GODDARD
GRANT
IN-32-CR
142709
P-140

[redacted]

Department of

ELECTRICAL AND COMPUTER ENGINEERING

UNIVERSITY OF NOTRE DAME, NOTRE DAME, INDIANA

Annual Status Report

Error Control Techniques for Satellite
and Space Communications
NASA Grant NAG5-557
June 1988

Daniel J. Costello, Jr.
Department of Electrical and Computer Engineering
University of Notre Dame
Notre Dame, IN 46556

Summary of Progress

During the period December 1, 1987 - May 31, 1988, progress was made in the following areas:

1) Construction of Multi-Dimensional Bandwidth Efficient Trellis Codes with MPSK Modulation.

Multi-dimensional trellis coded modulation schemes using either 8PSK or 16PSK modulation appear to have great promise for achieving high data dates on satellite communication channels. Work by Ungerboeck [1,2], Hemmati and Fang [3], Fujino et.al. [4], and others has demonstrated that 2-dimensional (one signal/time unit) rate 2/3 (3/4) trellis coded 8PSK (16PSK) modulation is capable of achievilng data rates in excess of 100 Mbps on satellite channels. The promise of even higher rates is possible with multi-dimensional trellis coded schemes. For example, with 2L-dimensional schemes, $L \geq 2$, where L signals are transmitted per time unit, speeds of up to L times those achievable with 2-dimensional schemes may be possible. This depends on fast computational techniques being developed to compute the metric of L successive signals on a trellis branch in the Viterbi algorithm. For moderate values of L ($L \leq 4$), this seems feasible using table look-up methods.

We have conducted an extensive search for good multi-dimensional trellis codes with $2 \leq L \leq 4$ for both 8PSK and 16PSK modulation. These codes achieve coding gains (over uncoded transmission at the same rate) of up to 5.5 dB. In addition, many of the codes are fully transparent to discrete phase rotations of the signal set (45° transparency for 8PSK and 22.5° transparency for 16PSK) through the use of differential encoding. A paper summarizing our work in this area has been accepted for publication by the IEEE Transactions on Information Theory and is included as Appendix A of this report [5].

It is recommended that NASA proceed with the development of one of these codes for their high speed satellite transmission schemes of the future. A good choice would be the six-dimensional ($L = 3$), 16-state, rate 7/8, 8PSK code listed in Table 9(b) of the paper. This code has a 3.57 dB coding gain compared to uncoded modulation of the same rate and is transparent to 90° phase rotations of the signal set. With proper decoder implementation, this code would be capable of operating at three times ($L = 3$) the speed of a comparable two-dimensional code. In terms of current technology, this offers the possibility of reliable transmission at speeds in excess of 300 Mbps.

2) Performance Analysis of Bandwidth Efficient Trellis Coded Modulation Schemes

Most of the bandwidth efficient trellis code constructions which have been published in the literature measure performance with a parameter d^2_{free}, the minimum free squared Euclidean distance of the code. This is determined by the two codewords (signal sequences) which are closest together in terms of squared Euclidean distance. This parameter determines the asymptotic (high signal-to-noise ratio) coding gain γ of the system through the formula

$$\gamma = 10 \log_{10} \frac{d^2_{free}}{d^2_u},$$

where d_u^2 is the minimum squared Euclidean distance of an uncoded system with the same rate.

Unfortunately, γ or d_{free}^2 may not give a very accurate picture of relative code performance at more moderate signal-to-noise ratios (SNR's), where most practical systems operate. In particular, for SNR's which result in decoded bit error rates of around $10^{-4}-10^{-6}$, the asymptotic coding gain may be a poor estimate of code performance. This effect, which is also true for convolutional codes with binary modulation, seems to be more pronounced for bandwidth efficient trellis codes due to increased numbers of nearest neighbors. We have found that in order to accurately determine performance for bandwidth efficient trellis codes, it is necessary to find not only the minimum free distance but several of the next highest distances. This involves considerably more computation than just finding the minimum free distance.

Another problem with determining the performance of trellis coded modulation schemes is that the codes are not linear, due to the non-linear mapping from encoder outputs into signal points. This makes the determination of the code distances much more involved than for linear codes, since we can no longer assume that the all-zero codeword was transmitted. Indeed, the computation of a distance spectrum for a non-linear trellis code must involve an average over all possible transmitted codewords.

The above difficulties notwithstanding, we have been able to develop an efficient algorithm for determining the distance spectrum of trellis codes. A paper based on this algorithm has been submitted to the *IEEE Journal on Selected Areas in Communications* and is included as Appendix B of this report [6]. Using this algorithm, we can obtain an accurate performance estimate for most of the best known trellis coded modulation schemes.

3) Performance Analysis of Bandwidth Efficient Trellis Codes on Fading Channels.

In the area of mobile satellite communications, it is necessary to use coding techniques which are designed to combat signal fading. For binary coding, this simply involves the use of interleaving. For bandwidth efficient codes using MPSK modulation, however, it has been shown by Hagenauer et.al. [7], Hagenauer and Lutz [8], and Simon and Divsalar [9] that codes designed for the AWGN channel will not perform well on a fading channel, even with interleaving.

We have derived performance bounds for bandwidth efficient trellis codes on Rayleigh and Rician fading channels. These bounds show that two new parameters, the effective length and the minimum product distance, are more important than the free distance and the path mullti-plicity when designing codes for fading channels. New trellis codes for fading channels with 8PSK modulation have been constructed, and it is shown that these codes outperform codes of the same complexity designed for the AWGN channel. A paper summarizing these results has been submitted to the *IEEE Journal on Selected Areas in Communications* and is included as Appendix C of this report [10].

References

[1] G. Ungerboeck, "Channel Coding with Multilevel Phase Signals," *IEEE Transactions on Information Theory*, IT-28, pp. 55-67, Jan. 1982.

[2] G. Ungerboeck, "Trellis Coded Modulation with Redundant Signal Sets - Part I: Introduction and Part II: State of the Art," *IEEE Communications Magazine*, Vol. 25, no. 2, pp. 5-21, Feb. 1987.

[3] F. Hemmati and R. J. F. Fang, "Low Complexity Coding Methods for High Data Rate Channels," *Comsat Technical Review*, Vol. 16, pp. 425-447, Fall 1986.

[4] T. Fujino et.al., "A 120 Mbits/s Coded 8PSK Modem with Soft-Decision Viterbi Decoder," *IEEE International Conference on Communications Conference Record*, pp. 1774-1780, Toronto, Canada, June 1986.

[5] Robert H. Deng, Steven S. Pietrobon, Alain Lafanechére, Gottfried Ungerboeck, and Daniel J. Costello, Jr., "Multi-Dimensional Trellis Coded Phase Modulation," *IEEE Transactions on Information Theory*, to appear.

[6] Marc Rouanne and Daniel J. Costello, Jr., "An Algorithm for Computing the Distance Spectrum of Trellis Codes," submitted to *IEEE Journal on Selected Areas in Communications.*

[7] J. Hagenauer et.al., "The Maritime Satellite Communication Channel," *IEEE Journal on Selected Areas in Communications*, Vol. SAC-5, No. 4, pp. 701-713, May 1987.

[8] J. Hagenauer and E. Lutz, "Forward Error Correction Coding for Fading Compensation in Mobile Satellite Channels," *IEEE Journal on Selected Areas in Communications*, Vol. SAC-5, No. 2, pp. 215-225, February 1987.

[9] M. K. Simon and D. Divsalar, "Trellis Coded Modulation for 4800-9600 bits/s Transmission Over a Fading Mobile Satellite Channel," *IEEE Journal on Selected Areas in Communications*, Vol. SAC-5, No. 2, pp. 162-177, February 1987.

[10] Christian Schlegel and Daniel J. Costello, Jr., "Bandwidth Efficient Coding for Fading Channels," submitted to *IEEE Journal on Selected Areas in Communications.*

Appendix A

Multi-Dimensional Trellis Coded Phase Modulation

Multi-Dimensional Trellis Coded Phase Modulation*

Robert H. Deng,† Steven S. Pietrobon,‡
Alain LaFanéchere,§ Gottfried Ungerboeck,¶
and Daniel J. Costello, Jr.‖

May 12, 1988

Abstract

In this paper, multi-dimensional trellis coded MPSK modulation is investigated. A $2L$-dimensional ($L \geq 2$) MPSK signal set is obtained by forming the Cartesian product of L 2-dimensional MPSK signal sets. A systematic approach to partitioning multi-D signal sets is used which is based on block coding. An encoder system design approach is developed which incorporates the design of a differential precoder, a systematic convolutional encoder, and a signal set mapper. Multi-dimensional trellis coded 8PSK and 16PSK modulation schemes are found, for a variety of rates and decoder complexities, many of which are fully transparent to discrete phase rotations of the signal set. Asymptotic coding gains up to 5.5 dB have been found for these codes.

*This work was supported by NASA Grant NAG5-557 and OTC(Australia) R&D Programme No. 4.

†Institute of Systems Science, National University of Singapore, Kent Ridge, Singapore 0511. Formerly with the Dept. of Electrical and Computer Engineering, University of Notre Dame.

‡Dept. of Electrical and Computer Engineering, University of Notre Dame, Notre Dame, IN, 46556, U.S.A. and School of Electronic Engineering, South Australian Institute of Technology, The Levels, P.O. Box 1, Ingle Farm S.A. 5098, Australia.

§Enertec Schlumberger, 1 Rue Nieuport, 78141 Velizy, France. Formerly with the Dept. of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, IL, 60616, U.S.A.

¶IBM Zurich Research Laboratory, Säumerstrasse 4, CH-8803 Rüschlikon, Switzerland.

‖Dept. of Electrical and Computer Engineering, University of Notre Dame, Notre Dame, IN, 46556, U.S.A.

1 Introduction

Since the publication of the paper by Ungerboeck [1], Trellis Coded Modulation (TCM) has become a very active research area [2–9]. The basic idea of TCM is that by trellis coding onto an expanded signal set (relative to that needed for uncoded transmission), both power and bandwidth efficient communication can be achieved.

TCM can be classified into two basic types, the lattice type (e.g., M-PAM, M-QASK) and the constant-envelope type (e.g., MPSK). The latter has a lower power efficiency compared with the former but is more suitable for band-limited satellite channels containing nonlinear amplifiers such as traveling wave tubes (TWT). Taylor and Chan [10] and Wilson et. al. [11] have studied the performance of rate 2/3 TC-8PSK and rate 3/4 TC-16PSK, respectively, for various channel bandwidths and TWT operating points. Their results showed that TC-MPSK modulation schemes are quite robust under typical channel conditions.

In any TCM design, partitioning of the signal set into subsets with increasing intra-subset minimum distances plays a central rule. It defines the signal mapping used by the modulator and provides a tight bound on the minimum free squared Euclidean distance (FSED) between code sequences, allowing an efficient search for optimum codes. For lattice-type TCM, Calderbank and Sloane [8] have made the important observation that partitioning the signal set into subsets corresponds to partitioning a lattice into a sublattice and its cosets. Forney [9] has developed a method, called the "squaring construction", of partitioning higher dimensional lattices from a lower dimensional lattice by using a coset code.

In this paper, we investigate a class of multi-dimensional (multi-D) trellis coded MPSK (TC-MPSK) modulation schemes. The $2L-$dimensional ($2L-$D) MPSK signal set is generated by simply repeating an MPSK signal set L times ($L \geq 2$). Therefore, the $2L-$D MPSK signal set is the Cartesian product of L 2–D MPSK signal sets. Multi-D MPSK signal sets provide us with a number of advantages that can't be found in a 2–D signal set: (i) flexibility in achieving higher effective information rates, (ii) better coding gains, (iii) easy construction of some codes which are invariant to phase rotations, and (iv) due to their byte oriented nature, suitability for use as inner codes in a concatenated coding system [12].

In Section 3, we introduce a block coding technique for partitioning a multi-D MPSK signal set. We will show that partitioning a $2L-$D MPSK signal set

is isomorphic to partitioning an $L \times \log_2 M$ binary matrix space. This section is mathematically rigourous. Thus, a brief description of the major ideas and concepts is given in Section 2 by way of an example. Section 4 describes how the encoder system, comprising a differential precoder, a systematic convolutional encoder, and a multi-D signal set mapper, is constructed from the best codes found in a systematic code search. The signal sets are constructed such that the codes are transparent to integer multiples of $360°/M$ rotations of the MPSK signal set. The systematic code search is based on maximizing the FSED (and thus the asymptotic coding gain) as well as minimizing the number of nearest neighbors for each phase transparency. 4–D, 6–D, and 8–D TC-8PSK codes and 4–D and 6–D TC-16PSK codes are listed with coding gains up to 5.5 dB compared to an uncoded system. In addition, these codes require no bandwidth expansion.

2 A Block Coding View of Set Partitioning

In this section we give a description of how partitioning a 2–D 8PSK signal set can be viewed in terms of block coding. This relatively simple example is used to describe the concepts used to partition multi–D MPSK signal sets in Section 3.

A naturally mapped 8PSK signal set is illustrated in Figure 1. The reason for using natural mapping is that the three mapped bits can be used directly to indicate the minimum squared subset distance (MSSD). If each of the three bits y^0, y^1, and y^2 is allowed to be 0 or 1, i.e., $y^j \in \{0,1\}$ for $j = 0, 1$ and 2, then there will be some combinations (e.g., 000 and 111) where the minimum possible distance between two points is achieved. In this case the MSSD will be $2 - \sqrt{2} \simeq 0.586$ if the average energy of the signal set is taken to be one. However, if we set $y^0 = 0$ and let $y^j \in \{0,1\}$ for $j = 1$ and 2, then the MSSD of the resulting subset will be 2. One can view this as y^0 belonging to a simple length one block code that has only one code word, i.e., 0. We say that the Hamming distance of this block code is infinity, since that is the distance required to reach all the other (non-existent) codewords. This length one block code concept can also be applied to the other two mapping bits y^1 and y^2. These can be thought of as uncoded cases where there are two code words, 0 and 1, and where the Hamming distance is one.

One may ask "What is the use of this block code description, when we have the much simpler description given by Ungerboeck [1]?". As will be seen, although this is a complicated description for the simple 2–D case, for higher dimensions this description yields a powerful and easy method of partitioning a multi-D signal set.

Now that we have described the signal set in terms of block codes, albeit they are trivial, we can use the equation for MSSD given by Sayegh [13] from work originally done by Cusack [14]. Before we give this equation, some notation is needed. Let d_j^H be the Hamming distance for the block codes corresponding to the bits y^j for $j = 0$, 1, and 2. Also let the MSSD that corresponds to setting $y^0, \ldots, y^{j-1}$ to 0 be δ_j^2 for $j = 0$, 1, and 2. We have already determined that $\delta_0^2 = 0.586$ and $\delta_1^2 = 2$. For the remaining MSSD, it can easily be shown that $\delta_2^2 = 4$. The 8PSK signal set can be seen to have three levels of partitioning. A parameter p is used to designate the partitioning level. The initial level of partitioning is denoted by $p = 0$. This corresponds to all eight 8PSK signal points. The first level of partitioning ($p = 1$), corresponds to a subset of four

points. This can be continued until we reach the final level with $p = 3$ and only a single point. From [16], the MSSD at partition level p is

$$\Delta_p^2 \geq \min(\delta_2^2 d_2^H, \delta_1^2 d_1^H, \delta_0^2 d_0^H).$$

Due to the symmetry of the 8PSK signal set, the lower bound is an equality. For $p = 0$, $d_2^H = d_1^H = d_0^H = 1$, and thus $\Delta_0^2 = 0.586$. For $p = 1$, $d_0^H = \infty$, and thus $\Delta_1^2 = 2$. Similarly $\Delta_2^2 = 4$ and $\Delta_3^2 = \infty$. Note that in this case $\Delta_p^2 = \delta_p^2$. However, for multi-D schemes, more complicated block codes are used where d_j^H can have values of 2, 3, or more. Thus, in some cases, $\Delta_p^2 \neq \delta_p^2$.

