is very difficult. In particular, the ML decoding of TRM codes contains a high degree of parallelism, which can bring down dramatically the implementation complexity. Nevertheless, it is safe to say that TRM codes are more complex than the other codes of similar coding rate under study

4.6 Conclusions

In this section, we have assessed the performance of TRM codes in the presence of AWGN (and multi-user interference modeled as such). It was found that TRM2 codes in particular provide a better coding gain than other low rate codes of similar rate. The analysis of the effect of different codes on the number of users that a system can support has shown the superiority of TRM2 codes.

The capacity improvement carried by TRM2 codes comes at the cost of a larger decoding complexity. Despite the fact that our measure of complexity is somewhat simplistic, it is clear that TRM2 codes are substantially more complex to decode than the other low rate codes of similar rate (and poorer performance). For these reasons, the investigation of sub-optimal decoding techniques might be useful. In particular, all sub-optimum trellis search techniques, which have been widely studied in the context of convolutional codes, can be used for TRM2 codes. The complexity of the FHTs can be reduced by using semi-soft decoding, whereby the received vector is hard-quantized. The FHT implementation is then simpler, and the components of the FHT output vector are integers over $[-2^m - 1, 2^m - 1]$. The metrics to the trellis search algorithm are then naturally quantized to $m + 1$ bits and can be further quantized if needed by simple truncation of the least significant bit(s). Naturally, the benefits of such an approach depend on the performance degradation versus the reduction in complexity that can be achieved.

Chapter 5

Conclusions and Recommendations

In this thesis we have addressed the problems of chip shaping and channel coding in asynchronous DS-CDMA systems. We have derived a criterion for chip shapes which quantifies their interference rejection capabilities while taking their bandwidth into account. We also gave some quasi-optimal pulse shapes for a 95% energy containment bandwidth. The important role that error control coding plays in CDMA was pointed out and quantified. It was revealed that careful chip shaping combined with powerful error control coding could allow CDMA to compete with other multiple access schemes, even in a single cell with AWGN and no other type of interference, which strongly favors orthogonal multiple access schemes.

We have then focused our efforts towards the design of low rate error control codes which would outperform the other known low rate codes that had been proposed for CDMA. The TRM codes presented in this thesis are the result of that effort. We discussed in the detail the construction of TRM codes, both from conceptual and practical points of view, thereby revealing the tradeoffs and constraints particular to this class of codes. We also provided analytical tools which allow the estimation of the bit error probability of TRM codes. In Chapter 4, we used these tools in conjunction with computer simulations to show the improvement obtained with TRM codes over the other low rate error control codes that we consider. It was shown that the impact of such codes yield substantial increases in system capacity, of the order of 20 to 40%.

We also looked at the decoding complexity, where it was found that TRM codes are in general more complex than the other codes. Some reduced-complexity decoding methods were then suggested, which might deserve to be further investigated.

The analyses presented in this work were limited to the AWGN channel and approximations of the other-user interference by Gaussian noise. It is expected that the coding gains exhibited on more realistic channel models are actually greater than those obtained on the AWGN channel. It would be interesting to quantify those "realistic" coding gains, in the presence of multi-user interference not approximated by Gaussian noise, multi-path fading, imperfect power control, varying number of users, other-cell interference, etc...

Our work could also be generalized, for example by replacing the first-order Reed-Muller code by another code rich in structure, which could lead to another family of interesting codes. Another avenue that could be explored consists in the use of coset leaders which are not necessarily bent functions. Indeed, the number of coset leaders candidates in the TRM code constructions presented earlier is limited to 2^{m-1}. For example, instead of partitioning the Kerdock code into its first-order Reed-Muller cosets, one can also partition the second-order Reed-Muller code, $R(2, m)$, into its $2^{m(m-1)/2}$ first-order Reed-Muller cosets. Along any incorrect state transition, the minimum Hamming distance is at least 2^{m-2}. Although this is less than the $2^{m-1} - 2^{\frac{m}{2}-1}$ achievable with bent functions as coset leaders, this alternative construction has the following advantages

- A large number of coset leader candidates allowing for coset-selecting codes of higher rate
- Odd values of m are also permitted, which provides a greater flexibility in the overall coding rates

On the other hand, because of the shorter minimum Hamming distance on different branches, it is predictable that the alternative codes might require coset-selecting codes with a greater constraint length to achieve the same free distance as the TRM codes presented earlier. This in turn would increase the complexity of the Viterbi algorithm. However, the linearity of the second-order Reed-Muller code might allow more efficient branch metric computations, which is the bottleneck in the decoding complexity of TRM1 and TRM2 codes.

In summary. starting with a new approach to the combination of trellis and block codes. we have constructed low rate codes which are superior in performance to all other low rate codes known to us. Needless to say. there are many possible extensions to the work reported in this thesis. in terms of finer analysis. alternative decoding schemes. and variations or generalizations on the proposed TRM codes. It is my wish that the ideas presented in this thesis open up new directions in error control coding. and help in further improvements and refinements in the search for good low rate codes.

Appendix A

A.1 Computation of the Statistics of the Decision Variable y_k

A.1.1 Mean of the Decision Variable

The mean of the decision variable is meaningful only when the transmitted information is known. More precisely, we are interested in computing the mean of y_k conditioned on x_k.

$$\begin{aligned} E[y_k \mid x_k] &= E\left[\int \left(\sum_{j=1}^{M} x_j(t-D_j)\cos\theta_j - \sqrt{2}\eta(t)\cos\omega t\right) c^k(t)\, dt \;\Big|\; x_k\right] \\ &= E\left[\int \left(\sum_{j=1}^{M} \sqrt{\alpha_j} \sum_{r=-\infty}^{\infty} x_{j,r}\, c_j^r(t-D_j)\, c^k(t)\cos\theta_j\right) dt \;\Big|\; x_k\right] \qquad (A.1) \end{aligned}$$

where we used the fact that $\eta(t)$ is independent of all the other random quantities in the expression of $E[y_k \mid x_k]$, and that $E[\eta(t)] = 0$. Interchanging integration (and summation) with expectation,

$$\begin{aligned} E[y_k \mid x_k] &= \int \sum_{j=1}^{M} \sqrt{\alpha_j} \sum_{r=-\infty}^{\infty} E[x_{j,r}\, c_j^r(t-D_j)\, c^k(t)\cos\theta_j \mid x_k]\, dt \qquad (A.2) \\ &= \int \sum_{j=1}^{M} \sqrt{\alpha_j} \sum_{r=-\infty}^{\infty} E[x_{j,r} \mid x_k]\, E[c_j^r(t-D_j)\, c^k(t)]\, E[\cos\theta_j]\, dt \end{aligned}$$

where we used the mutual independence of the data, the spreading sequences, and the random phases. On the other hand, the random variable θ_j ($j \neq M$) is uniformly distributed on $[0, 2\pi]$, if $j \neq M$, whereas $\theta_M = 0$. Hence,

$$E[\cos\theta_j] = \begin{cases} 1 & j = M \\ 0 & j \neq M \end{cases} \tag{A.3}$$

Therefore, recalling that $\alpha_M = 1$, we end up with

$$E[y_k|x_k] = \int \sum_{r=-\infty}^{\infty} E[x_r|x_k] E[c^{(r)}(t)\ c^{(k)}(t)]\ dt \tag{A.4}$$

Let us examine the quantity

$$\begin{aligned} E[c^{(r)}(t)\ c^{(k)}(t)] &= E\left[\sum_{m=0}^{l-1} c_{M,m+rl}\ \rho(t - mT_c - rlT_c) \sum_{n=0}^{l-1} c_{M,n+kl}\ \rho(t - nT_c - klT_c)\right] \\ &= \sum_{m=0}^{l-1}\sum_{n=0}^{l-1} E[c_{M,m+rl}\ c_{M,n+kl}]\rho(t - mT_c - rlT_c)\rho(t - nT_c - klT_c) \end{aligned} \tag{A.5}$$