Note that the above example considers only a single branch of a partition. Going back to our original description, we can also set $y^0 = 1$, $y^1 \in \{0,1\}$, and $y^2 \in \{0,1\}$. Due to the symmetry of the 8PSK signal set the subset selected also has an MSSD of δ_1^2. A block code view of this subset is that it is the *coset* of the subset selected by $y^0 = 0$. The reason is that we can take the coset representative (which is 1) of the coset $\{1\}$ corresponding to the simple block code $\{0\}$ and add it modulo-2 to $y^0 = 0$, which selects the first subset, to obtain $y^0 = 1$, which selects the other subset. This coset representative is a codeword at the previous partition level, but is not a codeword at the current partition level. That is, the coset representative (1) belongs to the code $\{0,1\}$ at $p = 0$ but not to the code $\{0\}$ at $p = 1$. In a similar manner, codes corresponding to y^1 and y^2 can be partitioned using coset representatives until all 8 signal points belong to subsets containing only a single point. This is an important concept, since a multi-D signal set partition can be directly described by its cosets, right down to a single point, as will be shown in Section 3.

To obtain a multi-D signal set, one can view y^j as containing more than one bit. In fact, y^j becomes a vector $\mathbf{v}^j$ that corresponds to the jth bits of two or more signal sets. This vector $\mathbf{v}^j$ contains only one bit for a 2–D signal set, and thus the block codes have length one. However, for a $2L$-D signal set, there are L bits in the vector $\mathbf{v}^j$, which will belong either to a block code of length L or to one of its cosets, depending on which partition path is chosen. If there are $M = 2^I$ signals in each 2–D signal set, then there will be $I = \log_2 M$ sets of these codewords.

3 Multi-D MPSK Signal Set Partitioning

We begin this section with a discussion of partitioning a binary matrix space. We then show that partitioning a $2L$−D MPSK signal space is isomorphic to partitioning an $I = \log_2 M \times L$ binary matrix space.

3.1 Partitioning a binary matrix space

Let $\mathbf{C}_m$, with $m = 0, 1, \ldots, L$, be a sequence of $(L, L-m)$ binary linear block codes with generator matrices $\mathbf{G}_m$ and Hamming distances d_m such that $\mathbf{C}_L \subset \mathbf{C}_{L-1} \subset \cdots \mathbf{C}_1 \subset \mathbf{C}_0$. Denote the L−D binary vector space by $\mathbf{V}_L = \{0,1\}^L$. Then $\mathbf{C}_0 = \mathbf{V}_L$, and $\mathbf{C}_0/\mathbf{C}_1/\cdots/\mathbf{C}_{L-1}/\mathbf{C}_L$ forms a 2/2/.../2/2 (L times)-way binary vector space partition chain. The 2^m−way binary vector space partition, $\mathbf{C}_0/\mathbf{C}_m$, divides $\mathbf{C}_0$ into $\mathbf{C}_m$ and its $2^m - 1$ *cosets*, $\mathbf{C}_m(1)$, $\mathbf{C}_m(2), \ldots, \mathbf{C}_m(2^m - 1)$. Let $\mathbf{t}_m(u)$ be the *coset representative* of $\mathbf{C}_m(u)$, where u is the integer representation of the binary vector $\mathbf{u} = [u^{m-1}, \ldots, u^1, u^0]$, i.e., $u = 2^{m-1}u^{m-1} + \cdots + 2^1 u^1 + 2^0 u^0$. Then $\mathbf{C}_m(u)$ and $\mathbf{C}_m$ are related by

$$\mathbf{C}_m(u) = \mathbf{C}_m \oplus \mathbf{t}_m(u). \tag{1}$$

where $\oplus$ indicates modulo-2 addition.

The coset representative of $\mathbf{C}_m$ is the L−D all-zero vector and is denoted by $\mathbf{t}_m(0)$. Let $\underline{\tau}_m \triangleq \mathbf{t}_m(2^{m-1})$ be the coset representative such that

$$\underline{\tau}_m \in \mathbf{C}_{m-1},\ \underline{\tau}_m \notin \mathbf{C}_m.$$

We call $\underline{\tau}_m$ a *principle coset representative*, since these are the particular coset representatives which can be used to fully describe all the cosets. The mapping that we assume is that the first $m-1$ bits of $\mathbf{u}$ are all 0 so that $u^{m-1} = 0$ selects $\mathbf{C}_m$ and $u^{m-1} = 1$ selects the coset $\mathbf{C}_m(2^{m-1})$. Note that $\underline{\tau}_m$ can be any codeword in $\mathbf{C}_m(2^{m-1})$. Thus an expression for any coset representative is

$$\mathbf{t}_m(u) = u^{m-1}\underline{\tau}_m \oplus \cdots \oplus u^1\underline{\tau}_2 \oplus u^0\underline{\tau}_1 = \bigoplus_{j=0}^{m-1} u^j \underline{\tau}_{j+1}. \tag{2}$$

Example 3.1:

For the 2–D binary vector space $\mathbf{V}_2=\{0,1\}^2$, we may form the following block codes:

$$\mathbf{C}_0:\ (2,2)\text{ code, }\mathbf{G}_0=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix},\ d_0=1;$$
$$\mathbf{C}_1:\ (2,1)\text{ code, }\mathbf{G}_1=[1\quad 1],\ d_1=2;$$
$$\mathbf{C}_2:\ (2,0)\text{ code, }\mathbf{G}_2=[0\quad 0],\ d_2=\infty.$$

Note that $\mathbf{C}_2\subset\mathbf{C}_1\subset\mathbf{C}_0$. The 2/2–way partition chain $\mathbf{C}_0/\mathbf{C}_1/\mathbf{C}_2$, along with the 2–way partitions $\mathbf{C}_0/\mathbf{C}_1$ and $\mathbf{C}_1/\mathbf{C}_2$ and the 4–way partition $\mathbf{C}_0/\mathbf{C}_2$, is shown in Figure 2. The principle coset representatives are $\underline{\tau}_1=[0\ 1]^T$ and $\underline{\tau}_2=[1\ 1]^T$.

We now describe how I of the above block codes can be used to describe an $L\times I$ binary matrix space. Let $\mathbf{C}_{m_i}$ be an $(L,L-m_i)$ linear block code that is a subspace of $\mathbf{V}_L$, $m_i=0,1,\ldots,L$. Define $\Omega^p=\Omega(\mathbf{C}_{m_{I-1}},\ldots,\mathbf{C}_{m_1},\mathbf{C}_{m_0})$, where $p=\sum_{i=0}^{I-1}m_i$ is the level of partitioning, as the set of all $L\times I$ binary matrices:

$$\underline{\omega}=[\mathbf{v}^{I-1}\cdots\mathbf{v}^1\ \mathbf{v}^0]=\begin{bmatrix} v_0^{I-1} & \cdots & v_0^1 & v_0^0\\ v_1^{I-1} & \cdots & v_1^1 & v_1^0\\ \vdots & \ddots & \vdots & \vdots\\ v_{L-1}^{I-1} & \cdots & v_{L-1}^1 & v_{L-1}^0\end{bmatrix},\qquad(3)$$

where $\mathbf{v}^i\in\mathbf{C}_{m_i}$, $i=0,1,\ldots,I-1$, and each $\mathbf{v}^i$ is an L–dimensional column vector. Ω^p is a subspace of $\Omega^0=\Omega(\mathbf{V}_L,\ldots,\mathbf{V}_L,\mathbf{V}_L)$ and is a group under binary modulo-2 matrix addition. Ω^p is called the principle subset of Ω^0. Ω^0/Ω^p is a 2^p way binary matrix space partition which divides Ω^0 into Ω^p and its 2^p-1 cosets, $\Omega^p(z)=\Omega(\mathbf{C}_{m_{I-1}}(z),\ldots,\mathbf{C}_{m_0}(z))$, $1\le z\le 2^p-1$ where z is the integer representation of the binary vector $\mathbf{z}=[z^{p-1},\ldots,z^1,z^0]$. $\mathbf{C}_{m_i}(z)$ is either $\mathbf{C}_{m_i}$ or a coset of $\mathbf{C}_{m_i}$, depending on the partition level p and the particular value of z. The coset representative of $\Omega^p(z)$ is given by $\underline{\omega}^p(z)=\underline{\omega}(\mathbf{t}_{m_{I-1}}(z),\ldots,\mathbf{t}_{m_0}(z))$, where $\mathbf{t}_{m_i}(z)$ is the coset representative of $\mathbf{C}_{m_i}(z)$. The principle subset and its cosets are related by

$$\Omega^p(z)=\Omega^p\oplus\underline{\omega}^p(z).\qquad(4)$$

Ω^p is said to be a subspace of $\Omega^{p'}$ if and only if $p>p'$ and $\mathbf{C}_{m_i}\subseteq\mathbf{C}_{m_i'}$, $i=0,1,\ldots,I-1$. In this case Ω^p partitions $\Omega^{p'}$ and forms a $2^{p-p'}$ way binary

matrix space partition, and $\Omega^0/\Omega^{p'}/\Omega^p$ forms a $2^{p'}/2^{p-p'}$–way binary matrix space partition chain.

Example 3.2:

Let $\mathbf{C}_0 = \mathbf{V}_2$, $\mathbf{C}_1$, and $\mathbf{C}_2$ be the (2,2), (2,1), and (2,0) binary block codes defined in Example 3.1. Table 1 illustrates a partition chain for $I = 3$. Thus Ω^0 is a 2×3 binary matrix space, and $\Omega^1, \Omega^2, \ldots, \Omega^6$ are all principle subsets of Ω^0. Moreover, $\Omega^0 \supset \Omega^1 \supset \Omega^2 \supset \Omega^3 \supset \Omega^4 \supset \Omega^5 \supset \Omega^6$. Therefore, $\Omega^0/\Omega^1/\Omega^2/\Omega^3/\Omega^4/\Omega^5/\Omega^6$ forms a 2/2/2/2/2/2–way binary matrix space partition chain. The first three levels of this partition chain are shown schematically in Figure 3. When $\mathbf{C}_{m_i}(z)$, $z > 0$, is the same as $\mathbf{C}_{m_i}$, then $\mathbf{C}_{m_i}$ is given. The determination of the coset representatives can be found using a technique similar to that described in (2) at the beginning of this subsection, that is, by the use of principle coset representatives. However, it is not always necessary to use linear or modulo-2 arithmetic, as will be shown in Section 4, where non-linear arithmetic is used. This allows the binary matrix space to have special properties that will be described later. The coset representatives for Figure 3 can be determined easily from the partition chains. For example, the coset representatives of Ω^3 are given by

$$\mathbf{t}_{m_0}(z) = z^1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} \oplus z^0 \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

$$\mathbf{t}_{m_1}(z) = (z^2 \oplus z^0 \cdot z^1) \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

$$\mathbf{t}_{m_2}(z) = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

where $m_0 = 2$, $m_1 = 1$, $m_2 = 0$ and $z^0 \cdot z^1$ indicates the logical AND of z^0 and z^1. Note the differences between the above equations and (2). For $\mathbf{t}_{m_1}(z)$ it can be seen that z^2 selects $\underline{r}_1$ (along with a non-linear term), since we are now partitioning $\mathbf{C}_0$ for $i = 1$. It should be noted that there are many other partition chains of the 2×3 binary matrix space.

3.2 Partitioning a 2L-D MPSK signal space

The 2–D MPSK signal set, denoted by $S_2(M)$, is the set of complex M^{th} roots of unity, that is, $S_2(M) = \{e^{(\pi/M)\theta}, e^{(3\pi/M)\theta}, \ldots, e^{((2M-1)\pi/M)\theta}\}$, where

$\theta = \sqrt{-1}$ and $M = 2^I$ for any positive integer I. $S_2(M)$ is a group under complex multiplication. For simplicity, we write $S_2(M) = \{y : y = 0, 1, \ldots, M-1\}$, where y is the integer representation of the binary number $\mathbf{y} = [y^{I-1}, \ldots, y^1, y^0]$. This binary or natural mapping of the signal points in $S_2(M)$ is assumed throughout the paper. The $2L$–D, $L \geq 2$, MPSK signal set is defined as the Cartesian product of L 2–D MPSK signal sets, that is,

$$S_{2L}(M) = S_2(M) \times S_2(M) \times \cdots \times S_2(M) \quad (L \text{ times}). \tag{5}$$

Therefore, the $2L$–D MPSK signal set is generated simply by repeating an MPSK signal set L times.

Letting $\mathbf{y}_l$, $l = 0, 1, \ldots, L-1$ be a sequence of L signal points in $S_{2L}(M)$, we now form an $L \times I$ matrix

$$\mathbf{Y} = \begin{bmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_{L-1} \end{bmatrix} = \begin{bmatrix} y_0^{I-1} & \cdots & y_0^1 & y_0^0 \\ y_1^{I-1} & \cdots & y_1^1 & y_1^0 \\ \vdots & \ddots & \vdots & \vdots \\ y_{L-1}^{I-1} & \cdots & y_{L-1}^1 & y_{L-1}^0 \end{bmatrix}, \tag{6}$$

where the rwo vectors y_l represent points in 2–D space and $\mathbf{Y}$ represents a point in $2L$–D space.

Using the notation introduced in the last subsection, the $L \times I$ matrix subspace Ω^p consists of the $L \times I$ matrices $\underline{\omega}$ defined in (3). Using (6) and (3), a $2L$–D MPSK signal subset, denoted by $\mathbf{P}^p = \mathbf{P}(\mathbf{C}_{m_{I-1}}, \ldots, \mathbf{C}_{m_0})$ is obtained from Ω^p by the following mapping:

$$\mathbf{Y} = \underline{\omega}, \tag{7}$$

i.e., $y_l^i = v_l^i$, $i = 0, 1, \ldots, I-1$ and $l = 0, 1, \ldots, L-1$. Since the matrix $\underline{\omega}$ contains L rows, it is mapped into a $2L$–D signal point in $S_{2L}(M)$, with the first row corresponding to the first two dimensions and the last row corresponding to the last two dimensions of the signal point, or equivalently, $\underline{\omega}$ is the binary representation of a signal point in $S_{2L}(M)$. Moreover, since Ω^p contains 2^{IL-p} $L \times I$ matrices, the signal subset $\mathbf{P}^p$ contains 2^{IL-p} $2L$–D signal points. Therefore , Ω^p

and $\mathbf{P}^p$ are isomorphic. The minimum squared Euclidean subset distance (MSSD or Δ_p^2) of $\mathbf{P}^p$ is given by [13]

$$\Delta_p^2 \geq \min(\delta_{I-1}^2 d_{m_{I-1}}, \ldots, \delta_1^2 d_{m_1}, \delta_0^2 d_{m_0}), \tag{8a}$$

where

$$\delta_i^2 = 2 - 2\cos\left(\frac{2^{i+1}\pi}{M}\right), \; i = 0, 1, \ldots, I-1, \tag{8b}$$

is the MSSD of $S_2(M/2^i)$ (recall that $M = 2^I$) and d_{m_i} is the minimum Hamming distance of $\mathbf{C}_{m_i}$, $i = 0, 1, \ldots, I-1$. Due to the symmetry of MPSK signal sets, the inequality in (8a) becomes an equality, and thus for 8PSK and 16PSK (I = 3 and 4, respectively), (8a) and (8b) lead to

$$\Delta_p^2 = \min(4d_{m_2}, 2d_{m_1}, 0.5858 d_{m_0}), \tag{9a}$$

and

$$\Delta_p^2 = \min(4d_{m_3}, 2d_{m_2}, 0.5858 d_{m_1}, 0.1522 d_{m_0}), \tag{9b}$$

respectively. The mapping in (7) is used in [13–16] to construct block codes in signal space.

The signal set corresponding to Ω^p is just the $2L$–D MPSK signal set $S_{2L}(M)$. It is easy to see that $\mathbf{P}^p$ is a subset of $S_{2L}(M)$, provided that Ω^p is a subspace of Ω^0. $\mathbf{P}^p$ is called the principle subset of $S_{2L}(M)$ and is a group under complex multiplication. Hence, $S_{2L}(M)/\mathbf{P}^p$ is a 2^p–way partition which divides $S_{2L}(M)$ into $\mathbf{P}^p$ and its $2^p - 1$ cosets $\mathbf{P}^p(z) = \mathbf{P}(\mathbf{C}_{m_{I-1}}(z), \ldots, \mathbf{C}_{m_0}(z))$, $1 \leq z \leq 2^p - 1$. The signal cosets can be obtained from the corresponding matrix cosets through the mapping in (7).

Example 3.3:

Using the mapping in (7), a 4–D 8PSK signal set partition chain, based on the 2×3 binary matrix space partition chain of Example 3.2, is shown in Table 2.

The first three levels of the partition chain are shown schematically in Figure 4. The MSSD's are obtained from (9a). The partition chain in Figure 4 has special properties in relation to phase rotations which can be found from the principle coset representatives. The derivation of these properties will be explained in Section 4.

Example 3.4:

This example illustrates how to partition the 6–D 8PSK signal set. In the 3–D binary vector space $\mathbf{C}_0 = \mathbf{V}_3 = \{0,1\}^3$, there exists a (3,2) code and a (3,1) code with Hamming distances 2 and 3, respectively. However, the (3,1) code is not a subcode of the (3,2) code. Consequently, three different 2/2/2–way binary matrix space partition chains are possible: $\mathbf{C}_0/\mathbf{C}_1^1/\mathbf{C}_2^1/\mathbf{C}_3$, $\mathbf{C}_0/\mathbf{C}_1^2/\mathbf{C}_2^2/\mathbf{C}_3$, and $\mathbf{C}_0/\mathbf{C}_1^2/\mathbf{C}_2^1/\mathbf{C}_3$, where $\mathbf{C}_3$ is the (3,0) code and

$$\mathbf{C}_1^1:\ (3,2)\ \text{code},\ \mathbf{G}_1^1 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix},\ d_1^1 = 2,$$

$$\mathbf{C}_2^1:\ (3,1)\ \text{code},\ \mathbf{G}_2^1 = [\,0 \quad 1 \quad 1\,],\ d_2^1 = 2,$$

$$\mathbf{C}_1^2:\ (3,2)\ \text{code},\ \mathbf{G}_1^2 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix},\ d_1^2 = 1,$$

$$\mathbf{C}_2^2:\ (3,1)\ \text{code},\ \mathbf{G}_2^2 = [\,1 \quad 1 \quad 1\,],\ d_2^2 = 3.$$

Note that $\mathbf{C}_3 \subset \mathbf{C}_2^1 \subset \mathbf{C}_1^1 \subset \mathbf{C}_0$, $\mathbf{C}_3 \subset \mathbf{C}_2^2 \subset \mathbf{C}_1^2 \subset \mathbf{C}_0$, and $\mathbf{C}_3 \subset \mathbf{C}_2^1 \subset \mathbf{C}_1^2 \subset \mathbf{C}_0$. A variety of 6–D 8PSK signal set partition chains can be constructed based on these three 3×3 binary matrix space partition chains. Two 6–D 8PSK signal set partition chains obtained by the mapping in (7) are given in Tables 3(a) and 3(b).

An 8–D 8PSK signal set partition chain is given in Table 4. Before leaving this section, we give one more example to show how to partition multi-D 16PSK signal sets.

Example 3.5:

In this example, we partition the 4–D 16PSK signal set. Let $\mathbf{C}_0$, $\mathbf{C}_1$, and $\mathbf{C}_2$ be the (2,2), (2,1) and (2,0) binary block codes defined in Example 3.1. For $I = \log_2 16 = 4$, Table 5 illustrates the partition chain that is used. Then Ω^0 is a 2×4 binary matrix space, and $\Omega^1, \Omega^2, \ldots, \Omega^8$ are all principle subsets

of Ω^0. Moreover, $\Omega^0 \supset \Omega^1 \supset \cdots \supset \Omega^8$. Therefore $\Omega^0/\Omega^1/\cdots/\Omega^8$ forms a 2/2/2/2/2/2/2/2–way binary matrix space partition chain. The first three levels of the partition chain are shown in Figure 5. The MSSD's are found from (9b).

Three 6–D 16PSK signal set partition chains are listed in Tables 6(a)-6(c), respectively. The corresponding binary matrix space partition chains can be read from these tables.

From the above discussion, we observe that various partitions can be constructed for a given multi-D MPSK signal set, and this establishes the basis for constructing good codes. It should be pointed out that Forney's [9] squaring construction and 3–construction can also be applied to partitioning multi-D MPSK signal sets. The resulting partitions, however, may be inferior to the partitions introduced above. For example, in partitioning the 8–D 8PSK signal set using the squaring construction (or 4–construction), Δ_4^2 =2 instead of 2.343 as shown in Table 4.

4 Multi-D TC-MPSK Design

This section describes how convolutional codes are constructed for the $2L-$D MPSK signal sets described previously. We first describe how to construct signal sets which have good phase rotation properties. Following this, the method used to find good convolutional codes based on parity check equations is presented.

4.1 Construction of signal sets

In the previous section, a signal set was described in terms of the principle subset Ω^p and its cosets $\Omega^p(z)$, $0 \leq z \leq 2^p - 1$. In Section 3.1, it was shown that for $I = 1$, cosets can be constructed by using the principle coset representatives $\underline{\tau}_m$. For $I > 1$ we can use a similar technique, where the principle coset representative $\underline{\tau}^p$ at partition level $p = \sum_i m_i$ is given by

$$\underline{\tau}^p = \underline{\omega}^p(2^{p-1}).$$

If m_i retains the same value going from partition level $p-1$ to p, then $\mathbf{t}_{m_i}(2^{p-1})$ equals the all zero vector $\mathbf{0} = [0...0]^{\mathrm{T}}$. This can be seen in Figures 3 to 5, where at partition level p and $z = 2^{p-1}$, only those m_i's that increase from $p-1$ to p have any effect on the coset. There is no principle coset representative for $p = 0$, since the binary matrix space Ω^0 has no cosets. Also note that $\underline{\tau}^p \in \Omega^{p-1}$ and $\underline{\tau}^p \notin \Omega^p$.

Example 4.1

For the partition chain in Table 2, the principle coset representatives $\underline{\tau}^p$ for the 4–D 8PSK signal space are

$$\begin{aligned}
\underline{\tau}^1 &= [\mathbf{t}_0(1), \mathbf{t}_0(1), \mathbf{t}_1(1)] = [\mathbf{0}, \mathbf{0}, \underline{\tau}_1], \\
\underline{\tau}^2 &= [\mathbf{t}_0(2), \mathbf{t}_0(2), \mathbf{t}_2(2)] = [\mathbf{0}, \mathbf{0}, \underline{\tau}_2], \\
\underline{\tau}^3 &= [\mathbf{t}_0(4), \mathbf{t}_1(4), \mathbf{t}_2(4)] = [\mathbf{0}, \underline{\tau}_1, \mathbf{0}], \\
\underline{\tau}^4 &= [\mathbf{t}_0(8), \mathbf{t}_2(8), \mathbf{t}_2(8)] = [\mathbf{0}, \underline{\tau}_2, \mathbf{0}], \\
\underline{\tau}^5 &= [\mathbf{t}_1(16), \mathbf{t}_2(16), \mathbf{t}_2(16)] = [\underline{\tau}_1, \mathbf{0}, \mathbf{0}], \\
\underline{\tau}^6 &= [\mathbf{t}_2(32), \mathbf{t}_2(32), \mathbf{t}_2(32)] = [\underline{\tau}_2, \mathbf{0}, \mathbf{0}],
\end{aligned}$$

where $\underline{\tau}_1 = [0\ 1]^{\mathrm{T}}$, $\underline{\tau}_2 = [1\ 1]^{\mathrm{T}}$, and $\mathbf{0}$ is the all zero vector $[0\ 0]^{\mathrm{T}}$.

As in (2), we can find the coset representatives at any partition level by the modulo-2 addition of the respective principle coset representatives, i.e.,

$$\underline{\omega}^p(z) = \bigoplus_{j=0}^{p-1} z^j \underline{\tau}^{j+1};\ 0 \le z \le 2^p - 1. \tag{10}$$

An alternative and more useful way of forming the cosets is as follows. An $L \times 1$ $\tilde{M}-$ary vector space ω can be formed such that

$$\omega = \sum_{i=0}^{I-1} 2^i \mathbf{v}^i, \tag{11}$$

where modulo-$\tilde{M}$ arithmetic is used and $\tilde{I}$ is the number of non-zero values of m_i at partition level p. Then the principle coset representatives can be expressed in integer form as

$$\tau^p = \sum_{i=0}^{I-1} 2^i \mathbf{t}_{m_i}(2^{p-1}) = \begin{bmatrix} \tau_0^p \\ \tau_1^p \\ \vdots \\ \tau_{L-1}^p \end{bmatrix}, \tag{12}$$

where modulo-$\tilde{M} = 2^{\tilde{I}}$ arithmetic is used. Note that τ^p is an $L \times 1$ vector and that its elements belong to the set $0,1,\ldots,\tilde{M}-1$. The coset representatives at partition level p are then

$$\omega^p(z) = \sum_{j=0}^{p-1} z^j \tau^{j+1};\ 0 \le z \le 2^p - 1, \tag{13}$$

where modulo-$\tilde{M}$ arithmetic is used.

Example 4.2:
For the principle coset representatives found in Example 4.1, the principle coset representatives in integer from for 4–D 8PSK are found from (12) as

$$\tau^6 = \begin{bmatrix} 4 \\ 4 \end{bmatrix},\ \tau^5 = \begin{bmatrix} 0 \\ 4 \end{bmatrix},\ \tau^4 = \begin{bmatrix} 2 \\ 2 \end{bmatrix},\ \tau^3 = \begin{bmatrix} 0 \\ 2 \end{bmatrix},\ \tau^2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix},\ \tau^1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

To find the coset representative (in integer form) at partiton level $p = 3$ and for $z = 3$, we see from Table 2 that $\tilde{I} = 2$, and hence from (13)

$$\omega^3(3) = z^2\tau^3 + z^1\tau^2 + z^0\tau^1 = 0\begin{bmatrix}0\\2\end{bmatrix} + 1\begin{bmatrix}1\\1\end{bmatrix} + 1\begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}1\\2\end{bmatrix},$$

where modulo-4 arithmetic is used.

In a practical implementation of an encoder, a single point in multi-D space, given a value of z, can be found by partitioning down to level $p = IL$. At this partition level the coset representatives themselves are the actual points in signal space. We call this *full partitioning*. Let $y(z)$ represent each $2L$–D MPSK point $\mathbf{Y}$ in integer form, i.e,

$$y(z) = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{L-1} \end{bmatrix}, \text{ where } y_l = \sum_{i=0}^{I-1} 2^i y_l^i, \; l = 0, 1, ..., L-1. \qquad (14)$$

The variable z is used in $y(z)$ since each point in $\mathbf{Y}$ can now be described in terms of z. Thus, with full partitioning, we obtain (for $p = IL$)

$$y(z) = \omega^{IL}(z) = \sum_{j=0}^{IL-1} z^j \tau^{j+1}; \; 0 \le z \le 2^{IL} - 1, \qquad (15)$$

where addition is modulo-M.

Equation (15) can now be used to describe a signal point in $2L$–D space with MPSK modulation. The number of bits z^j used to describe a signal point is IL. If the least significant bit (lsb) is used for coding, we can form a rate $(IL-1)/IL$ code. Other rates can also be formed by letting the q lsb's of the mapping be set to 0. We do this to insure that the MSSD's are as large as possible, and thus the best codes can be found. Therefore we let

$$y^q(z) = \sum_{j=q}^{IL-1} z^{j-q} \tau^{j+1}; \; 0 \le z \le 2^{IL-q} - 1, \; 0 \le q \le L-1, \qquad (16)$$

where $y^q(z)$ represents a point z in $2L$–D MPSK signal space such that the first q bits of (15) are 0 and addition is modulo-M. Now $\mathbf{z} = [z^{IL-q-1}, \ldots, z^1, z^0]$, where the lsb of z is always the coding bit. This insures that the parity check equations can always be expressed in terms of z without depending on the type and partition level of the signal set used. From (16), codes of rates $(IL-q-1)/(IL-q)$ can be formed. An upper limit of $q = L-1$ is set because for $q \geq L$ the signal set is partitioned such that $d_{m_0} = \infty$, i.e., an $M/2^j$ – PSK, $j \geq 1$, signal set is being used (one exception is the 8–D 8PSK signal set (Table 4) where $d_{m_0} = \infty$ for $q \geq L+1$). The MSSD's range from Δ_q^2 to Δ_{IL}^2 and the uncoded minimum squared Euclidean distance (SED) is Δ_{q+1}^2.

Example 4.3:
We can form a rate 4/5 code with 4–D 8PSK modulation ($q = 1,\ L = 2,\ I = 3$). Then

$$y^1(z) = z^4 \begin{bmatrix} 4 \\ 4 \end{bmatrix} + z^3 \begin{bmatrix} 0 \\ 4 \end{bmatrix} + z^2 \begin{bmatrix} 2 \\ 2 \end{bmatrix} + z^1 \begin{bmatrix} 0 \\ 2 \end{bmatrix} + z^0 \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

where addition is modulo-8.The uncoded minimum SED is $\Delta_2^2 = 2.0$, which is the same as uncoded QPSK.

4.2 Effect of a 360°/M phase rotation on a Multi-D MPSK signal set

The reason for constructing the signal set as in (16) is that there are at most I bits in z affected by a signal set rotation of $\Psi = 360°/M$. For 8PSK and 16PSK, this corresponds to rotations of 45° and 22.5°, respectively. Initially, we consider all possible mapped bits, and thus $q = 0$.

Consider that a 2–D MPSK signal set has been rotated by Ψ. Since we are using natural mapping, the integer representation of the rotated signal point is $y_r = y + 1$, where y is the integer representation of the signal point before rotation and modulo-M addition is used. If binary notation is used, then

$$y_r^0 = y^0 \oplus 1, \tag{17a}$$
$$y_r^1 = y^1 \oplus y^0, \tag{17b}$$
$$y_r^2 = y^2 \oplus y^0 \cdot y^1, \tag{17c}$$
$$\vdots$$

If there are $I = \log_2 M$ bits in a signal set, then all I bits are affected by a phase rotation of Ψ.