Note that the indices $m + rl = n + kl$ if and only if $m = n$ and $r = k$. Using (2.1), the above expression is non-zero only for $r = k$ and we have

$$E\left[\left(c^{(k)}(t)\right)^2\right] = \sum_{m=0}^{l-1} \rho^2(t - mT_c - klT_c) \tag{A.6}$$

Moreover, since $E[x_k|x_k] = x_k$, (A.4) reduces to

$$E[y_k|x_k] = x_k \int E\left[\left(c^{(k)}(t)\right)^2\right]\ dt = x_k \sum_{m=0}^{l-1} \underbrace{\int \rho^2(t - mT_c - klT_c)dt}_{E_c} \tag{A.7}$$

The conditional mean of the decision variable is then

$$E[y_k|x_k] = lE_c x_k \tag{A.8}$$

A.1.2 Variance of the decision variable

We start by computing the second moment of $y_k|x_k$.

$$\begin{aligned} E[y_k^2|x_k] = E\Bigg[\Bigg\{ & \int \left(\sum_{j=1}^{M} x_j(t + D_j)\ \cos\theta_j + \sqrt{2}\eta(t)\cos\omega t\right) c^{(k)}(t)\ dt \\ & \times \int \left(\sum_{m=1}^{M} x_m(\tau + D_m)\ \cos\theta_m + \sqrt{2}\eta(\tau)\cos\omega\tau\right) c^{(k)}(\tau)\ d\tau \Bigg\}\Bigg| x_k\Bigg] \end{aligned} \tag{A.9}$$

Using the independence of the noise from all the other random quantities in $E[y_k^2|x_k]$ and the fact that the noise has a zero mean, we obtain

$$E[y_k^2|x_k] = \underbrace{\iint 2E[\eta(t)\eta(\tau)]E[c^{(k)}(t)c^{(k)}(\tau)\Big| x_k]\cos\omega t\cos\omega\tau\, dt\, d\tau}_{\mathcal{N}_\eta} \qquad (A.10)$$
$$+\iint E\left[\sum_{j=1}^{M}\sum_{m=1}^{M} x_j(t+D_j)x_m(\tau+D_m)\cos\theta_j\cos\theta_m c^{(k)}(t)c^{(k)}(\tau)\Bigg| x_k\right] dt d\tau$$

Replacing $E[\eta(t)\eta(\tau)]$ by $\frac{N_0}{2}\delta(t-\tau)$, and using the independence between the data and the spreading sequences, the thermal noise contribution $\mathcal{N}_\eta$ becomes

$$\begin{aligned}\mathcal{N}_\eta &= \frac{N_0}{2}\int 2E\left[\left(c^{(k)}(t)\cos^2\omega t\right)^2\right] dt \\ &= \frac{N_0}{2}\underbrace{\int E\left[\left(c^{(k)}(t)\right)^2\right] dt}_{lE_c} + \frac{N_0}{2}\underbrace{\int E\left[\left(c^{(k)}(t)\right)^2\right]\cos 2\omega t\, dt}_{0 \text{ for } \omega \gg 1/T_c} = lE_c\frac{N_0}{2} \qquad (A.11)\end{aligned}$$

Returning now to the full expression of the second moment of $y_k|x_k$, and replacing the $x_j(t)$'s by their expansion in function of the data and the spreading sequences, we get

$$E[y_k^2|x_k] = lE_c\frac{N_0}{2} + \int\sum_{j=1}^{M}\sum_{m=1}^{M}\sqrt{\alpha_j\alpha_m}\sum_{p,r=-\infty}^{\infty}\Big\{ E[x_{j,p}x_{m,r}|x_k]E[\cos\theta_j\cos\theta_m]$$
$$E[\,c_j^{(p)}(t+D_j)\,c_m^{(r)}(\tau+D_m)\,c^{(k)}(t)\,c^{(k)}(\tau)]\Big\}\, dt\, d\tau \qquad (A.12)$$

Given that

$$E[\cos\theta_j\cos\theta_m] = \begin{cases} 1 & m=j=M \\ \frac{1}{2} & m=j\neq M \\ 0 & m\neq j\end{cases} \qquad (A.13)$$

We have,

$$E[y_k^2|x_k] = lE_c\frac{N_0}{2} + \frac{1}{2}\iint\sum_{j=1}^{M-1}\alpha_j\sum_{p,r=-\infty}^{\infty} E[x_{j,p}x_{j,r}|x_k]\,\underbrace{E[\,c_j^{(p)}(t+D_j)\,c_j^{(r)}(\tau+D_j)\,c^{(k)}(t)\,c^{(k)}(\tau)]}_{\mathcal{A}_j(t,\tau)}\, dt\, d\tau$$
$$+\iint\sum_{p,r=-\infty}^{\infty} E[x_p x_r|x_k]\,E[\,c^{(p)}(t)\,c_j^{(r)}(\tau)\,c^{(k)}(t)\,c^{(k)}(\tau)]\, dt\, d\tau \qquad (A.14)$$

The first term is the AWGN contribution. the second term is the OUI contribution and the third term corresponds to the contribution of the desired user. We have

$$\begin{aligned}\mathcal{A}_j(t,\tau) &= E\left[\sum_{n=0}^{l-1} c_{j,n+pl}\,\rho(t-nT_c-plT_c+D_j)\sum_{m=0}^{l-1} c_{j,m+rl}\,\rho(\tau-mT_c-rlT_c+D_j)\right.\\ &\quad\left.\times\sum_{u=0}^{l-1} c_{M,u+kl}\rho(t-uT_c-klT_c)\sum_{v=0}^{l-1} c_{M,v+kl}\,\rho(\tau-vT_c-klT_c)\right] \qquad \text{(A.15)}\\ &= \sum_{n,m,u,v=0}^{l-1}(\cdots)\times E[c_{j,n+pl}c_{j,m+rl}c_{M,u+kl}c_{M,v+kl}] \qquad \text{(A.16)}\end{aligned}$$

and with the understanding that $j \neq M$,

$$E[c_{j,n+pl}c_{j,m+rl}c_{M,u+kl}c_{M,v+kl}] = E[c_{j,n+pl}c_{j,m+pl}]E[c_{M,u+kl}c_{M,v+kl}] = \begin{cases}1 & n=m,\ u=v \\ & r=p \\ 0 & \text{Otherwise}\end{cases}$$

In the OUI contribution of (A.14), we can drop all the terms in double sum when $p \neq r$. Hence, we can write

$$\begin{aligned}\mathcal{A}_j(t,\tau) &= E\left[\sum_{n=0}^{l-1}\rho(t-nT_c-plT_c+D_j)\,\rho(\tau-nT_c-plT_c+D_j)\sum_{u=0}^{l-1}\rho(t-uT_c-klT_c)\rho(\tau-uT_c-klT_c)\right]\\ &= \sum_{n,u=0}^{l-1}\frac{1}{T_c}\int_0^{T_c}\rho(t-(n+pl)T_c+\xi)\,\rho(\tau+(n+pl)T_c+\xi)\rho(t-(u+kl)T_c)\rho(\tau-(u+kl)T_c)d\xi\end{aligned}$$

since D_j is uniformly distributed on $[0,T_c]$. We notice that $\mathcal{A}_j(t,\tau)$ does not depend on j anymore (as long as $j \neq M$). On the other hand, it depends on p. Let us then define $\mathcal{B}_p(t,\tau) \equiv \mathcal{A}_j(t,\tau)$ as given above. As p describes $\mathbb{Z}$, and n goes from 0 to $l-1$, $n+pl$ also describes $\mathbb{Z}$. Thus,

$$\begin{aligned}\iint\sum_{p=-\infty}^{\infty}\mathcal{B}_p(t,\tau)dtd\tau &= \iint\sum_{q=-\infty}^{\infty}\sum_{u=0}^{l-1}\frac{1}{T_c}\int_0^{T_c}\rho(t-qT_c+\xi)\,\rho(\tau+qT_c+\xi)\rho(t-(u+kl)T_c)\rho(\tau-(u+kl)T_c)dtd\tau\\ &= \frac{1}{T_c}\iiint\sum_{u=0}^{l-1}\rho(t+\xi)\,\rho(\tau+\xi)\rho(t-(u+kl)T_c)\rho(\tau-(u+kl)T_c)d\xi dtd\tau \qquad \text{(A.17)}\end{aligned}$$