We now consider the first partition of a multi-D MPSK signal set. z^0 is used to select one of two partitions, $\mathbf{P}(\mathbf{C}_0, \ldots, \mathbf{C}_0, \mathbf{C}_1)$ or $\mathbf{P}(\mathbf{C}_0, \ldots, \mathbf{C}_0, \mathbf{C}_1(1))$. We know from (17a) that the lsb's are inverted by a Ψ phase rotation. Then, if all the code words in $\mathbf{C}_1$ remain code words in $\mathbf{C}_1$ when inverted, then z^0 will remain the same after a phase rotation. That is, if $\overline{\mathbf{C}_1} = \mathbf{C}_1$, then $z_r^0 = z^0$. However, if $\overline{\mathbf{C}_1} = \mathbf{C}_1(1)$, then $z_r^0 = z^0 \oplus 1$, as can be seen from the set partition. A simple way to tell if a block code has the property that $\overline{\mathbf{C}_{m_i}} = \mathbf{C}_{m_i}$, or if $\overline{\mathbf{C}_{m_i}}$ equals one of its cosets, is to examine its coset representative at that partition level. Assume $\underline{\tau}_{m_i} = [1 \ldots 1]^T = \mathbf{1}$. Since $\underline{\tau}_{m_i} \in \mathbf{C}_{m_i - 1}$, then $\overline{\mathbf{C}_{m_i - 1}} = \mathbf{C}_{m_i - 1}$ follows from code linearity (the inverse of $\underline{\tau}_{m_i} = \mathbf{1}$ is the all zero vector $\mathbf{0}$). However, we also have $\underline{\tau}_{m_i} \notin \mathbf{C}_{m_i}$, and thus the inverse of $\mathbf{0}$ and all the other vectors in $\mathbf{C}_{m_i}$ form a coset of $\mathbf{C}_{m_i}$(again for linear codes). Thus, if $\underline{\tau}_{m_i} \neq \mathbf{1}$ at partition level p then $z_r^{p-1} = z^{p-1}$; otherwise, $z_r^{p-1} \neq z^{p-1}$.

For $\mathbf{C}_{m_0}$, we can always say that if $\underline{\tau}_{m_0} = \mathbf{1}$ at partition level p, then $z_r^{p-1} = z^{p-1} \oplus 1$; otherwise, $z_r^{p-1} = z^{p-1}$, since the additions for y_l^0, $l = 0, 1, \ldots, L-1$, are modulo-2 using either (10) or (16) to map a signal point. However, for $i \geq 1$, (10) gives signal sets which have $IL - I - 1$ bits affected by a phase rotation. This is because an inverted z^p which affects $\mathbf{C}_{m_0}$ will cause some signal points to rotate in different directions. However, using the mapping in (16), all the signal points will rotate in the same direction, since modulo-M arithmetic is used. Thus, using the mapping in (16),

$$\begin{aligned} z_r^{p_0-1} &= z^{p_0-1} \oplus 1, \\ z_r^{p_1-1} &= z^{p_1-1} \oplus z^{p_0-1}, \\ z_r^{p_2-1} &= z^{p_2-1} \oplus z^{p_0-1} \cdot z^{p_1-1}, \\ &\vdots \end{aligned}$$

where the p_i's, $0 \leq i \leq I-1$, correspond to the partition levels where $\underline{\tau}_{m_i} = \mathbf{1}$, and for all other partition levels, $z_r^{p-1} = z^{p-1}$. That is, the p_i's indicate which bits are affected by a phase rotation of Ψ.

Example 4.4:
Consider the 4–D 8PSK signal set with a rate 5/6 encoder. By examining Table 2,

we see that $p_0 = 2$, $p_1 = 4$, and $p_2 = 6$ correspond to the partition levels where $\underline{\tau}_{m_i} = \mathbf{1} = [1\ 1]^T$. Thus the effect of a 45° phase rotation on the signal set is

$$\begin{aligned} z_r^0 &= z^0 \\ z_r^1 &= z^1 \oplus 1 \\ z_r^2 &= z^2 \\ z_r^3 &= z^3 \oplus z^1 \\ z_r^4 &= z^4 \\ z_r^5 &= z^5 \oplus z^1 \cdot z^3. \end{aligned} \tag{18}$$

The phase invariance of the mapping used for the 4–D 8PSK signal set can be checked as follows. From (14) and (15) the signal outputs can be described in terms of z as

$$\begin{aligned} \begin{bmatrix} y_0 \\ y_1 \end{bmatrix} &= \begin{bmatrix} 4y_0^2 + 2y_0^1 + y_0^0 \\ 4y_1^2 + 2y_1^1 + y_1^0 \end{bmatrix} \\ &= z^5 \begin{bmatrix} 4 \\ 4 \end{bmatrix} + z^4 \begin{bmatrix} 0 \\ 4 \end{bmatrix} + z^3 \begin{bmatrix} 2 \\ 2 \end{bmatrix} + z^2 \begin{bmatrix} 0 \\ 2 \end{bmatrix} + z^1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} + z^0 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ &= (4z^5 + 2z^3 + z^1) \begin{bmatrix} 1 \\ 1 \end{bmatrix} + (4z^4 + 2z^2 + z^0) \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \end{aligned}$$

where all additions are modulo-8. After a 45° phase rotation, we have $y_{l,r} = y_l + 1$, for $l = 0, 1$. Thus from above we can form the following phase rotation equations,

$$\begin{bmatrix} y_{0,r} \\ y_{1,r} \end{bmatrix} = (4z^5 + 2z^3 + z^1 + 1) \begin{bmatrix} 1 \\ 1 \end{bmatrix} + (4z^4 + 2z^2 + z^0) \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Note that a 1 is added to the term whose coset is $[1\ 1]^T$. Hence this term "absorbs" the affect of the phase rotation, leaving the remaining term unaffected. Thus from (17), we can form the phase rotation equations given in (18). Had the signal set been constructed using (10), only z^0 would have remained unchanged by a 45° phase rotation.

We have shown that for $q = 0$, the bits that are affected by a phase rotation of Ψ are z^{p_j-1}, $0 \leq j \leq I-1$. For $q > 0$ the bits that are affected are z^{p_j-q-1}, $0 \leq j \leq I-1$. However, depending on the signal set, $p_j - q - 1$ for some j may be less than zero. If this is true, the minimum phase transparency will be $2^{\tilde{d}}\Psi$, where $\tilde{d}$ is the number of terms $p_j - q - 1$ that are less than zero, and the number of bits ($\tilde{s}$) that are affected by a $2^{\tilde{d}}\Psi$ phase rotation is $\tilde{s} = I - \tilde{d}$. For example, the 6–D 8PSK signal set in Table 3(a) has $p_0 = 1$, $p_1 = 4$, and $p_2 = 7$. Thus if $q = 1$, then $p_0 - q - 1 = -1$, which is less than zero, implying that $\tilde{d} = 1$, and thus there will be only $\tilde{s} = I - \tilde{d} = 2$ bits affected by a $2\Psi = 90°$ phase rotation. Note that a phase rotation of $\Psi = 45°$ of this signal set will produce its coset.

Fortunately, for the codes and signal sets considered in this paper, the above complication does not occur. This is partly due to the fact that for many signal sets with $q = 0$, the $L-1$ lsb's are not affected by a phase rotation of Ψ. Since we consider only signal sets with $0 \leq q \leq L-1$ in this paper, $\tilde{d} = 0$. For those signal sets where this is not true (e.g., in some 6–D signal sets), it has been found that the convolutional codes produced are inferior (in either minimum FSED or number of nearest neighbors) to an alternative signal set with $\tilde{d} = 0$. Therefore, we will not consider the above effect further.

When a signal set is combined with a convolutional encoder, we must consider the effect of rotating coded sequences. A similar result is obtained as above in that, depending on the code and the signal set, the signal set can be rotated in multiples of $2^d\Psi$ and still produce valid code sequences. We define d to be the degree of transparency. The actual determination of d is described in section 4.4. Also, the number of bits (s) that are affected by a phase rotation is $s = I - d$.

For $0 \leq q \leq L-1$, the actual bits that are affected by a phase rotation of Ψ are z^{b_j}, where $b_j = p_j - q - 1$, $0 \leq j \leq I-1$. More generally, the bits that are affected by a phase rotation of $2^d\Psi$ are z^{c_j}, where $c_j = p_{j+d} - q - 1$, $0 \leq j \leq s-1$. These two seperate notations (b_j and c_j) are used because the determination of d depends on b_j.

4.3 The general encoder system

From the above information we can now construct a suitable encoder which is illustrated in Figure 6. The general multi-D encoder system consists of five sections. These sections are the differential precoder, the binary convolutional encoder, the multi-D signal mapper, the parallel to serial converter, and the 2–D signal mapper. In this paper the convolutional encoder is assumed to be systematic with feedback as in [1]. That is, $z^j(D) = x^j(D),\ 1 \leq j \leq k$, where D is the delay operator and polynomial notation is used. The parity sequence, $z^0(D)$ will be some function of itself and the $x^j(D),\ 1 \leq j \leq k$. The parity check equation of an encoder describes the relationship in time of the encoded bit streams. It is a very useful and efficient means of describing a convolutional code, since it is independent of the input/output encoder relationships. For an $R = k/(k+1)$ code, the parity check equation is

$$H^{\tilde{k}}(D)z^{\tilde{k}}(D) \oplus \cdots \oplus H^1(D)z^1(D) \oplus H^0(D)z^0(D) = 0(D),\ 1 \leq \tilde{k} \leq k, \quad (19)$$

where $\tilde{k}$ is the number of input sequences that are checked by the encoder, $H^j(D),\ 0 \leq j \leq \tilde{k}$, is the parity check polynomial of $z^j(D)$, $0(D)$ is the all zero sequence.

Since the encoder is systematic, the differential precoder only preconditions those bits which are affected by a phase rotation, i.e., the input bits into the encoder which need to be preconditioned are $w^{c_0}, w^{c_1}, \ldots, w^{c_{s-1}}$. If $c_0 = 0$, we replace w^0 (which does not exist) by z^0, as shown in Figure 6 by the dashed line. For example, an encoder for a rate 8/9 code which uses the 6–D (partition I) 8PSK signal set given in Table 6(a) may (depending on the phase transparency) need this modification. This is because this signal set has $b_0 = 0$, and thus if the code has $d = 0$, then z^0 will need to be precoded. Figure 7 illustrates the two types of precoders. Note that the storage elements have a delay of LT, where T is the symbol period in time of each 2–D signal point that is transmitted by the 2–D signal mapper. Figure 7(a) illustrates the precoder with $c_0 > 0$, where there are s inputs that need to be precoded. The basic component of the precoder is the modulo-2^s binary adder. For most codes this is the precoder to be used. Figure 7(b) gives the other case where $c_0 = 0$ and $s - 1$ input bits are precoded (the other precoded bit being z^0). For the bits that are not precoded, $x^i = w^i,\ i \neq c_j$.

At this point, we summarize the notation and indicate the limits on the parameters used in the search for good codes. For a rate $(IL-q-1)/(IL-q)$ code,

I = no. of bits in each 2–D signal ($3 \leq I \leq 4$),
$M = 2^I$ = no. of signal points in each 2–D signal set,
L = no. of 2–D signal sets ($2 \leq L \leq 4$ for 8PSK and $2 \leq L \leq 3$ for 16PSK)
p = partition level of signal set ($0 \leq p \leq IL$),
q = the partition level p where mapping begins ($0 \leq q \leq L-1$),
z = signal set mapping parameter ($0 \leq z \leq 2^{p-q}-1$),
$k = IL-q-1$ = no. of input bits to encoder,
$\Psi = 360°/M$ = minimum phase transparency with $q=0$,
d = degree of phase transparency ($2^d\Psi$, $0 \leq d \leq I$),
$s = I-d$ = no. of bits in z affected by a $2^d\Psi$ phase rotation ($0 \leq s \leq I$),
$c_j = p_{j+d}-q-1$ = the bits z^{c_j} affected by a $2^d\Psi$ phase rotation ($0 \leq j \leq s-1$).

There are two types of systematic convolutional encoders that can be constructed. Before proceeding with the description of these encoders, we return to the parity check equation given in (19). As in [1], we define v to be the maximum degree of all the parity check polynomials $H^j(D), 0 \leq j \leq \tilde{k}$. For $\tilde{k} < j \leq k$, $H^j(D) = 0$, since the bits corresponding to these polynomials are not checked by the encoder. If $\tilde{k} < v$, the parity check polynomials are of the form

$$H^j(D) = 0 \oplus h^j_{v-1}D^{v-1} \oplus \cdots \oplus h^j_1 D \oplus 0, \ 1 \leq j \leq \tilde{k}, \qquad (20a)$$

$$H^0(D) = 1 \oplus h^0_{v-1}D^{v-1} \oplus \cdots \oplus h^0_1 D \oplus 1. \qquad (20b)$$

Equations (20) insure that the SED between paths in a trellis leaving or entering a state is at least $2\Delta^2_{q+1}$. Thus codes can be found that have a FSED or d^2_{free} (the minimum SED between all possible coded sequences) of at least $2\Delta^2_{q+1}$, where Δ^2_{q+1} is the d^2_{free} of the uncoded comparison system. A theoretical justification for constructing codes in this manner has been found in [17] where it is shown, using random coding arguments, that these codes have a large FSED on the average. A minimal systematic encoder can be implemented from (20), since $h^0_0 = 1$ [1]. The encoding equations are

$$z^j(D) = x^j(D),\ 1 \le j \le k, \tag{21a}$$

$$z^0(D) = H^{\tilde{k}}(D)x^{\tilde{k}}(D) \oplus \cdots \oplus H^1(D)x^1(D) \oplus (H^0(D) \oplus 1)z^0(D). \tag{21b}$$

An encoder implementation using (20) is shown in Figure 8(a).

For all codes with $v = 1$ and for some codes with $v > 1$, $\tilde{k} = v$. For these codes we cannot set $h_v^j = 0,\ 1 \le j \le \tilde{k}$. This is because $\tilde{k}$ checked bits require at least $\tilde{k}$ terms in $H^j(D),\ 1 \le j \le \tilde{k}$, that are variable. If there are not enough variables, then there will be some non-zero $\mathbf{x}^{\tilde{k}} = [x^{\tilde{k}}, \ldots, x^2, x^1]$ such that $\bigoplus_{j=1}^{\tilde{k}} h_m^j x^j = 0,\ 1 \le m \le v$. That is, there will be more than $2^{k-\tilde{k}}$ parallel transitions between states in the trellis. To avoid this problem, when $\tilde{k} = v$, we let the parity check polynomials be

$$H^j(D) = h_v^j D^v \oplus \cdots \oplus h_1^j D \oplus 0,\ 1 \le j \le \tilde{k}, \tag{22a}$$

$$H^0(D) = h_v^0 D^v \oplus \cdots \oplus h_1^0 D \oplus 1. \tag{22b}$$

In (22), ther is always at least one term $h_v^j,\ 1 \le j \le \tilde{k}$, that is equal to one, if the number of variables $\tilde{k}$ is to be maintained. Thus the degree of the encoder remains at v. The d_{free}^2 is at least $\Delta_q^2 + \Delta_{q+1}^2$, since the minimum incremental SED between paths leaving a state is Δ_{q+1}^2 (since $h_0^j = 0,\ 1 \le j \le \tilde{k}$, and $h_0^0 = 1$) and between paths entering a state is Δ_q^2 (since $h_v^j \in \{0,1\}$ for $0 \le j \le \tilde{k}$). The encoding equations are given by (21) and an encoder implementation for $\tilde{k} = v$ is shown in Figure 8(b).