We perform a first change of variables $x = t-(u+kl)T_c$ and $y = \tau-(u+kl)T_c$.

$$\iint\sum_{p=-\infty}^{\infty}\mathcal{B}_p(t,\tau)dtd\tau = \frac{1}{T_c}\iiint\sum_{u=0}^{l-1}\rho(x+(u+kl)T_c+\xi)\,\rho(y+(u+kl)T_c+\xi)\rho(x)\rho(y)d\xi dxdy$$

We continue with $z = (u+kl)T_c+\xi$ which yields

$$\begin{aligned}\iint\sum_{p=-\infty}^{\infty}\mathcal{B}_p(t,\tau)dtd\tau &= \frac{1}{T_c}\iiint\sum_{u=0}^{l-1}\rho(x+z)\,\rho(y+z)\rho(x)\rho(y)dzdxdy\\ &= \frac{l}{T_c}\int R_\rho^2(z)dz \qquad \text{(A.18)}\end{aligned}$$

where the last line is obtained by recognizing $R_\rho(z) = \int \rho(x+z)\,\rho(x)\,dx$ as the autocorrelation function of $\rho(z)$. Since the Fourier transform of $R_\rho(z)$ is nothing but the power spectral density $|F_\rho(f)|^2$, of $\rho(t)$, we have by Parseval's equality

$$\frac{l}{T_c}\int R_\rho^2(z)dz = \frac{l}{T_c}\int |F_\rho(f)|^4 df \qquad (A.19)$$

We can now go back to (A.14), and write the second moment of the decision variable as

$$E[y_k^2|x_k] = lE_c\frac{N_0}{2} + \frac{1}{2}\sum_{j=1}^{M-1}\alpha_j\frac{l}{T_c}\int |H(f)|^4 df + \text{desired user contribution}(A.20)$$

We can avoid the explicit calculation of the desired user contribution by noting that if $\alpha_j = 0$ for $j = 1,\cdots,M-1$, then the communications system is the familiar BPSK system over an AWGN channel. In particular, the variance of the decision variable in the absence of noise ($N_0 = 0$) and OUI should be zero. Thus the other user contribution must be the square of the conditional mean of the decision variable, given explicitly by $(lE_c)^2$. We end up with the following relations

$$E[y_k|x_k] = lE_c\,x_k = E_b\,x_k \qquad (A.21)$$

$$\text{Var}[y_k|x_k] = \frac{lE_c}{2}(N_o + E_c\chi_\rho\gamma(M)) \qquad (A.22)$$

where

$$\gamma(M) \equiv \sum_{j=1}^{M-1}\alpha_j \quad \text{and} \quad \chi_\rho \equiv \frac{1}{T_c}\frac{\int |F_\rho(f)|^4 df}{E_c^2} \qquad (A.23)$$

and $E_b = lE_c$ is the energy per bit of the desired user when no channel coding is used.

A.2 Shape Factor of Square Root Raised Cosine Pulses

As given in [13, pp. 534–536], the square-root raised cosine pulse Fourier transform $F_\rho(f)$ such that

$$|F_\rho(f)|^2 = \begin{cases} E_cTc & 0 \leq |f| \leq (1-\delta)/2T_c \\ E_c\frac{T_c}{2}\left[1+\cos\frac{\pi T_c}{\delta}\left(|f|-\frac{1-\delta}{2T_c}\right)\right] & \frac{1-\delta}{2T_c} \leq |f| \leq \frac{1+\delta}{2T_c} \\ 0 & |f| > \frac{1+\delta}{2T_c} \end{cases} \tag{A.24}$$

Using the even parity of $|F_\rho(f)|^2$, the pulse shape factor is then

$$\begin{aligned} \chi_\rho &= \frac{2}{E_c^2T_c}\left\{E_c^2T_c\frac{1-\delta}{2} + \frac{E_c^2T_c^2}{4}\int_{\frac{1-\delta}{2T_c}}^{\frac{1+\delta}{2T_c}}\left(1+\cos\frac{\pi T_c}{\delta}\left(f-\frac{1-\delta}{2T_c}\right)\right)^2 df\right\} \\ &= 1-\delta+\frac{T_c}{2}\frac{\delta}{\pi T_c}\left[\frac{3}{2}\frac{\pi T_c}{\delta}f + 2\sin\frac{\pi T_c}{\delta}\left[f-\frac{1-\delta}{2T_c}\right]_{\frac{1-\delta}{2T_c}}^{\frac{1+\delta}{2T_c}} + \frac{1}{4}\sin\frac{2\pi T_c}{\delta}\left(f-\frac{1-\delta}{2T_c}\right)\right]_{\frac{1-\delta}{2T_c}}^{\frac{1+\delta}{2T_c}} \\ &= 1-\delta+\frac{3T_c}{4}\times\frac{\delta}{T_c} = 1-\frac{\delta}{4} \end{aligned} \tag{A.25}$$

We also use this opportunity to compute the bandwidth of the square root raised cosine pulse. We will consider only the energy containment bandwidth, i.e. the energy band around 0 which contains a given fraction η of the total energy of the pulse. In other words, the bandwidth W is such that

$$\int_{-W}^{W}|F_\rho(f)|^2\,df = \eta E_c \tag{A.26}$$

The left hand side of the above equation is

$$\begin{aligned} \int_{-W}^{W}|F_\rho(f)|^2 &= (1-\delta)E_c + E_cT_c\int_{\frac{1-\delta}{2T_c}}^{W}\left[1+\cos\frac{\pi T_c}{\delta}\left(f-\frac{1-\delta}{2T_c}\right)\right]df \\ &= (1-\delta)E_c + E_cT_c\left(W-\frac{1-\delta}{2T_c}+\frac{\delta}{\pi T_c}\sin\frac{\pi T_c}{\delta}\left(f-\frac{1-\delta}{2T_c}\right)\Big|_{\frac{1-\delta}{2T_c}}^{W}\right) \\ &= \frac{E_cT_c}{2}W + \frac{\delta E_c}{\pi}\sin\frac{\pi T_c}{\delta}\left(W-\frac{1-\delta}{2T_c}\right) \end{aligned} \tag{A.27}$$

Thus (A.26) becomes

$$\Leftrightarrow T_cW + \frac{\delta}{\pi}\cos\left(\frac{\pi}{\delta}\left(T_cW-\frac{1}{2}\right)\right) = \eta \tag{A.28}$$

The latter can be easily solved numerically for given values of δ and η

Appendix B

B.1 Statistics of $\mathcal{H}(r\otimes\lambda')$

$$\begin{aligned}
E\left[\mathcal{H}(\underline{r}\otimes\underline{\lambda}')\right] &= E\left[\mathcal{H}(\underline{c}_i\otimes\underline{\lambda}\otimes\underline{\lambda}' + \underline{n}\otimes\underline{\lambda}')\right] = 2^{-m/2}(\underline{c}_i\otimes\underline{\lambda}\otimes\underline{\lambda}')H_m + 2^{-m/2}E\left[(\underline{n}\otimes\underline{\lambda}')H_m\right] \\
&= 2^{-m/2}(\underline{c}_i\otimes\underline{\lambda}\otimes\underline{\lambda}')H_m \\
&= \begin{cases} (0,\cdots,0,\pm 2^{m/2},0,\cdots,0) & \lambda=\lambda' \\ 2^{-m/2}(\underline{c}_i\otimes\underline{\lambda}\otimes\underline{\lambda}')H_m & \lambda\neq\lambda' \end{cases} \qquad \text{(B.1)}
\end{aligned}$$

$$\begin{aligned}
\text{Cov}\left[\mathcal{H}(\underline{r}\otimes\underline{\lambda})\right] &= E\left[\mathcal{H}^T(\underline{r}\otimes\underline{\lambda}')\mathcal{H}(\underline{r}\otimes\underline{\lambda}')\right] - E\left[\mathcal{H}(\underline{r}\otimes\underline{\lambda}')\right]^T E\left[\mathcal{H}(\underline{r}\otimes\underline{\lambda}')\right] \\
&= 2^{-m}E\left[H_m^T(\underline{n}\otimes\underline{\lambda}')^T(\underline{n}\otimes\underline{\lambda}')H_m\right] \\
&= 2^{-m}E\left[\begin{pmatrix} <\underline{c}_1|\underline{n}\otimes\underline{\lambda}'> \\ \vdots \\ <\underline{c}_{2^m}|\underline{n}\otimes\underline{\lambda}'> \end{pmatrix}(<\underline{c}_1|\underline{n}\otimes\underline{\lambda}'>,\cdots,<\underline{c}_{2^m}|\underline{n}\otimes\underline{\lambda}'>)\right] \qquad \text{(B.2)}
\end{aligned}$$

Hence,

$$\begin{aligned}
\text{Cov}\left[\mathcal{H}(\underline{r}\otimes\underline{\lambda}')\right]_{kl} &= 2^{-m}E[<\underline{c}_k|\underline{n}\otimes\underline{\lambda}'><\underline{c}_l|\underline{n}\otimes\underline{\lambda}'>] = 2^{-m}E[\sum_{u=1}^{2^m} c_{ku}n_u\lambda'_u \sum_{v=1}^{2^m} c_{lv}n_v\lambda'_v] \\
&= 2^{-m}\sum_{u=1}^{2^m}\sum_{v=1}^{2^m} c_{ku}c_{lv}\lambda'_u\lambda'_v \underbrace{E[n_u n_v]}_{\sigma^2\delta_{uv}} = 2^{-m}\sigma^2\sum_{u=1}^{2^m} c_{ku}c_{lu}\underbrace{\lambda'^2_u}_{1} \\
&= 2^{-m}\sigma^2 <\underline{c}_k|\underline{c}_l> \\
&= \begin{cases} 2^{-m}\sigma^2 & k=l \\ 0 & \text{Otherwise} \end{cases} \qquad \text{(B.3)}
\end{aligned}$$

B.2 Upper bound on $P(d, l)$

In this section, we find an upper bound on $P(d, l)$ which will allow an easy evaluation of (3.33). Let $\underline{c}$ be a codeword of some Reed-Muller coset code, represented by one branch in the TRM full trellis. All Reed-Muller coset codes (represented by the groups of 2^{m+1} parallel branches in the TRM full trellis) which do not contain $\underline{c}$, contain 2^m codewords at a Hamming distance $w_1 = 2^{m-1} - 2^{m/2-1}$ from $\underline{c}$, and 2^m codewords at a Hamming distance $w_2 = 2^{m-1} + 2^{m/2-1}$ from $\underline{c}$.

Consider now a path in the primary trellis with a modified distance d to the zero path, and which diverges from the zero path for $l > d$ branches. This path, when mapped to the full TRM trellis (refer to Figure 3.5), generates $2^{(m+1)l}$ paths. We wish to find out the Hamming weight distribution of these $2^{(m+1)l}$ paths.

Each of the d primary branches with a non-zero modified weight generates, in the full TRM trellis, 2^m branches with a Hamming weight of w_1 and 2^m branches with a Hamming weight of w_2. For a given distribution of w_1 and w_2 in the d primary branches, there are then $(2^m)^d$ different paths. The number of distinct distributions with i occurrences of w_1 and $d - i$ occurrences of w_2 is just $\binom{d}{i}$. Thus, among the $2^{(m+1)d}$ paths generated by the d primary branches in question, there are $\binom{d}{i} 2^{md}$ paths of weight $i\, w_1 + (d-i)w_2$, (with $i = 0, \cdots, d$).

The $l-d$ remaining primary branches (with a zero modified distance to the correct path), generate $(2^{m+1})^{l-d}$ paths. Each primary branch is associated to 2^{m+1} full trellis branches, one of Hamming weight 0, one of Hamming weight 2^{m+1} and $2^{m+1} - 2$ of Hamming weight 2^m (this corresponds to the weight distribution of the Reed-Muller code). Of the $(2^{m+1})^{l-d}$ full trellis paths, there are then $\binom{l-d}{j} \binom{l-d-j}{k} (2^{m+1}-2)^{l-d-j-k}$ of Hamming weight $k2^{m+1} + (l-d-j-k)2^m$. This is obtained by noticing that there are $\binom{l-d}{j}$ ways of choosing j branches of Hamming weight 0 in a path of length $l-d$. Once these are chosen, there are $\binom{l-d-j}{k}$ ways of choosing k branches of Hamming weight 2^{m+1}. The remaining $l-j-k$ branches must have Hamming weights of 2^m.

Putting it all together, we have

$$A(i,j,k,d,l) = \begin{cases} \binom{d}{i} 2^{md} \binom{l-d}{j} \binom{l-d-j}{k} (2^{m+1}-2)^{l-d-j-k} & \begin{cases} i = 0,\cdots,d \\ j = 0,\cdots,l-d \\ k = 0,\cdots,l-d-j \end{cases} \\ 0 & \text{Otherwise} \end{cases} \quad \text{(B.4)}$$

where $A(i,j,k,d,l)$ is the number of paths of length l, modified weight d and Hamming weight $iw_1 + (d-i)w_2 + k2^{m+1} + (l-d-j-k)2^m$. The probability of selecting an incorrect primary trellis path can be upper bounded by using a union bound on $P(d,l)$. Let $\bar{P}(d,l)$ denote that upper bound.

$$\bar{P}(d,l) = \sum_{i=0}^{d}\sum_{j=0}^{l-d}\sum_{k=0}^{l-d-j} A(i,j,k,d,l) Q\left(\sqrt{\frac{2(iw_1 + (d-i)\,w_2 + k2^{m+1} + (l-d-j-k)2^m)E_s}{N_o}}\right)$$

where the energy per channel symbol E_s is related to the energy per data bit E_b and the energy per codeword E_d by

$$E_s = \frac{m+b+1}{2^m} E_b = \frac{1}{2^m} E_d \quad \text{(B.5)}$$

We then use (3.33) to obtain

$$\begin{aligned} P_b \leq\ & \frac{m+1}{m+b+1} p_{RM} + \frac{m+1}{m+b+1}(p_{bent} - p_{RM}) \sum_{l=d_{mf}}^{\infty} \sum_{d=d_{mf}}^{l} d\,c(d,l)\bar{P}(d,l) \\ & + \frac{m+1}{m+b+1} p_{RM} \sum_{l=d_{mf}}^{\infty} \sum_{d=d_{mf}}^{l} c(d,l)\, l\bar{P}(d,l) + \frac{1}{m+b+1} \sum_{l=d_{mf}}^{\infty} \sum_{d=d_{mf}}^{l} h(d,l)\bar{P}(d,l) \quad \text{(B.6)} \end{aligned}$$

In practice, the infinite sums in (B.6) are truncated to a finite number of terms.

It is possible to obtain a simpler albeit looser upper bound on the bit error probability. The idea is to consider that all branches have either a Hamming weight of zero (if they belong to a primary branch of zero modified weight) or a Hamming weight of w_1 (if they belong to a primary branch with a modified weight of 1). Then $\bar{P}(d,l)$ becomes

$$\begin{aligned} P_{dw_1} &= A(d, l-d, 0, d, l) Q\left(\sqrt{\frac{2(dw_1)E_s}{N_o}}\right) \\ &= 2^{(m+1)l} Q\left(\sqrt{\frac{2(dw_1)E_s}{N_o}}\right) \quad \text{(B.7)} \end{aligned}$$

and we have

$$P_{dw_1} \leq 2^{(m+1)l} Q\left(\sqrt{\frac{2d_f w_1 E_s}{N_o}}\right) e^{\frac{d_f w_1 E_s}{N_o}} e^{-\frac{dw_1 E_s}{N_o}} \quad \text{(B.8)}$$

where we used the relation (see for example [19, pp. 247])

$$Q(\sqrt{x+y}) \;\leq\; Q(\sqrt{x})e^{-y/2}, \quad x \geq 0, y \geq 0 \tag{B.9}$$

Defining $f(x) \equiv Q(\sqrt{2x})e^{x}$, we have

$$P_{dw_1} \;\leq\; 2^{(m+1)l} f\left(\frac{d_f w_1 E_s}{N_o}\right) e^{-dw_1 E_s/N_o} \tag{B.10}$$

The sums of (3.28) or of (3.33) can now be expressed in terms of the modified transfer function $T(D, L, I)$.