The multi-D signal mapper can be implemented by using cosets of the signal set, the value of q, and (16). Figure 9 illustrates an implementation of the multi-D signal mapper. Note that only modulo-M adders are required to implement the signal mapper. The thick lines in Figure 9 represent the I bits for each MPSK signal point. Due to the set partitioning, many of the coefficients are equal to zero and the non-zero coefficients have only one non-zero bit. Thus, only one line is needed to represent each coefficient.

The second to last section of the encoder is the parallel to serial converter, which takes the L groups of I bits and forms a stream with I bits in each group. That is, we are assuming a channel which is limited to transmitting one 2–D signal point at a time. A representation of a parallel to serial converter is shown in Figure 10. Finally the 2–D signal mapper takes the I bits for each 2–D signal point and produces the required real and imaginary (or amplitude and phase) components for a modulator.

Example 4.5

In this example, we describe how to implement a particular code. The code is used with a 6–D 8PSK signal set. Thus $L = 3$ and $I = 3$. We also have $q = 1$, so that a rate 7/8 code is formed. The partition that is used is given in Table 3(b), from which we obtain $p_0 = 3$, $p_1 = 4$, and $p_2 = 7$. The code is 90° transparent and thus $d = 1$ and $s = 2$. Therefore $c_0 = p_1 - q - 1 = 2$, and $c_1 = p_2 - q - 1 = 5$. Thus bits w^2 and w^5 are precoded using a modulo-4 adder. Since $c_0 > 0$, the precoder given in Figure 7(a) is used. For this code, $\tilde{k} = 2$, and the parity check polynomials are $H^0(D) = D^4 \oplus D^3 \oplus D \oplus 1$, $H^1(D) = D$, and $H^2(D) = D^3 \oplus D^2$. Excluding the parallel to serial converter and the 2–D signal mapper, the encoder is shown in Figure 11. This code has 16 states ($v = 4$). Note that the multi-D signal mapper does not exactly correspond to Figure 9. This is due to the fact that the terms have been collected so as to minimize the number of modulo-8 adders that are required. Also note that bits other than z^1 which are tapped $L = 3$ times are checked by the precoder, since the code is 90° transparent.

4.4 Convolutional Encoder Effects on Transparency

As mentioned previously the convolutional encoder can affect the total transparency of the system. The method used to determine transparency is to examine the parity check equation and the bits that are affected by a phase rotation. A code is transparent if its parity check equation, after substituting $z^j(D)$ with $z_r^j(D)$, $0 \leq j \leq \tilde{k}$ (the rotated sequences), remains the same. There will normally be at most I bits that are affected by a phase rotation, $z^{b_0}, \ldots, z^{b_{I-1}}$, $b_j = p_j - q - 1$, $0 \leq j \leq I - 1$. That is,

$$z_r^{b_0} = z^{b_0} \oplus 1, \tag{23a}$$

$$z_r^{b_1} = z^{b_1} \oplus z^{b_0}, \tag{23b}$$
$$z_r^{b_2} = z^{b_2} \oplus z^{b_0}.z^{b_1}. \tag{23c}$$
$$\vdots$$

Assume that $0 \leq b_0 \leq \tilde{k}$ and $b_j > \tilde{k}$; $1 \leq j \leq I-1$. Then only one term in the parity check equation is affected by a phase rotation. The other bits have no effect since they are not checked by the encoder. The parity check equation after a phase rotation of Ψ becomes

$$\begin{aligned} H^{\tilde{k}}(D)z^{\tilde{k}}(D) \oplus \cdots \oplus H^{b_0}(D)(z^{b_0}(D) \oplus 1(D)) \\ \oplus \cdots \oplus H^0(D)z^0(D) = 0(D), \\ H^{\tilde{k}}(D)z^{\tilde{k}}(D) \oplus \cdots \oplus H^{b_0}(D)z^{b_0}(D) \\ \oplus \cdots \oplus H^0(D)z^0(D) = E[H^{b_0}(D)](D), \end{aligned} \tag{24}$$

where $E[H^{b_0}(D)]$ is the modulo-2 number of non-zero terms in $H^{b_0}(D)$ and $1(D)$ is the all ones sequence. Thus if there is an even number of terms in $H^{b_0}(D)$, (24) will be the same as (19). That is, the code is transparent to integer multiples of Ψ phase rotations of the signal set. However, if there is an odd number of terms in $H^{b_0}(D)$, then $E[H^{b_0}(D)] = 1$ and the coset of the convolutional code is produced. Even though the two equations are closely related, the codes are quite different and a decoder will not be able to produce correctly decoded data from a Ψ phase rotation of the signal set.

Now assume that the first two terms are affected by a phase rotation, i.e., $0 \leq b_0, b_1 \leq \tilde{k}$, and $b_j > \tilde{k}$, $2 \leq j \leq I-1$. The terms in the parity check polynomial $H^{b_0}(D)z^{b_0}(D) \oplus H^{b_1}(D)z^{b_1}(D)$ now become

$$(H^{b_0}(D) \oplus H^{b_1}(D))z^{b_0}(D) \oplus H^{b_1}(D)z^{b_1}(D) \oplus E[H^{b_0}(D)](D).$$

In this case the parity check equation will be different after a phase rotation. This does not mean that the code is not transparent to any multiple of Ψ phase rotations. In fact, the code could be transparent to 2Ψ or 4Ψ phase rotations. This is because the phase rotation equations reduce to

$$
\begin{aligned}
z_r^{b_0} &= z^{b_0} \\
&\vdots \\
z_r^{b_{d-1}} &= z^{b_{d-1}} \\
z_r^{b_d} &= z^{b_d} \oplus 1 \\
z_r^{b_{d+1}} &= z^{b_{d+1}} \oplus z^{b_d} \\
&\vdots
\end{aligned}
$$

for a $2^d\Psi$ phase rotation, where $d = 1$ or 2. If there is an even number of terms in $H^{b_1}(D)$, then $d = 1$. This is because the even number of non-zero terms in $H^{b_1}(D)$ cancels the effect on $z^{b_1}(D)$ when the signal set is rotated by 2Ψ. That is, the code is transparent to integer multiples of 2Ψ phase rotations and no less. If there is an odd number of non-zero terms, this canceling effect can not occur, and then $d = 2$ giving a phase transparency of 4Ψ.

In general, for $0 \leq b_0, \ldots, b_f \leq \tilde{k},\ 0 \leq f \leq I-1,\ d = f + E[H^{b_f}(D)]$. Then we can determine those bits z^{c_j} which are affected by a $2^d\Psi$ phase rotation, i.e., $c_j = b_{j+d} = p_{j+d} - q - 1,\ 0 \leq j \leq s-1$, where $s = I - d$.

Example 4.6:
For the code given in Example 4.5, $\tilde{k} = 2$, $I = 3$, and $q = 1$. Thus $b_0 = 1$, $b_1 = 2$, $b_2 = 5$, and $0 \leq b_0, b_1 \leq 2$. Therefore $f = 1$ and $d = 1 + E[H^{b_1}(D)] = 1 + E[D^3 \oplus D^2] = 1$. Thus the code is 90° transparent and $c_0 = 2$ and $c_1 = 5$.

4.5 Systematic search for good small constraint length codes

For each multi-D signal set considered there are a number of code rates for which v can range from one to as large as one wishes. As v is increased a comprehensive code search becomes time consuming due to the greater complexity of each code. We have thus limited our search to $v \leq 6$. The criteria used to find the best codes are the FSED (d^2_{free}), the number of nearest neighbors ($N(d^2_{\text{free}})$) and the code transparency (d). The code search algorithm that was implemented is similar to that in [1], but with a number of differences which include the extra criterias mentioned above.

The actual code search involves using a rate $\tilde{k}/(\tilde{k}+1)$ code. Thus two seperate notations are used to distinguish the rate $k/(k+1)$ encoder and the simplified rate $\tilde{k}/(\tilde{k}+1)$ encoder. For the rate $k/(k+1)$ encoder, we have $\mathbf{x}_n = [x_n^k, \ldots, x_n^1]$ (the input to the encoder) and $\mathbf{z}_n = [z_n^k, \ldots, z_n^1, z_n^0]$ (the mapped bits or encoder output) at time n. Also, $\mathbf{e}_n = [e_n^k, \ldots, e_n^1, e_n^0]$ is the modulo-2 difference between two encoder outputs $\mathbf{z}_n$ and $\mathbf{z}'_n$ at time n, i.e., $\mathbf{e}_n = \mathbf{z}_n \oplus \mathbf{z}'_n$. There are 2^{k+1} combinations of $\mathbf{z}_n$ and $\mathbf{z}'_n$ that give the same $\mathbf{e}_n$. For the rate $\tilde{k}/(\tilde{k}+1)$ code, we denote reduced versions of $\mathbf{x}_n$, $\mathbf{z}_n$, and $\mathbf{e}_n$ as $\mathbf{x}_n^{\tilde{k}} = [x_n^{\tilde{k}}, \ldots, x_n^1]$, $\mathbf{z}_n^{\tilde{k}} = [z_n^{\tilde{k}}, \ldots, z_n^1, z_n^0]$, and $\mathbf{e}_n^{\tilde{k}} = [e_n^{\tilde{k}}, \ldots, e_n^1, e_n^0]$, respectively.

In order to find d^2_{free} for a particular code, the Squared Euclidean Weights (SEW) $w^2(\mathbf{e}_n)$ were used. As defined in [1], $w^2(\mathbf{e}_n)$ is the minimum SED between all combinations $a(\mathbf{z}_n)$ and $a(\mathbf{z}'_n)$ such that $\mathbf{e}_n = \mathbf{z}_n \oplus \mathbf{z}'_n$ and $a(\mathbf{z}_n)$ is the actual signal point in $2L-$D space. This can be defined as

$$w^2(\mathbf{e}_n) = \min_{\text{all } \mathbf{z}_n} d^2[a(\mathbf{z}_n), a(\mathbf{z}_n \oplus \mathbf{e}_n)], \tag{25}$$

where $d^2[a(\mathbf{z}_n), a(\mathbf{z}'_n)]$ is the SED between $\mathbf{z}_n$ and $\mathbf{z}'_n$. One can then use the all zero path to find d^2_{free} in a code search, i.e.,

$$d^2_{\text{free}} = \min \sum_n w^2(\mathbf{e}_n),$$

where the minimization is over all allowable code sequences with the exception of the all-zero sequence.

Since there are 2^{k+1} values of $\mathbf{e}_n$, there are a total of 2^{2k+2} computations required to find all the values of $w^2(\mathbf{e}_n)$. Thus, for a rate 11/12 code with 8–D 8PSK modulation, there are nearly 17 million computations required. This can be reduced by letting $z_n^0 = 0$ (or 1) in $\mathbf{z}_n$ and minimizing (25) over all $\mathbf{z}_n = [z_n^k, \ldots, z_n^1, 0]$, as suggested in [1]. This reduces the number of computations to 2^{2k+1}. In fact it is possible to even further decrease the number of computations. It can be shown that the L output bits z_n^pcorresponding to cosets τ^p with the largest integer value can be set to zero. This is due in part to the MPSK signals being antipodal for these values. Thus the total number computations required is 2^{2k-L+1}.

In order to reduce the time needed to find d^2_{free}, we note that the trellis is equivalent to a rate $\tilde{k}/(\tilde{k}+1)$ code with $2^{k-\tilde{k}}$ parallel transitions. There are $2^{\tilde{k}+1}$

different sets of these transitions. If the minimum SEW is found for each of these sets of parallel transitions, the code search is greatly simplified, since a rate $\tilde{k}/(\tilde{k}+1)$ code is all that needs to be searched and $\tilde{k}$ is usually small. Thus, the SEW's required for a rate $\tilde{k}/(\tilde{k}+1)$ code search are

$$w^2(\mathbf{e}_n^{\tilde{k}}) = \min\ w^2(\mathbf{e}_n), \tag{26}$$

where the minimization is over all $[e_n^k, \ldots, e_n^{\tilde{k}+1}]$. The FSED for this reduced code (which we call $d_{\text{free}}^{2\ (\tilde{k})}$) can be larger than d_{free}^2 since this FSED might be limited along the parallel transitions by a MSSD of $\Delta^2_{q+\tilde{k}+1}$, i.e.,

$$d_{\text{free}}^2 = \min\ (d_{\text{free}}^{2\ (\tilde{k})}, \Delta^2_{q+\tilde{k}+1}). \tag{27}$$

The best value of $\tilde{k}$ can be determined from the FSED of the best code for the previous value of v. The search starts with $v = 1$ and $\tilde{k} = 1$. Then v is increased by one, and if the FSED of the previous best code was $d_{\text{free}}^{2\ (\tilde{k})}$, then $\tilde{k}$ remains the same. This is because the limit of the parallel transitions $\Delta^2_{q+\tilde{k}+1}$ has not yet been reached and the trellis connectivity needs to be reduced in order to increase d_{free}^2. If the FSED of the previous best code was $\Delta^2_{q+\tilde{k}+1}$, then $\tilde{k}$ is increased by one from the previous value; otherwise, the FSED and the number of nearest neighbors would remain the same. If $d_{\text{free}}^{2\ (\tilde{k})} = \Delta^2_{q+\tilde{k}+1}$ for the previous best code, then $\tilde{k}$ can remain the same or increase by one. Both values of $\tilde{k}$ must be tried in order to find the best code.

$N(d_{\text{free}}^2)$ is the number of nearest neighbors between all paths with SED of d_{free}^2. If $d_{\text{free}}^2 = d_{\text{free}}^{2\ (\tilde{k})}$, an upper bound on $N(d_{\text{free}}^2)$ can be found by determining the number (A) of paths with weight $d_{\text{free}}^{2\ (\tilde{k})}$ in the equivalent rate $\tilde{k}/(\tilde{k}+1)$ code. Let the binary error sequence which occurs along a path α, with length N_α and FSED $d_{\text{free}}^{2\ (\tilde{k})}$, be

$$\mathbf{e}^{\tilde{k}}(D) = \mathbf{e}_1^{\tilde{k}} D \oplus \mathbf{e}_2^{\tilde{k}} D^2 \oplus \cdots \oplus \mathbf{e}_{N_\alpha}^{\tilde{k}} D^{N_\alpha},\ \mathbf{e}_1^{\tilde{k}}, \mathbf{e}_{N_\alpha}^{\tilde{k}} \neq \mathbf{0},\ N_\alpha \geq 1.$$

An upper bound on $N(d_{\text{free}}^2)$ is

$$N(d^2_{\text{free}}) \leq \sum_{\alpha=1}^{A} \prod_{n=1}^{N_\alpha} \bar{m}(\mathbf{e}_n^{\tilde{k}}), \tag{28}$$

where $\bar{m}(\mathbf{e}_n^{\tilde{k}}) = \#[w^2(\mathbf{e}_n^{\tilde{k}}) = w^2(\mathbf{e}_n)]$ is the number of times that $w^2(\mathbf{e}_n^{\tilde{k}}) = w^2(\mathbf{e}_n)$ over all $[e_n^k, \ldots, e_n^{\tilde{k}+1}]$. That is, we sum the multiplicities of all possible minimum weight error events. On the other hand, if $d^2_{\text{free}} = \Delta^2_{q+\tilde{k}+1}$, then

$$N(d^2_{\text{free}}) \leq \#[\Delta^2_{q+\tilde{k}+1} = w^2(\mathbf{e}_n)] \tag{29}$$

over all $\mathbf{e}_n = [e_n^k, \ldots, e_n^{\tilde{k}+1}, 0, \ldots, 0]$. If $d^{2\ (\tilde{k})}_{\text{free}} = \Delta^2_{q+\tilde{k}+1}$, then the RHS of (28) and (29) are added to determine an upper bound on $N(d^2_{\text{free}})$.