1. Indeed

$$\sum_{i=1}^{\infty} d_i P_i \;\leq\; f\left(\frac{d_f w_1 E_s}{N_o}\right) \sum_{l=d_f}^{\infty} \sum_{d=d_f}^{\infty} d2^{(m+1)l} c(d,l) e^{-dw_1 E_s/N_o} \tag{B.11}$$

$$= f\left(\frac{d_f w_1 E_s}{N_o}\right) \left[D \frac{\partial T(D,L,I)}{\partial D}\right]_{\substack{D=e^{-w_1 E_s/N_o} \\ I=1, L=2^{m+1}}} \tag{B.12}$$

2. Similarly,

$$\sum_{i=1}^{\infty} l_i P(i) \;\leq\; f\left(\frac{d_f w_1 E_s}{N_o}\right) \sum_{l=d_f}^{\infty} \sum_{d=d_f}^{\infty} l\, 2^{(m+1)l} c(d,l) e^{-dw_1 E_s/N_o} \tag{B.13}$$

$$= f\left(\frac{d_f w_1 E_s}{N_o}\right) \left[L \frac{\partial T(D,L,I)}{\partial L}\right]_{\substack{D=e^{-w_1 E_s/N_o} \\ I=1, L=2^{m+1}}} \tag{B.14}$$

3. And finally, in the same way,

$$\sum_{i=1}^{\infty} \beta_i P(i) \;\leq\; f\left(\frac{d_f w_1 E_s}{N_o}\right) \sum_{l=d_f}^{\infty} \sum_{d=d_f}^{\infty} \sum_{j=1}^{\infty} jg(j,d,l) 2^{(m+1)l} e^{-dw_1 E_s/N_o} \tag{B.15}$$

$$= f\left(\frac{d_f w_1 E_s}{N_o}\right) \left[\frac{\partial T(D,L,I)}{\partial I}\right]_{\substack{D=e^{-w_1 E_s/N_o} \\ I=1, L=2^{m+1}}} \tag{B.16}$$

Naturally, (B.12-B.16) are valid provided the infinite sums converge. Putting it all together yields

$$P_b \;\leq\; \frac{m+1}{m+b+1} p_{RM} + \frac{m+1}{m+b+1} f\left(\frac{d_f w_1 E_s}{N_o}\right) \left\{ (p_{\text{bent}} - p_{RM}) \left[D \frac{\partial T(D,L,I)}{\partial D}\right]_{\substack{D=e^{-w_1 E_s/N_o} \\ I=1, L=2^{m+1}}} \right.$$

$$\left. + p_{RM} \left[L \frac{\partial T(D,L,I)}{\partial L}\right]_{\substack{D=e^{-\frac{w_1 E_s}{N_o}} \\ I=1, L=2^{m+1}}} + \frac{1}{m+1} \left[\frac{\partial T(D,L,I)}{\partial I}\right]_{\substack{D=e^{-\frac{w_1 E_s}{N_o}} \\ I=1, L=2^{m+1}}} \right\} \tag{B.17}$$

B.3 Upper Bound for TRM1 Codes

With TRM1 codes, the trellis structure and the coset leader mapping ensure that there can be no path with a diverging length greater than its modified weight. Indeed the only trellis branches of zero modified weight link a zero state to another zero state. Hence,

$$c(d,l) = \begin{cases} a(d) & d = l \\ 0 & \text{Otherwise} \end{cases} \tag{B.18}$$

and $h(d,l) = 0$ for $d \neq l$. Noting that $d_f = 2$, (B.6) reduces to

$$P_b \leq \frac{m+1}{m+b+1}\left[p_{RM} + p_{\text{bent}} \sum_{d=2}^{\infty} da(d) \sum_{i=0}^{d} A(i,d,d) P_{iw_1+(d-i)\,w_2}\right] + \frac{1}{m+b+1} \sum_{d=2}^{\infty} h(d,d) \sum_{i=0}^{d} A(i,d,d) P_{iw_1+(d-i)\,w_2} \tag{B.19}$$

The number of paths at modified distance d from the correct path is

$$a(d) = \begin{cases} 0 & d = 1 \\ (2^b - 1)^{d-1} & 2 \leq d \end{cases} \tag{B.20}$$

Indeed, for each of the first $d-1$ branches, there are $2^b - 1$ possible state transitions since the path cannot return to the zero state. The last state transition on the other hand, must return to the zero state.

The expressions for $c(d,l)$ and $a(d)$ are also easily derived from the transfer function of (3.45), which can be rewritten as

$$\begin{aligned} T(D,L,1) &= DL\frac{x}{1-x}\Big|_{x=DL(2^b-1)} = DL(DL(2^b-1) + D^2L^2(2^b-1)^2 + \cdots) \\ &= \sum_{d=2}^{\infty} D^d L^d (2^b - 1)^{d-1} \end{aligned} \tag{B.21}$$

verifying (B.20) and (B.18).

Let us now derive an expression for $h(d,d)$. First, without using the transfer function, we notice that the number of data sequences of length b and non-zero weight j is

$$\zeta(j) \equiv \begin{cases} \binom{b}{j} & 1 \leq j \leq b \\ 0 & \text{Otherwise} \end{cases} \tag{B.22}$$

Any path which has a modified distance of 2 from the zero path is generated by only one non-zero data block of b bits. Hence $g(j,2,2) = \zeta(j)$. A path with modified distance 3 is generated by two consecutive non-zero data blocks of b bits. Thus,

$$g(j,3,3) = \begin{cases} \sum_{k=1}^{j} \binom{b}{j-k}\binom{b}{k} & 2 \le i \le 2b \\ 0 & \text{Otherwise} \end{cases} \qquad \text{(B.23)}$$

In general,

$$g(j,d+1,d+1) = \begin{cases} \sum_{k=1}^{j} g(k,d)\zeta(j-k) & d \le j \le bd \\ 0 & \text{Otherwise} \end{cases} \qquad \text{(B.24)}$$

Thus $g(j,d+1,d+1)$ is obtained from the convolution of $g(j,d,d)$ with $\zeta(j)$. One can view $p_X(x) \equiv \frac{\zeta(j)}{2^b-1}$ as the probability mass function (p.m.f.) of a discrete random variable X. Then $\frac{g(j,d,d)}{(2^b-1)^{d-1}}$ is the p.m.f. of $Y = \sum_{i=1}^{d-1} X_i$ where X_i are i.i.d. random variables with p.m.f. $p_X(x)$. Hence,

$$\begin{aligned} h(d,d) &= \sum_{j=1}^{\infty} jg(j,d,d) = \sum_{j=d}^{bd} jg(j,d,d) = (2^b-1)^{d-1}E[Y] = (2^b-1)^{d-1}E\left[\sum_{j=1}^{d-1} X_j\right] \\ &= (2^b-1)^{d-1}(d-1)E[X_1] = (2^b-1)^{d-1}d\frac{b2^{b-1}}{2^b-1} \\ &= (2^b-1)^{d-2}(d-1)b2^{b-1} \end{aligned} \qquad \text{(B.25)}$$