The reason that (28) and (29) are upper bounds is that for some $\mathbf{e}_n$ and $\mathbf{z}_n$, $w^2(\mathbf{e}_n) \neq d^2[a(\mathbf{z}_n), a(\mathbf{z}_n \oplus \mathbf{e}_n)]$, due to the definition of $\omega^2(\mathbf{e}_n)$ in (25). This results in average numbers of nearest neighbours which must be determined. Equations (28) and (29) assumes the worst case and hence results in an upper bound. A precise value of $N(d^2_{\text{free}})$ for $d^2_{\text{free}} = d^{2\ (\tilde{k})}_{\text{free}}$ [17] is

$$N(d^2_{\text{free}}) = \sum_{\alpha=1}^{A} \prod_{n=1}^{N_\alpha} m(\mathbf{e}_n^{\tilde{k}}), \tag{30}$$

where

$$m(\mathbf{e}_n^{\tilde{k}}) = \sum \left(\frac{\#_{\text{all } \mathbf{z}_n}[w^2(\mathbf{e}_n^{\tilde{k}}) = d^2[a(\mathbf{z}_n), a(\mathbf{z}_n \oplus \mathbf{e}_n)]]}{2^k} \right), \tag{31}$$

the $\sum$ is over all $[e_n^k, \ldots, e_n^{\tilde{k}+1}]$ for which $w^2(\mathbf{e}_n^{\tilde{k}}) = w^2(\mathbf{e}_n)$, and the $\#$ is over all $\mathbf{z}_n = [z_n^k, \ldots, z_n^1, 0]$. That is, $m(\mathbf{e}_n^{\tilde{k}})$ is the sum of all the average number of nearest neighbors for each signal point in each set of parallel transitions. Note that the summation in (31) is upper bounded by $\bar{m}(\mathbf{e}_n^{\tilde{k}})$. Similarly, for $d^2_{\text{free}} = \Delta^2_{q+\tilde{k}+1}$,

$$N(d^2_{\text{free}}) = \sum \left(\frac{\#_{\text{all } \mathbf{z}_n}[\Delta^2_{q+\tilde{k}+1} = d^2[a(\mathbf{z}_n), a(\mathbf{z}_n \oplus \mathbf{e}_n)]]}{2^k} \right), \tag{32}$$

where the $\sum$ in (32) is over all $[e_n^k, \ldots, e_n^{\tilde{k}+1}, 0, \ldots, 0]$ for which $\Delta^2_{q+\tilde{k}+1} =$

$w^2(\mathbf{e}_n)$. If $d^{2\ (\tilde{k})}_{\text{free}} = \Delta^2_{q+\tilde{k}+1}$, then $N(d^2_{\text{free}})$ is the sum of the RHS's of (30) and (32).

Example 4.7:

For the code given in Example 4.5 we have $\tilde{k} = 2$ for a rate 7/8 code with a 6–D 8PSK type II signal set. After determining the mapping of the signal set, (25) can be used to find the SEW's for each signal point. Equation (26) determines the $w^2(\mathbf{e}^{\tilde{k}}_n)$'s that are to be used to find the best rate 2/3 codes. For this signal set $\Delta^2_{q+\tilde{k}+1} = \Delta^2_4 = 4.0$. That is, $\Delta^2_{q+\tilde{k}+1} = 4.0$ is the minimum SED that occurs between parallel transitions. Using (29), we can determine an upper bound of 19 on $N(\Delta^2_{q+\tilde{k}+1})$. In the code search for the best rate 2/3 code, there may be many codes which have the largest $d^{2\ (\tilde{k})}_{\text{free}}$ of 4.343. Thus (28) was used to determine an upper bound on $N(d^{2\ (\tilde{k})}_{\text{free}})$ for each best code using an appropriate algorithm and $\bar{m}(\mathbf{e}^{\tilde{k}}_n)$. Table 7 gives for each $\mathbf{e}^{\tilde{k}}_n$, the values of$w^2(\mathbf{e}^{\tilde{k}}_n)$ and $\bar{m}(\mathbf{e}^{\tilde{k}}_n)$ that were used in the code search. The best code with a transparency of 90° was found to have $N(d^{2\ (\tilde{k})}_{\text{free}}) \leq 432$. Thus $d^2_{\text{free}} = 4.0$ and $d^2_{\text{next}} = 4.343$, where d^2_{next} is the next smallest SED.

In order to reduce the number of codes that need to be tested in a code search algorithm, rejection rules can be used. As in [1], time reversal of the parity check polynomials can be used to reject codes. Since $w^2(\mathbf{e}^{\tilde{k}}_n)$ and $\bar{m}(\mathbf{e}^{\tilde{k}}_n)$ are used to find the best codes, Rule 2 in [1] cannot be fully exploited. In the code search, a rate $\tilde{k}/(\tilde{k}+1)$ code is used at a particular v. For some of these codes parallel transitions can occur. These codes may be rejected before the algorithms are used to generate an encoder trellis and find $d^{2\ (\tilde{k})}_{\text{free}}$. If for some input $\mathbf{x}^{\tilde{k}}_n \neq \mathbf{0}$, the inputs into the systematic encoder are all zero, then parallel transitions will occur. This is because this non-zero input will cause the state of the encoder to go from one state to the next as if a zero input had occurred. Thus parallel transitions will occur in the rate $\tilde{k}/(\tilde{k}+1)$ code, which should not have parallel transitions. That is, if for some $\mathbf{x}^{\tilde{k}}_n \neq \mathbf{0}$, $\bigoplus_{j=1}^{\tilde{k}} x^j_n \mathbf{h}^j = \mathbf{0}$, where $\mathbf{h}^j = [h^j_v, \ldots, h^j_1, h^j_0]$, then the code is rejected. Similarly, we can reject codes with parity check polynomials $\mathbf{h}^l$, $1 \leq l \leq \tilde{k}$, if $\bigoplus_{j=1}^{l} x^j_n \mathbf{h}^j = \mathbf{0}$ for some $\mathbf{x}^l_n \neq \mathbf{0}$. Rule 3 in [1] can also be used to eliminate codes.

An approximate lower bound for the symbol error probability [1] is

$$P_s(e) \gtrsim \frac{N(d^2_{\text{free}})}{L} Q\left(\sqrt{\frac{d^2_{\text{free}} R_{\text{eff}}}{2} \frac{E_b}{N_0}}\right), \tag{33}$$

where E_b/N_0 is the energy per information bit to single sided noise density ratio and $R_{\text{eff}} = (IL - q - 1)/L$ (bit/T) is the average number of information bits per 2–D signal transmitted. Thus in our code search we attempt to maximize d^2_{free} and to minimize $N(d^2_{\text{free}})$. In (33) the average multiplicity of errors is normalized to that of a 2–D signal set.

Two programs were used in the code search, one for codes with $v < \tilde{k}$ and the other for codes with $v = \tilde{k}$. For specific values of I, L, and q, $y^q(z)$, $0 \leq z \leq 2^{IL-q-1}$, was generated, using the coset representatives τ^p, $1 \leq p \leq IL$, that are given in Tables 2–6. The squared Euclidean weights $w^2(\mathbf{e}_n)$ were then calculated using (25) for all $\mathbf{e}_n$. Since the value of $\tilde{k}$ can change with each v, $w^2(\mathbf{e}^{\tilde{k}}_n)$ and $\bar{m}(\mathbf{e}^{\tilde{k}}_n)$ were computed, if neccesary, as the program went from the smallest v to the largest v.

The code search used the various rejection rules before the time consuming tasks such as finding $d^{2\,(\tilde{k})}_{\text{free}}$ (using the bi-directional search algorithm [18]) and $N(d^{2\,(\tilde{k})}_{\text{free}})$ (using a trellis search technique). A variable d^2_{lim} was used (as in [1]) to indicate the largest $d^{2\,(\tilde{k})}_{\text{free}}$ found at a particular stage in the search. Another variable N^d_{lim} was used to indicate the smallest $N(d^2_{\text{lim}})$ found during the code search with a phase transparency of $2^d\Psi$. d^2_{lim} and N^d_{lim} were set to zero and infinity, respectively, before the code search began. Alternatively, d^2_{lim} and N^d_{lim} could be set equal to the best $d^{2\,(\tilde{k})}_{\text{free}}$ and $N(d^{2\,(\tilde{k})}_{\text{free}})$ found in a previous search. This was the case when one program was used for $v = \tilde{k}$ (since we start with $v = 1$) and then the other program was used for $v < \tilde{k}$.

Any code that passed through the code rejection rules based on the parity check equation had its $d^{2\,(\tilde{k})}_{\text{free}}$ computed. If it was less then d^2_{lim}, this code was rejected and the next code searched. For those codes whose $d^{2\,(\tilde{k})}_{\text{free}}$ was the same or greater than d^2_{lim}, $N(d^{2\,(\tilde{k})}_{\text{free}})$ was then computed. Also, from the values of p_i, $0 \leq i \leq I-1$, for the signal set used, the phase transparency (d) of the code was determined. Another stage of rejection was applied to those codes that had

$d^{2\,(\tilde{k})}_{\text{free}} = d^2_{\text{lim}}$. Those codes were rejected if $N(d^{2\,(\tilde{k})}_{\text{free}}) > N^d_{\text{lim}}$. When $d^{2\,(\tilde{k})}_{\text{free}}$ was greater than d^2_{lim}, then d^2_{lim} and N^d_{lim} were set to the corresponding values of this new code. The code was then listed along with its $d^{2\,(\tilde{k})}_{\text{free}}$, $N(d^{2\,(\tilde{k})}_{\text{free}})$, and phase transparency d. A small list of codes was then produced from which the best codes could be chosen. Note that only those codes with the largest $d^{2\,(\tilde{k})}_{\text{free}}$ were accepted regardless of their phase transparency. The advantage of this is that it reduces the number of codes to be searched, usually at a cost of rejecting codes with a better phase transparency and a smaller $N(d^{2\,(\tilde{k})}_{\text{free}})$, but a reduced $d^{2\,(\tilde{k})}_{\text{free}}$.

Since Δ^2_p for each of the signal sets is given in Tables 2 to 6, it was a simple matter to determine d^2_{free} for each code. For those codes where d^2_{free} occurred along parallel transitions only, d^2_{next} is also given in the code tables, since this is equal to $d^{2\,(\tilde{k})}_{\text{free}}$. Note that since $\bar{m}(\mathbf{e}^{\tilde{k}}_n)$ was used in the code search, the $N(d^2_{\text{free}})$ given in the tables is an upper bound.

The asymptotic coding gain γ of each code compared to the uncoded case is shown in the tables, i.e.,

$$\gamma = 10\log_{10}\left(\frac{d^2_{\text{free}}}{d^2_{\text{u}}}\right) \text{ (dB)}, \tag{34}$$

where d^2_{u} is the smallest FSED of an equivalent uncoded 2–D or multi-D scheme. In nearly all cases,$d^2_{\text{u}} = \Delta^2_{q+1}$, since for codes with a non-integer R_{eff}, no equivalent 2–D MPSK code exists which has the same R_{eff}, and so the equivalent uncoded multi-D signal set is used instead. For the 8–D 8PSK signal set with $q = 3$, $R_{\text{eff}} = 2$ bit/T. Thus, a natural comparison would be against uncoded QPSK, which has $d^2_{\text{u}} = 2$. In this case, $\Delta^2_{q+1} = 2.343$, which would give lower coding gains and be inconsistent with other codes that also have $R_{\text{eff}} = 2$ bit/T. The asymptotic coding gains compared to uncoded $(M/2)$PSK are found by adding to γ the appropriate correction factor

$$\gamma_{M/2} = 10\log_{10}\left(\frac{k}{L(I-1)}\frac{\Delta^2_{q+1}}{\delta^2_1}\right) \text{ (dB)}, \tag{35}$$

as shown in the code tables. The transparency (in degrees) is also given for each code. An alternative and more abstract comparison is to uncoded $2^{R_{\text{eff}}}$PSK, as suggested by Forney [19]. The correction factor in this case is

$$\gamma_F = 10 \log_{10} \left(\frac{\Delta_{q+1}^2}{4 \sin^2(2^{-R_{\text{eff}}} \pi)} \right) \text{ (dB)}. \tag{37}$$

Codes for rate $R = 5/6$ and 4/5, 4–D TC-8PSK are listed in Tables 8(a) and 8(b), respectively. Equivalent $R = 5/6$, 4–D TC-8PSK codes with up to 16 states have been found independently by LaFanéchere and Costello [6] and by Wilson [7], although with reduced phase transparency. Rate $R = 8/9$, 7/8, and 6/7, 6–D TC-8PSK codes are given in Tables 9(a), 9(b), and 9(c), respectively. Rate $R =$ 11/12, 10/11, 9/10, and 8/9, 8–D TC-8PSK codes are listed in Tables 10(a), 10(b), 10(c), and 10(d), respectively. Rate $R = 7/8$ and 6/7, 4–D TC-16PSK codes are given in Tables 11(a) and 11(b), respectively. Finally, rate $R = 11/12$ and 10/11, 6–D TC-16PSK codes are listed in Tables 12(a) and 12(b), respectively. The multi-D, 2 state TC-8PSK and TC-16PSK codes were also found by Divsalar and Simon [20]. The parity check polynomials are expressed in octal notation in the code tables, e.g., $H^0(D) = D^6 + D^4 + D^2 + D + 1 \equiv (001\ 010\ 111)_2 \equiv (127)_8$.

4.6 Decoder implementation

When the Viterbi algorithm is used in the decoder implementation, a measure of decoding complexity is given by $2^{v+\tilde{k}}/L$. This is the number of distinct transitions in the trellis diagram for TCM schemes normalized to a 2–D signal set. The maximum bit rate of the decoder is kf_d, where f_d is the symbol speed of the decoder. Since k is quite large for multi-D signal sets (at least $(I-1)L$), high bit rates can be achieved. For example, a Viterbi decoder has been constructed for a rate 7/9 periodically time varying trellis code (PTVTC) with $v = 4$, $\tilde{k} = 2$, and 8PSK modulation [21]. This decoder has $f_d = 60$ MHz and a bit rate of 140 Mbit/s, where f_d equals the 2–D symbol rate. However, with the equivalent rate 7/8 code with 6–D 8PSK modulation, the bit rate will be $L = 3$ times as fast, i.e., 420 Mbit/s. The branch metric calculator, though, will be more complicated due to the larger number of parallel transitions between states. Alternatively, one could build a decoder at 20 MHz for the same bit rate of 140 Mbit/s. In addition to providing decreased decoder complexity, this multi-D code has an asymptotic coding gain which is 0.56 dB greater and is 90° transparent, compared with a 180° transparency for the PTVTC [22].

Although the decoding complexity of the Viterbi algorithm is measured in terms of $2^{v+\tilde{k}}/L$, for multi-D schemes the complexity of subset (parallel

transition) decoding must also be taken into account due to the large number of parallel transitions. For the multi-D TC-MPSK codes considered here, since the subsets are block codes in signal space (the principle subset) or cosets of a block code, the suboptimum algorithm proposed in [13] can be used to decode each subset. At high signal to noise ratios, this algorithm is only slightly inferior to optimum decoding, while the subset decoding complexity is significantly reduced. Optimum decoding requires $2^{k-\tilde{k}} - 1$ branch comparisons, while suboptimum decoding requires $\sum_i (2^{L-m_i} - 1)$ comparisons to decode a subset at partition level $p = \tilde{k} + q + 1$. For example, for the 3.67 bit/T 6–D 16PSK code with 16 states given in Table 12(a) where $\tilde{k} = 3$, we require $2^{k-\tilde{k}} - 1 = 2^8 - 1 = 255$ comparisons for an optimum decoder. As $q = 0$ the partition level is $p = 4$ and from Table 6(a) we have $m_0 = 3$, $m_1 = 1$, and $m_2 = m_3 = 0$. Therefore a suboptimal decoder only requires $\sum_i (2^{L-m_i} - 1) = 17$ comparisons. Thus a reduction in the number of comparisons of $255/17 = 15$ times can be made between an optimum and sub-optimum decoders.

4.7 Discussion

The asymptotic coding gains for all the codes obtained have been plotted against complexity factor $\beta = v + \tilde{k} - \log_2 L$ in Figures 12 and 13. Note that these graphs do not take into account complexity due to parallel transitions. In Figure 12(a), a plot of all the best codes found for 8PSK modulation and 2 bit/T is shown. The 2–D codes are from [23]. Notice that the one state or "uncoded" codes are shown as well. Although the multi-D codes with one state have negative complexity, the 8–D uncoded case has a coding gain above 0 dB. These one state codes correspond to simple block coded modulation schemes that have recently become an active research area [13–16]. Those codes marked with an asterisk indicate that these are the best codes found in an incomplete code search. Those marked with a question mark are an attempt to predict the coding gains of higher complexity codes that have yet to be found. A set of prediction rules was used, where

$$d^2_{\text{free}} \leq \min(2\Delta^2_{q+1} + (v - \tilde{k})\Delta^2_q, \Delta^2_{q+\tilde{k}+1}) \text{ for } \Delta^2_q + \Delta^2_{q+1} < \Delta^2_{q+2},\ v > \tilde{k},$$
$$\text{or } d^2_{\text{free}} \leq \min(\Delta^2_{q+1} + \Delta^2_{q+2} + (v - \tilde{k} - 1)\Delta^2_q, \Delta^2_{q+\tilde{k}+1}),$$
$$\text{for } \Delta^2_q + \Delta^2_{q+1} > \Delta^2_{q+2},\ v > \tilde{k} + 1.$$

These rules attempt to predict the free distance based on observations of how d^2_{free} increased in the code tables and on a knowledge of the incremental SED leaving or entering a state. For large v, these rules can be tightened when there isn't an equality. For example, we would expect that for $v = 5$, the 4–D 8PSK rate 5/6 code would have $d^2_{\text{free}} \leq 7\delta_0^2 = 4.101$. However in reality the equality is not reached, since $d^2_{\text{free}} = 6\delta_0^2$. Thus the former equation from above should be modified to

$$d^2_{\text{free}} \leq \min(2\Delta^2_{q+1} + (v - \tilde{k} - 1)\Delta^2_q, \Delta^2_{q+\tilde{k}+1}).$$

This technique was used with general success in the code search to predict the values of $\tilde{k}$ for each v. Thus, with a few calculations by hand, an idea of how long a code search will be, as well as the achievable coding gain, can be obtained before doing the actual code search.

Also note from Figure 12(a) that for good codes with $v = 2$, as L increases the complexity decreases and γ increases, eventually reaching 3.0 dB for $L = 4$. Thus, for the 8–D signal set, the complexity factor can be reduced by a factor of four, while maintaining γ, compared to the rate 2/3 code with $v = 2$. Beyond $\beta = 4$ (and $\gamma = 3.0$ dB), increases in coding gain are possible with the new codes that have been found. For $L = 1$, the rate of increase of γ with β seems to be slower than for $L = 4$. With $L = 4$, a "code barrier" of $\gamma = 6.0$ dB will be reached due to the nature of the set partitioning. It would seem that very complex codes are required ($\beta \geq 14$) if this 6.0 dB limit is to be broken. The codes for L=2 also seem to trend towards this barrier. Although this code barrier appears difficult to break, the previous codes found indicate that it can be reached faster than for $L = 1$, perhaps with a complexity factor reduction of four. These large complexity codes may be of interest in deep-space communication systems. The effort needed to build such a system may be justified, as indicated by the extremely large Viterbi decoder now being constructed at JPL [24].

Figure 13(a) compares the 4–D codes with 3 bit/T and 16PSK modulation to the equivalent 2–D codes [23]. For low β, the same effect observed for 8PSK and 2 bit/T seems to be occuring. That is, β is decreasing and γ is increasing as L increases. Between $\beta = 3$ and $\beta = 9$ the codes are close together, with perhaps a divergance at $\beta = 10$ as indicated. In Figure 12(b) a variety of curves are shown

for 8PSK modulation. Notably, the same low β effect occurs for the curves at 2.5 bit/T. The other two curves have a rate of $(IL-1)/IL$ (as do the 2.5 bit/T curves). They start off at $\gamma = 0$ (for $v = 1$) and increase steadily. The curves for 3.5 and 3.67 bit/T and 16PSK modulation (figure 13(b)) seem to follow the same type of pattern as the 2.5 and 2.67 bit/T curves, respectively.

In Figure 12(c), codes of rates $[(I-1)L+1]/[(I-1)L+2]$ with 2.25 and 2.33 bit/T and 8PSK modulation are shown. These rates seem to be characterized by a quick increase of γ with β and then a levelling off between $\gamma = 3$ and $\gamma = 4$ dB. The apparent low coding gains are due to the fairly large d_u^2 they are compared with. The 3.33 bit/T codes with 16PSK modulation in Figure 13(b) also seem to follow a similar pattern to the codes in Figure 12(c).

Rate $k/(k+1)$, $2L$–D, TC-MPSK codes also have the potential advantage of being used as inner codes in a high rate concatenated coding system with Reed-Solomon (RS) outer codes over $GF(2^k)$. If the inner decoder makes errors, one trellis branch error will exactly match one symbol in the outer RS code word. It is shown in [12] that the symbol oriented nature of multi-D TC-MPSK inner codes can provide an improvement of up to 1 dB in the overall performance of a concatenated coding system when these codes replace bit oriented 2–D TC-MPSK inner codes of the same rate.

5 Conclusions

A means of systematically constructing multi-dimensional MPSK signal sets has been described. When these signal sets are combined with trellis coded modulation to form a rate $k/(k+1)$ code, significant asymptotic coding gains in comparison to an uncoded system can be achieved. These codes provide a number of significant advantages compared to trellis codes with 2–D signal sets. Most importantly, R_{eff} can vary from $I-1$ to $I-1/L$ bit/T, allowing the coding system designer a greater choice in data rate without sacrificing data quality. As R_{eff} approaches I, though, increased coding effort (in terms of decoder complexity) or higher SNR is required to achieve the same data quality.

Since the signal sets have been systematically constructed by using block code cosets, and systematic convolutional coding is used, a powerful total encoder system concept results. This approach has led to the construction of signal sets that allow codes to be transparent to discrete $360°/M$ phase rotations, in amounts depending on the code and the signal set used. In general, it has been found that increasing phase transparency usually results in a decrease in steady state code performance, due to an increase in the number of nearest neighbors or a decrease in free distance. A complete encoder system, from the differential precoder to the 2–D signal mapper, is presented, allowing an easy application of these codes.

Another advantage is decoder complexity. Using the Viterbi algoithm, very high bit rates can be achieved due to the high values of k compared to convolutional codes that map into a 2–D signal set only. The many branch metric computations in the Viterbi decoder can be reduced either through the use of a sub-optimal comparison technique or large look up tables. Multi-D codes are also suited for concatenated coding with a Reed-Solomon outer code. A synergistic effect is obtained, since the multi-D codes tend to produce errors in blocks, which are matched to the RS code symbol size.

Finally, this method of set partitioning and code construction can be applied to other signal sets such as QAM or QPSK. It is expected that similar coding gains will be achieved in comparison to existing codes with multi-D QAM signal sets. However, the advantages of the systematic approach described in this paper, we believe, will lead to faster acceptance and utilization of these multi-D codes.

6 References

[1] G. Ungerboeck, "Channel Coding with multilevel/ phase signals," *IEEE Trans. Inform. Theory*, Vol. IT-28, pp. 55-67, January 1982.

[2] G. D. Forney, Jr., R. G. Gallager, G. R. Lang, F. M. Longstaff, and S. U. Qureshi, "Efficient modulation for band-limited channels," *IEEE Trans. Selected Areas in Comm.*, Vol. SAC-2, pp. 632–647, Sept. 1984.

[3] A. R. Calderbank and J. E. Mazo, "A new description of trellis codes," *IEEE Trans. Inform. Theory*, Vol. IT-30, pp. 784–791, Nov. 1984.

[4] A. R. Calderbank and N. J. A. Sloane, "Four-dimensional modulation with an eight state trellis code," *AT & T Tech. Journal*, Vol. 64, pp. 1005–1017, May-June 1985.

[5] L. F. Wei, "Trellis-coded modulation with multi-dimensional constellations," *IEEE Trans. Inform. Theory*, Vol. IT-33, pp. 483–501, July 1987.

[6] A. LaFanéchere and D. J. Costello, Jr., "Multidimensional coded PSK systems using unit-memory trellis codes," *Proc. Allerton Conf. on Commun., Cont., and Comput.*, pp. 428–429, Monticello, IL, Sept. 1985.

[7] S. G. Wilson, "Rate 5/6 trellis-coded 8–PSK," *IEEE Trans. Commun.*, Vol. COM-34, pp. 1045–1049, Oct. 1986.

[8] A. R. Calderbank and N. J. A. Sloane, "New trellis codes based on lattices and cosets," *IEEE Trans. Inform. Theory*, Vol. IT-33, pp. 177–195, March 1987.

[9] G. D. Forney, Jr., "Coset Codes," *IEEE Trans. Inform. Theory*, to appear.

[10] D. P. Taylor and H. C. Chan, "A simulation of two bandwidth efficient modulation techniques," *IEEE Trans. Commun.*, Vol. COM-29, pp. 267–275, March 1981.

[11] S. G. Wilson, H. A. Sleeper, P. J. Schottler, and M. T. Lyons, "Rate 3/4 convolutional coding of 16PSK: code design and performance study," *IEEE Trans. Commun.*, Vol. COM-32, pp. 1308–1315, Dec. 1984.

[12] R. H. Deng and D. J. Costello, Jr., "High rate concatenated coding systems using multi-dimensional bandwidth efficient trellis inner codes," *IEEE Trans. Commun.*, to appear.

[13] S. I. Sayegh, "A class of optimum block codes in signal space," *IEEE Trans. Commun.*, Vol. COM-34, pp. 1043–1045, Oct. 1986.

[14] E. L. Cusack, "Error control codes for QAM signalling", *Electronics Letters*, Vol. 20, No. 2, pp. 62–63, 19 Jan. 1984.

[15] R. M. Tanner, "Algebraic construction of large euclidean distance combined coding/modulation systems," Computer Research Laboratory Technical Report, University of California, UCSC-CRL-87–7, 5 June 1987.

[16] S. Lin, "Bandwidth efficient block codes for M-ary PSK modulation," NASA Technical Report, Grant No. NAG 5–931, University of Hawaii, Dec. 1987.

[17] M. Rouanne, "Distance bounds and construction algorithms for trellis codes," PhD dissertation, University of Notre Dame, April 1988.

[18] K. J. Larson, "Comments on 'An efficient algorithm for computing free distance'," *IEEE Trans. Inform. Theory*, Vol. IT-18, pp. 437–439, May 1972.

[19] G. D. Forney, Jr., private communication, March 1987.

[20] D. Divsalar and M. K. Simon, "Multiple trellis coded modulation (MTCM)," *IEEE Trans. Commun.*, Vol. COM–36, No. 4, pp. 410–419, April 1988.

[21] F. Hemmati and R. J. F. Fang, "Low complexity coding methods for high data rate channels," *Comsat Technical Review*, Vol. 16, pp. 425–447, Fall 1986.

[22] S. S. Pietrobon, "Rotationally invariant convolutional codes for MPSK modulation and implementation of Viterbi decoders," Masters Thesis, South Australian Institute of Technology, June 1988.

[23] G. Ungerboeck, "Trellis-coded modulation with redundant signal sets," *IEEE Commun. Magazine*, Vol. 25, pp. 5–21, Feb. 1987.

[24] O. Collins, "Techniques for long constraint length Viterbi decoding," California Institute of Technology, submission to IEEE Information Theory Symposium, Kobe City, Japan, June 1988.

Table 1: A 2×3 Binary Matrix Space Partition

Partition Level (p)	Principal Subsets (Ω^p)	Generator Matrices $\mathbf{G}_{m_2}$	$\mathbf{G}_{m_1}$	$\mathbf{G}_{m_0}$	Coset Representatives $(\tau^p)^T$
0	$\Omega(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_0)$	$\mathbf{G}_0$	$\mathbf{G}_0$	$\mathbf{G}_0$	–
1	$\Omega(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_1)$	$\mathbf{G}_0$	$\mathbf{G}_0$	$\mathbf{G}_1$	(01)
2	$\Omega(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_2)$	$\mathbf{G}_0$	$\mathbf{G}_0$	-	(11)
3	$\Omega(\mathbf{C}_0, \mathbf{C}_1, \mathbf{C}_2)$	$\mathbf{G}_0$	$\mathbf{G}_1$	-	(02)
4	$\Omega(\mathbf{C}_0, \mathbf{C}_2, \mathbf{C}_2)$	$\mathbf{G}_0$	-	-	(22)
5	$\Omega(\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_2)$	$\mathbf{G}_1$	-	-	(04)
6	$\Omega(\mathbf{C}_2, \mathbf{C}_2, \mathbf{C}_2)$	-	-	-	(44)

Note: $\mathbf{G}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $\mathbf{G}_1 = [\,1\ 1\,]$, $\tau_1 = [\,0\ 1\,]^T$ $\tau_2 = [\,1\ 1\,]^T$

Table 2: A 4-D 8PSK Signal Set Partition

Partition Level (p)	Principal Subsets	MSSD (Δ_p^2)	Coset Representatives $(\tau^p)^T$
0	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_0)$	0.586	–
1	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_1)$	1.172	(01)
2	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_2)$	2	(11)
3	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_1, \mathbf{C}_2)$	4	(02)
4	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_2, \mathbf{C}_2)$	4	(22)
5	$\mathbf{P}(\mathbf{C}_1, \mathbf{C}_2, \mathbf{C}_2)$	8	(04)
6	$\mathbf{P}(\mathbf{C}_2, \mathbf{C}_2, \mathbf{C}_2)$	∞	(44)

$\mathbf{C}_0 : (2,2)$ code , $\mathbf{G}_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $d_0 = 1$

$\mathbf{C}_1 : (2,1)$ code , $\mathbf{G}_1 = [\,1\ 1\,]$, $d_1 = 2$ $\tau_1 = [\,0\ 1\,]^T$