A direct long division from (3.45) can also be performed to derive the expression of $h(d,d)$. Indeed,

$$\begin{aligned} \left.\frac{\partial T(D,L,I)}{\partial I}\right|_{I=1} &= b2^{b-1}\frac{D^2L^2}{[1-DL(2^b-1)]^2} \\ &= \frac{b2^{b-1}D^2L^2}{2^b-1}\left(\frac{\partial}{\partial x}\frac{1}{1-ax}\right)_{x=DL,a=2^b-1} \\ &= \frac{b2^{b-1}D^2L^2}{2^b-1}(a + 2a^2x + 3a^3x^2 + \cdots + ia^ix^{i-1} + \cdots)_{x=DL,a=2^b-1} \\ &= b2^{b-1}\sum_{d=2}(d-1)(2^b-1)^{d-2}(DL)^d \end{aligned} \qquad \text{(B.26)}$$

verifying (B.25). Substituting in (B.19),

$$\begin{aligned} P_b \le\ & \frac{m+1}{m+b+1}p_{RM} + \frac{m+1}{m+b+1}p_{\text{bent}}\sum_{d=d_{mf}}^{\infty} d(2^b-1)^{d-1}\sum_{i=0}^{d} A(i,d,d)P_{iw_1+(d-i)w_2} \\ & + \frac{b2^{b-1}}{m+b+1}\sum_{d=d_{mf}}^{\infty}(d-1)(2^b-1)^{d-2}\sum_{i=0}^{d} A(i,d,d)P_{iw_1+(d-i)w_2} \end{aligned} \qquad \text{(B.27)}$$

B.4 Matrices in the State Equations of TRM2 Codes

We give here the explicit form of the matrices and vectors in (3.55–3.59)), in terms of the dummy variables D, L, and I. We denote by S_i the state specified by the state vector $\underline{\nu}$ such that the base-2 representation of i is $(\nu_0\nu_1\cdots\nu_b)_2$. We define the function $\underline{\sigma}_n(i)$ which associates to each decimal integer i the last n bits of its base-2 representation, in vector form. Formally,

$$\begin{aligned} \underline{\sigma}_n(i) &= (\sigma_0, \sigma_1, \cdots, \sigma_{n-1}) \\ \Leftrightarrow \quad i \bmod 2^n &= \sum_{j=0}^{n-1} \sigma_j 2^{n-1-j} \end{aligned} \tag{B.28}$$

Naturally, if $i < 2^{n-1}$, then $\sigma_0 = 0$ and so on. For example, $\underline{\sigma}_4(3) = (0, 0, 1, 1)$, and $\underline{\sigma}_4(18) = (0, 0, 1, 0)$. From this definition, we have that $\underline{\sigma}_{b-1}(i)$ is the state vector corresponding to S_i. Let k and l be such that $\underline{\sigma}_{b-1}(k) = \underline{\nu}$and $\underline{\sigma}_b(l) = \underline{a}$. Then

$$\underline{f}(\underline{a}, \underline{\nu}) = (\nu_0, \underline{\sigma}_{b-1}(\lfloor l/2 \rfloor) - \underline{\sigma}_{b-1}(k), a_b - \nu_1) \tag{B.29}$$

If we start off in state S_0, any non-zero input vector $\underline{a} = (a_1, \cdots, a_b)$, generates a branch with a modified weight of 1. Moreover, this branch goes from S_0 to state S_n, such that $\underline{\sigma}_{b-1}(n) = \underline{a}$. The Hamming weight of the input sequence generating a branch from S_0 to S_n is then given by $\omega(n) \equiv w_H(\underline{\sigma}_{b-1}(n))$. From this, we have

$$\begin{aligned} [\underline{A}_0]_i &= DLI^{\omega(i)}, \qquad i = 1, \cdots, 2^{b-1} - 1 &\quad (B.30) \\ \Rightarrow \quad \underline{A}_0 &= [DLI, DLI, DLI^2, DLI, \cdots]^T &\quad (B.31) \end{aligned}$$

Similarly, in going from S_0 to S_i:

$$\begin{aligned} [\underline{A}_1]_i &= DLI^{\omega(i)+1}, \qquad i = 0, \cdots, 2^{b-1} - 1 &\quad (B.32) \\ \Rightarrow \quad \underline{A}_1 &= [DLI, DLI^2, DLI^2, DLI^3, DLI, \cdots]^T &\quad (B.33) \\ &= [DLI, \underline{A}_0^T]^T &\quad (B.34) \end{aligned}$$

Consider now going from some state S_i of U_0 or U_2 to S_0'. The input vector is the all-zero vector. The output depends on the specific state S_i, but its modified weight is 1 regardless of the particular S_i. Hence

$$\underline{B}_0 = [\underbrace{DL, DL, \cdots, DL}_{2^{b-1}-1}]^T \tag{B.35}$$

and

$$\underline{B}_2 = [\underbrace{DL, DL, \cdots, DL}_{2^{b-1}}]^T = [DL, \underline{B}_0^T]^T \tag{B.36}$$

Consider now a transition from $S_i^{(u)} \in U_u$ to $S_j^{(v)} \in U_v$. We have $0 \leq i < 2^{b-1}$, $0 \leq j < 2^{b-1}$, and $(u, v) \in \{(0,0), (0,1), (1,2), (1,3), (2,1), (2,0), (3,2), (3,3)\}$. Let $\underline{\nu}$ be the state vector corresponding to $S_i^{(u)}$, and $\underline{a}$ be the input generating the transition from $S_i^{(u)}$ to $S_j^{(v)}$. Then $\underline{\nu} = \underline{\sigma}_{b+1}(i + u2^{b-1})$ and $\underline{a} = \underline{\sigma}_b(j + v2^{b-1})$. For this transition, the Hamming weight of the input vector is

$$w_{\mathrm{H}}(\underline{a}) = w_{\mathrm{H}}\left(\underline{\sigma}_b(j + v2^{b-1})\right) = \begin{cases} 1 + \omega(j) & v = 1, 3 \\ \omega(j) & v = 0, 2 \end{cases} \tag{B.37}$$

The corresponding trellis code output vector is zero if

$$\begin{aligned} \underline{f}(\underline{a}, \underline{\nu}) = 0 \quad &\Leftrightarrow \begin{cases} j + v2^{b-1} = 2(i + u2^{b-1}) & \text{and} \quad i + u2^{b-1} < 2^{b-1} & \text{if } l \text{ is even} \\ j + v2^{b-1} = 2(i + u2^{b-1}) + 1 & \text{and} \quad 2^{b-1} \leq i + u2^{b-1} < 2^b & \text{if } l \text{ is odd} \end{cases} \\ &\Leftrightarrow \begin{cases} j + v2^{b-1} = 2i & \text{and} \quad u = 0 & \text{if } l \text{ is even} \\ j + v2^{b-1} = 2i + 2^b + 1 & \text{and} \quad u = 1 & \text{if } l \text{ is odd} \end{cases} \end{aligned} \tag{B.38}$$

Note that if $v \geq 2$, then $j + v2^{b-1} \geq 2^b$, whereas $2i < 2^b$. Also, if $v < 2$, then $j + v2^{b-1} < 2^b$, whereas $2i + 2^b + 1 \geq 2^b + 1$. Hence, for a transition from $S_i^{(u)}$ to $S_j^{(v)}$, the output of the trellis code is zero if

$$\begin{cases} j + v2^{b-1} = 2i & u = 0 \quad v < 2 \quad j \text{ even} \\ j + v2^{b-1} = 2i + 2^b + 1 & u = 1 \quad v \geq 2 \quad j \text{ odd} \end{cases} \tag{B.39}$$

Given (B.37) and (B.38), one can now construct all the matrices T_{uv}.

$$[T_{00}]_{ji} = \begin{cases} LI^{\omega(j)} & j = 2i \\ DLI^{\omega(j)} & \text{Otherwise} \end{cases} \qquad 0 < i < 2^{b-1}, 0 < j < 2^{b-1} \tag{B.40}$$

$$[T_{01}]_{ji} = \begin{cases} LI^{\omega(j)-1} & j + 2^{b-1} = 2i \\ DLI^{\omega(j)+1} & \text{Otherwise} \end{cases} \qquad 0 < i < 2^{b-1}, 0 \le j < 2^{b-1} \quad \text{(B.41)}$$

$$[T_{12}]_{ji} = \begin{cases} LI^{\omega(j)} & j = 2i + 1 \\ DLI^{\omega(j)} & \text{Otherwise} \end{cases} \qquad 0 \le i < 2^{b-1}, 0 \le j < 2^{b-1} \quad \text{(B.42)}$$

$$[T_{13}]_{ji} = \begin{cases} LI^{\omega(j)-1} & j + 2^{b-1} = 2i + 1 \\ DLI^{\omega(j)-1} & \text{Otherwise} \end{cases} \qquad 0 \le i < 2^{b-1}, 0 \le j < 2^{b-1} \quad \text{(B.43)}$$

and for $u > 1$,

$$[T_{uv}]_{ji} = [T_{uv}]_j = \begin{cases} DLI^{\omega(j)} & v = 0, 2 \\ DLI^{\omega(j)-1} & v = 1, 3 \end{cases} \quad \text{(B.44)}$$

or, more precisely,

$$[T_{20}]_{ji} = DLI^{\omega(j)} \qquad 0 \le i < 2^{b-1}, \quad 0 < j < 2^{b-1} \quad \text{(B.45)}$$

$$[T_{21}]_{ji} = DLI^{\omega(j)-1} \qquad 0 \le i < 2^{b-1}, \quad 0 \le j < 2^{b-1} \quad \text{(B.46)}$$

$$[T_{32}]_{ji} = DLI^{\omega(j)} \qquad 0 \le i < 2^{b-1}, \quad 0 \le j < 2^{b-1} \quad \text{(B.47)}$$

$$[T_{33}]_{ji} = DLI^{\omega(j)-1} \qquad 0 \le i < 2^{b-1}, \quad 0 \le j < 2^{b-1} \quad \text{(B.48)}$$

Appendix C

In this appendix we give the simulation results used in our plots. The lowest bit error rate estimates are not very reliable, because of the larger standard deviation

	TRM1 Code, $b = 2, m = 6$				
E_b/N_0 (dB)	0.0	1.15	2.3	3.45	4.6
$p_{b,\text{trellis}}$	$3.1\ 10^{-2}$	$4.7\ 10^{-3}$	$3.0\ 10^{-4}$	$1.0\ 10^{-5}$	$1.6\ 10^{-7}$
p_{m+1}	$6.0\ 10^{-2}$	$1.5\ 10^{-2}$	$2.3\ 10^{-3}$	$2.1\ 10^{-4}$	$8.1\ 10^{-6}$
p_{average}	$5.4\ 10^{-2}$	$1.3\ 10^{-2}$	$1.8\ 10^{-3}$	$1.7\ 10^{-4}$	$6.4\ 10^{-6}$

Table C.1: Bit error rates of TRM1 code with $b = 2, m = 6$

	TRM2 Code, $b = 4, m = 6$			
E_b/N_0 (dB)	0.0	1.33	2.67	4.0
$p_{b,\text{trellis}}$	$7.7\ 10^{-2}$	$4.0\ 10^{-3}$	$4.7\ 10^{-5}$	$5.2\ 10^{-8}$
p_{m+1}	$6.7\ 10^{-2}$	$5.9\ 10^{-3}$	$2.2\ 10^{-4}$	$5.4\ 10^{-6}$
p_{average}	$7.1\ 10^{-2}$	$5.2\ 10^{-3}$	$1.6\ 10^{-4}$	$3.4\ 10^{-6}$

Table C.2: Bit error rates of TRM2 code with $b = 4, m = 6$

	TRM2 Code, $b = 8, m = 12$			TRM2 Code, $b = 9, m = 12$		
E_b/N_0 (dB)	0.0	0.9	1.8	0.0	0.7	1.4
$p_{b,\text{trellis}}$	$2.2\ 10^{-2}$	$5.0\ 10^{-4}$	$3.1\ 10^{-6}$	$1.9\ 10^{-2}$	$9.3\ 10^{-4}$	$1.0\ 10^{-5}$
p_{m+1}	$1.7\ 10^{-2}$	$7.2\ 10^{-4}$	$2.4\ 10^{-5}$	$1.4\ 10^{-2}$	$1.1\ 10^{-3}$	$4.0\ 10^{-5}$
p_{average}	$1.9\ 10^{-2}$	$6.4\ 10^{-4}$	$1.6\ 10^{-5}$	$1.6\ 10^{-2}$	$1.0\ 10^{-3}$	$2.8\ 10^{-5}$

Table C.3: Bit error rates of TRM2 code with $m = 12$, $b = 8$ and 9.

Bibliography

[1] T.S. Rappaport. *Wireless Communications. Principles and Practice.* IEEE Press. Prentice-Hall PTR, New Jersey, 1996

[2] W.C.Y. Lee, *Mobile Cellular Telecommunications Systems.* McGraw-Hill. 1989

[3] S. Verdú, "Minimum probability of error for asynchronous Gaussian multiple-access channels", *IEEE Trans. Inform. Theory.* vol IT-32, pp. 85-96, January 1986

[4] R. Lupas and S. Verdú, "Linear multiuser detectors for synchronous code-division multiple-access channels", *IEEE Trans. Inform. Theory.* Vol IT-35, No 1. pp. 123-136. January 1989

[5] M.K. Varanasi and B. Aazhang, "Multistage detection in asynchronous Code-Division Multiple-Access communications", *IEEE Trans. on Comm.*, Vol COM-38, pp. 509-519. April 1990

[6] W.C.Y. Lee, "Overview of cellular CDMA", *IEEE Transactions on Vehicular Technology*, pp. 291-302, May 1991

[7] Interim Standard 95 (IS-95)

[8] C.F. Kou, H. Leib, "Power imbalance effects on packet CDMA", to appear in *IEEE Transactions Communications*

[9] A.J. Viterbi, "Spread spectrum communications - Myths and realities", *IEEE Comm. Magazine*, Vol. 17, No. 3, pp. 11-18, May 1979

[10] "Benefits of error control coding for CDMA", J.P. Chaib and H. Leib, *Proceedings of the 17th Biennial Symposium on Communications*, Kingston, pp. 88-91, May 1994

[11] Joseph Y. Hui, "Throughput analysis for Code-Division Multiple Accessing of the spread spectrum channel", *IEEE J. Selected Areas Comm.*, Vol. SAC-2, pp. 482-486, July 1984

[12] A.J. Viterbi, "Very low rate convolutional codes for maximum theoretical performance of spread-spectrum multiple-access channels", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 4, pp. 641-649, May 1990.

[13] John G. Proakis, *Digital Communications*, 2nd Edition, McGraw-Hill, 1989

[14] P.M. Woodward, *Probability and Information Theory, with Applications to Radar*, Pergamon Press, Oxford, 1953

[15] C.E. Cook and M. Bernfeld, *Radar Signals: An Introduction to Theory an Application*. Academic Press, New York. 1967

[16] Frank Amoroso, "The bandwidth of digital data signals", *IEEE Communications Society Magazine*, Vol. 18, No. 6, pp. 13-24, November 1980

[17] J.P. Chaib and H. Leib, "Chip shaping and channel coding for CDMA", *European Transactions on Telecommunications and Related Technologies*, Vol 17, No 2, pp. 133-143, March-April 1996

[18] F.J. MacWilliams and N.J.A. Sloane, *The Theory of Error-Correcting Codes*. North-Holland Mathematical Library, 1977

[19] A.J. Viterbi and J.K. Omura, *Principles of Digital Communication and Coding*, McGraw-Hill, 1979

[20] M. B. Pursley, "Performance analysis for phase-coded spread spectrum multiple access communication - Part I: System analysis", *IEEE Trans. Comm*, Vol. COM-25, No 8, pp. 795-799, August 1977

[21] K. L. Gilhousen, I.M. Jacobs, R. Padovani, A.J. Viterbi, L. A. Weaver, Jr., and C.E. Wheatley III, "On the capacity of a cellular CDMA system", *IEEE Trans. Vehic. Techn.*, Vol. 40, No. 2, May 1991

[22] P.K. Enge and D.V. Sarwate, "Spread-spectrum multiple-access performance of orthogonal codes: linear receivers", *IEEE Trans. Comm.* , Vol. COM-35, pp. 1309–1319, December 1987

[23] R. K. Morrow and J. S. Lehnert, "Bit-to-bit error dependence in slotted DS/SSMA packet systems with random signature sequences", *IEEE Trans. Comm.* , Vol. COM-37, No. 10, pp. 1052–1061. October 1989

[24] J. K. Holtzman, "A simple, accurate method to calculate spread-spectrum multiple-access error probabilities", *IEEE Trans. Comm.*, Vol. COM-40, No. 3, pp. 461–464, March 1992

[25] A.J. Viterbi, "Orthogonal tree codes for communications in the presence of white Gaussian noise", *IEEE Trans. Commun. Technol.*, Vol. COM-15, pp. 238-242, April 1967

[26] K. Rikkinen, "Comparison of very low rate coding methods for CDMA radio communications systems", Third International Symposium on Spread Spectrum Techniques and Applications, University of Oulu, Oulu, Finland, July 1994

[27] A.J. Viterbi, *Principles of Spread Spectrum Communications*, Addison-Wesley Wireless Communications Series, 1994

[28] W.C. Lindsey and M.K. Simon, *Telecommunication Systems Engineering*, Dover Publications, New York, 1971

[29] E. C. Posner, "Combinatorial structures in planetary reconnaissance", in H.B. Mann, ed., *Error Correcting Codes*, Wiley, New York, pp. 15–46, 1969

[30] N. Ahmed. K.R. Rao. *Orthogonal Transforms for Digital Processing*. Springer-Verlag. Berlin. 1975

[31] A.V. Oppenheim and R.W. Shafer, *Discrete-Time Signal Processing*. Prentice-Hall. New Jersey, 1989

[32] R. E. Blahut. *Digital Transmission of Information*. Addison-Wesley. 1990

[33] S. Lin and D.J. Costello. *Error Control Coding: Fundamentals and Applications*, Prentice-Hall. 1983

[34] "Trellis/Reed-Muller codes for CDMA". J.P. Chaib and H. Leib. *Proceedings of the sixth International Conference on Personal, Indoors, and Mobile Radio Communications (PIMRC'95)*. pp. 243-247. September 1995

[35] A. M. Kerdock. "A class of low-rate nonlinear binary codes". *Information and Control*. Vol. 20, pp. 182–187, 1972

[36] G. E. Bottomley, "Signature sequence selection in a CDMA system with orthogonal coding", *IEEE Transactions on Vehicular Technology*, Vol-42. No 1. pp. 62-68. February 1993

[37] G. Ungerboeck. "Channel coding with multilevel/phase signals", *IEEE Transactions on Information Theory*, Vol. IT-28. No. 1, pp. 55-67. January 1982

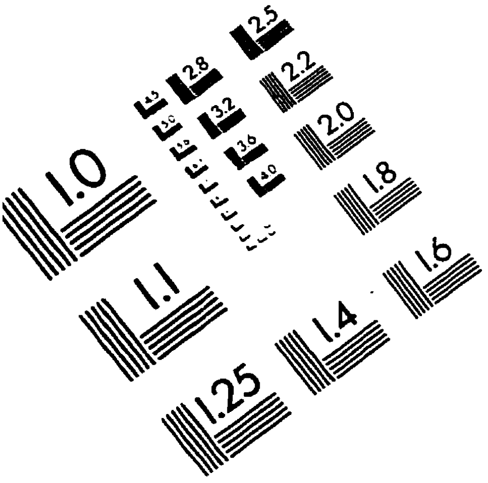
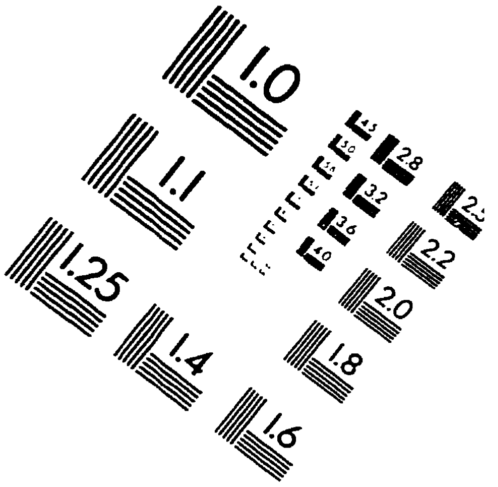
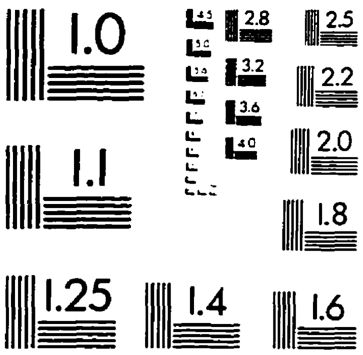
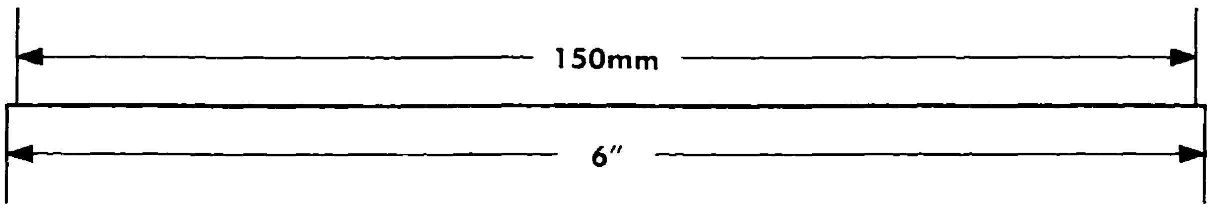
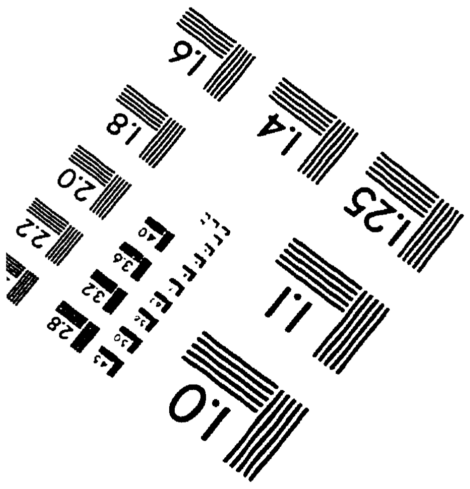
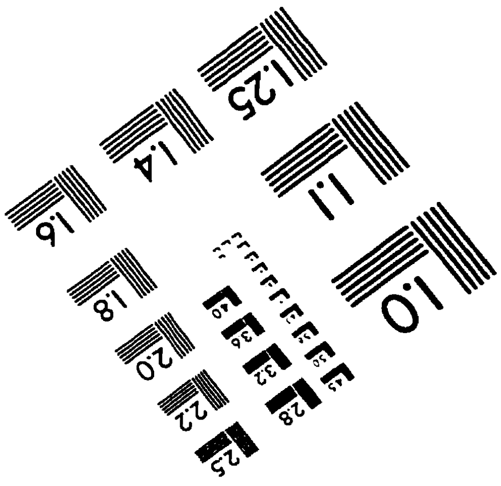
[38] G. Ungerboeck, "Trellis-coded modulation with redundant signal sets Part I: Introduction". *IEEE Communications Magazine*. Vol. 25. No. 2,pp. 5-11. February 1987

[39] E. Biglieri, D. Divsalar, P.J. McLane, M.K. Simon. *Introduction to Trellis-Coded Modulation with Application*, Macmillan Publishing Company. New York. 1991

[40] G.C. Clark, Jr. and J.B. Cain, *Error Correction Coding for Digital Communications*. Plenum Press. New York. 1981

[41] J.P. Adoul. "Fast ML decoding algorithm for the Nordstrom-Robinson code". *IEEE Transactions on Information Theory*, Vol. IT-33. No. 6. pp. 931-933. November 1987

IMAGE EVALUATION
TEST TARGET (QA-3)

APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989