$\mathbf{C}_2 : (2,2)$ code , $d_2 = \infty$ $\tau_2 = [\,1\ 1\,]^T$

$p_0 = 2,\ p_1 = 4,\ p_2 = 6$

Table 3(a): 6-D 8PSK Signal Set Partition I

Partition Level (p)	Principal Subsets	MSSD (Δ_p^2)	Coset Representatives $(\tau^p)^T$
0	$\mathbf{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0)$	0.586	-
1	$\mathbf{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_1^1)$	1.172	(111)
2	$\mathbf{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_2^1)$	1.172	(110)
3	$\mathbf{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_3)$	2	(011)
4	$\mathbf{P}(\mathrm{C}_0, \mathrm{C}_1^1, \mathrm{C}_3)$	4	(222)
5	$\mathbf{P}(\mathrm{C}_0, \mathrm{C}_2^1, \mathrm{C}_3)$	4	(220)
6	$\mathbf{P}(\mathrm{C}_0, \mathrm{C}_3, \mathrm{C}_3)$	4	(022)
7	$\mathbf{P}(\mathrm{C}_1^1, \mathrm{C}_3, \mathrm{C}_3)$	8	(444)
8	$\mathbf{P}(\mathrm{C}_2^1, \mathrm{C}_3, \mathrm{C}_3)$	8	(440)
9	$\mathbf{P}(\mathrm{C}_3, \mathrm{C}_3, \mathrm{C}_3)$	∞	(044)

$\mathrm{C}_0 : (3,3)$ code , $\quad \mathbf{G}_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $\quad d_0 = 1$

$\mathrm{C}_1^1 : (3,2)$ code , $\quad \mathbf{G}_1^1 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$, $\quad d_1^1 = 2$; $\quad \tau_1^1 = [\,1\ 1\ 1\,]^T$

$\mathrm{C}_2^1 : (3,1)$ code , $\quad \mathbf{G}_2^1 = [\,0\ 1\ 1\,]$, $\quad d_2^1 = 2$; $\quad \tau_2^1 = [\,1\ 1\ 0\,]^T$

$\mathrm{C}_3 : (3,0)$ code , $\quad d_3 = \infty$; $\quad \tau_3^1 = [\,0\ 1\ 1\,]^T$

$p_0 = 1, \ p_1 = 4, \ p_2 = 7$

Table 3(b): 6-D 8PSK Signal Set Partition II

Partition Level (p)	Principal Subsets	MSSD (Δ_p^2)	Coset Representatives $(\tau^p)^T$
0	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_0)$	0.586	–
1	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_1^2)$	0.586	(001)
2	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_2^2)$	1.757	(011)
3	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_3)$	2	(111)
4	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_1^1, \mathbf{C}_3)$	4	(222)
5	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_2^1, \mathbf{C}_3)$	4	(220)
6	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_3, \mathbf{C}_3)$	4	(022)
7	$\mathbf{P}(\mathbf{C}_1^1, \mathbf{C}_3, \mathbf{C}_3)$	8	(444)
8	$\mathbf{P}(\mathbf{C}_2^1, \mathbf{C}_3, \mathbf{C}_3)$	8	(440)
9	$\mathbf{P}(\mathbf{C}_3, \mathbf{C}_3, \mathbf{C}_3)$	∞	(044)

$\mathbf{C}_0 : (3,3)$ code , $\mathbf{G}_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $d_0 = 1$

$\mathbf{C}_1^2 : (3,2)$ code , $\mathbf{G}_1^2 = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$, $d_1^2 = 1$; $\tau_1^2 = [\,0\;0\;1\,]^T$

$\mathbf{C}_2^2 : (3,1)$ code , $\mathbf{G}_2^2 = [\,1\;\;1\;\;1\,]$, $d_2^2 = 3$; $\tau_2^2 = [\,0\;1\;1\,]^T$

$\mathbf{C}_3 : (3,0)$ code , $d_3 = \infty$; $\tau_3^2 = [\,1\;1\;1\,]^T$

Other codes are from Table 3(a).

$p_0 = 3,\;\; p_1 = 4,\;\; p_2 = 7$

Table 4: 8-D 8PSK Signal Set Partition

Partition Level (p)	Principal Subsets	MSSD (Δ_p^2)	Coset Representatives $(\tau^p)^T$
0	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0)$	0.586	–
1	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_1)$	1.172	(0001)
2	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_2)$	1.172	(0011)
3	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_3)$	2	(0101)
4	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_1, \mathrm{C}_3)$	2.343	(0002)
5	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_1, \mathrm{C}_4)$	4	(1111)
6	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_2, \mathrm{C}_4)$	4	(0022)
7	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_3, \mathrm{C}_4)$	4	(0202)
8	$\mathrm{P}(\mathrm{C}_1, \mathrm{C}_3, \mathrm{C}_4)$	8	(0004)
9	$\mathrm{P}(\mathrm{C}_1, \mathrm{C}_4, \mathrm{C}_4)$	8	(2222)
10	$\mathrm{P}(\mathrm{C}_2, \mathrm{C}_4, \mathrm{C}_4)$	8	(0044)
11	$\mathrm{P}(\mathrm{C}_3, \mathrm{C}_4, \mathrm{C}_4)$	16	(0404)
12	$\mathrm{P}(\mathrm{C}_4, \mathrm{C}_4, \mathrm{C}_4)$	∞	(4444)

$$\mathrm{C}_0 : (4,4) \text{ code}, \quad \mathrm{G}_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad d_0 = 1$$

$$\mathrm{C}_1 : (4,3) \text{ code}, \quad \mathrm{G}_1 = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}, \quad d_1 = 2; \quad \tau_1 = [\,0\,0\,0\,1\,]^T$$

$$\mathrm{C}_2 : (4,2) \text{ code}, \quad \mathrm{G}_2 = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}, \quad d_2 = 2; \quad \tau_2 = [\,0\,0\,1\,1\,]^T$$

$$\mathrm{C}_3 : (4,1) \text{ code}, \quad \mathrm{G}_3 = [\,1 \;\; 1 \;\; 1 \;\; 1\,], \quad d_3 = 4; \quad \tau_3 = [\,0\,1\,0\,1\,]^T$$

$$\mathrm{C}_4 : (4,0) \text{ code}, \quad d_4 = \infty; \quad \tau_4 = [\,1\,1\,1\,1\,]^T$$

$p_0 = 5, \;\; p_1 = 9, \;\; p_2 = 12.$

Table 5: 4-D 16PSK Signal Set Partition

Partition Level (p)	Principal Subsets	MSSD (Δ_p^2)	Coset Representatives $(\tau^p)^T$
0	$\mathbf{P}(\mathbf{C_0}, \mathbf{C_0}, \mathbf{C_0}, \mathbf{C_0})$	0.152	–
1	$\mathbf{P}(\mathbf{C_0}, \mathbf{C_0}, \mathbf{C_0}, \mathbf{C_1})$	0.304	(01)
2	$\mathbf{P}(\mathbf{C_0}, \mathbf{C_0}, \mathbf{C_0}, \mathbf{C_2})$	0.586	(11)
3	$\mathbf{P}(\mathbf{C_0}, \mathbf{C_0}, \mathbf{C_1}, \mathbf{C_2})$	1.172	(02)
4	$\mathbf{P}(\mathbf{C_0}, \mathbf{C_0}, \mathbf{C_2}, \mathbf{C_2})$	2	(22)
5	$\mathbf{P}(\mathbf{C_0}, \mathbf{C_1}, \mathbf{C_2}, \mathbf{C_2})$	4	(04)
6	$\mathbf{P}(\mathbf{C_0}, \mathbf{C_2}, \mathbf{C_2}, \mathbf{C_2})$	4	(44)
7	$\mathbf{P}(\mathbf{C_1}, \mathbf{C_2}, \mathbf{C_2}, \mathbf{C_2})$	8	(08)
8	$\mathbf{P}(\mathbf{C_2}, \mathbf{C_2}, \mathbf{C_2}, \mathbf{C_2})$	∞	(88)

Codes are from Table 2

$p_0 = 2, \quad p_1 = 4, \quad p_2 = 6, \quad p_3 = 8$

Table 6(a): 6-D 16PSK Signal Set Partition I

Partition Level (p)	Principal Subsets	MSSD (Δ_p^2)	Coset Representatives $(\tau^p)^T$
0	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_0)$	0.152	–
1	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_1^1)$	0.304	(111)
2	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_2^1)$	0.304	(110)
3	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_3)$	0.586	(011)
4	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_1^1, \mathbf{C}_3)$	1.172	(222)
5	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_2^1, \mathbf{C}_3)$	1.172	(220)
6	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_0, \mathbf{C}_3, \mathbf{C}_3)$	2	(022)
7	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_1^1, \mathbf{C}_3, \mathbf{C}_3)$	4	(444)
8	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_2^1, \mathbf{C}_3, \mathbf{C}_3)$	4	(440)
9	$\mathbf{P}(\mathbf{C}_0, \mathbf{C}_3, \mathbf{C}_3, \mathbf{C}_3)$	4	(044)
10	$\mathbf{P}(\mathbf{C}_1^1, \mathbf{C}_3, \mathbf{C}_3, \mathbf{C}_3)$	8	(888)
11	$\mathbf{P}(\mathbf{C}_2^1, \mathbf{C}_3, \mathbf{C}_3, \mathbf{C}_3)$	8	(880)
12	$\mathbf{P}(\mathbf{C}_3, \mathbf{C}_3, \mathbf{C}_3, \mathbf{C}_3)$	∞	(088)

Codes are from Table 3(a)

$p_0 = 1, \quad p_1 = 4, \quad p_2 = 7, \quad p_3 = 10$

Table 6(b): 6-D 16PSK Signal Set Partition II

Partition Level (p)	Principal Subset	MSSD (Δ_p^2)	Coset Representatives $(\tau^p)^T$
0	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0)$	0.152	–
1	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_1^2)$	0.152	(001)
2	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_2^2)$	0.457	(011)
3	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_3)$	0.586	(111)
4	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_1^1, \mathrm{C}_3)$	1.172	(222)
5	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_2^1, \mathrm{C}_3)$	1.172	(220)
6	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_3, \mathrm{C}_3)$	2	(022)
7	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_1^1, \mathrm{C}_3, \mathrm{C}_3)$	4	(444)
⋮	⋮	⋮	⋮

Codes are from Tables 3(a) and 3(b)
$p_0 = 3,\ \ p_1 = 4,\ \ p_2 = 7,\ \ p_3 = 10$

Table 6(c): 6-D 16PSK Signal Set Partition III

Partition Level (p)	Principal Subsets	MSSD (Δ_p^2)	Coset Representatives $(\tau^p)^T$
0	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0)$	0.152	–
1	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_1^2)$	0.152	(001)
2	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_2^2)$	0.457	(011)
3	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_3)$	0.586	(111)
4	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_1^2, \mathrm{C}_3)$	0.586	(002)
5	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_3^2, \mathrm{C}_3)$	1.757	(022)
6	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_0, \mathrm{C}_3, \mathrm{C}_3)$	2	(222)
7	$\mathrm{P}(\mathrm{C}_0, \mathrm{C}_1^1, \mathrm{C}_3, \mathrm{C}_3)$	4	(444)
⋮	⋮	⋮	⋮

Codes are from Tables 3(a) and 3(b)
$p_0 = 3,\ \ p_1 = 6,\ \ p_2 = 7,\ \ p_3 = 10.$

Table 7: Squared Euclidean Weights Used in Code Search for Rate 7/8 Code with 6-D 8PSK Signal Set II ($\tilde{k} = 2$).

$\mathbf{e}_n^{\tilde{k}}$	$w^2(\mathbf{e}_n^{\tilde{k}})$	$\bar{m}(\mathbf{e}_n^{\tilde{k}})$	$m(\mathbf{e}_n^{\tilde{k}})$
000	0.0	1	1
001	1.172	3	2
010	1.757	9	4
011	0.586	3	1
100	2.0	16	6
101	1.172	12	2
110	1.757	18	4
111	0.586	6	1

Table 8: 4-D (L=2) Trellis Coded 8PSK

8(a) $R = 5/6$, $R_{\text{eff}} = 2.5$ bit/T, $d_u^2 = 1.172$

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^2	h^1	h^0	Transparency $(2^d\Psi)$
1	1.757	81	–	–	1.76	–	2	3	90°
2	2.0	6	2.929	243	2.32	–	2	5	90°
3	2.929	198	–	–	3.98	02	06	13	45°
		180	–	–		04	02	17	90°
4	3.515	2079	–	–	4.77	16	12	23	45°
		1944	–	–		16	04	23	90°
5	3.515	252	–	–	4.77	32	22	57	45°
		144	–	–		26	04	53	90°
6	4.0	6	4.101	5058	5.33	004	030	127	45°
				2160		060	004	127	90°

$$\gamma_4 = -1.35\ dB, \quad \gamma_F = 0.23\ dB$$

8(b) $R = 4/5$, $R_{\text{eff}} = 2$ bit/T, $d_u^2 = 2.0$

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^3	h^2	h^1	h^0	Transparency $(2^d\Psi)$
1	3.172	36	–	–	2.00	–	–	2	3	45°
2	4.0	6	5.172	108	3.01	–	–	2	5	45°
3	4.0	2	5.172	64	3.01	–	04	02	17	90°
4	5.172	34	–	–	4.13	10	14	06	25	90°
		30	–	–		10	04	02	23	180°
5	6.0	6	–	–	4.77	14	24	06	43	90°
6	6.343	56	–	–	5.01	070	044	046	143	90°
		45	–	–		070	034	076	105	180°

$$\gamma_4 = 0\ dB, \quad \gamma_F = 0\ dB$$

Table 9: 6-D (L=3) Trellis Coded 8PSK

9(a) $R = 8/9$, $R_{\text{eff}} = 2.67$ bit/T, $d_u^2 = 1.172$

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^3	h^2	h^1	h^0	Transparency $(2^d\Psi)$
1	1.172	15	1.757	1512	0.0	–	–	2	3	45° (I)
2	1.757	432	–	–	1.76	–	–	2	5	45° (II)
3	2.0	16	2.343	225	2.32	–	04	02	11	45° (I)
4	2.343	81	–	–	3.01	14	04	02	23	90° (I)
		63	–	–		10	06	04	21	180° (I)
5	2.929	3969	–	–	3.98	14	24	02	77	90° (I)
*6	2.929	594	–	–	3.98	066	026	012	101	90° (I)

$\gamma_4 = -1.07\ dB$, $\gamma_F = 1.14\ dB$

* Search incomplete.

9(b) $R = 7/8$, $R_{\text{eff}} = 2.33$ bit/T, $d_u^2 = 1.757$.

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^4	h^3	h^2	h^1	h^0	Transparency $(2^d\Psi)$
1	2.0	16	2.343	243	0.56	–	–	–	2	3	90° (II)
2	2.586	48	–	–	1.68	–	–	6	4	7	90° (II)
3	3.757	144	–	–	3.30	–	–	04	02	11	180° (II)
4	4.0	19	4.343	432	3.57	–	–	14	02	33	90° (II)
5	4.0	7	4.343	360	3.57	–	30	14	26	41	90° (II)
				252		–	16	34	06	41	180° (II)
*6	4.0	3	4.343	260	3.57	074	14	024	002	101	90° (II)

$\gamma_4 = 0.11\ dB$, $\gamma_F = 1.10\ dB$

9(c) $R = 6/7$, $R_{\text{eff}} = 2.0$ bit/T, $d_u^2 = 2.0$

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^4	h^3	h^2	h^1	h^0	Transparency ($2^d\Psi$)
1	3.757	288	–	–	2.74	–	–	–	2	3	180° (II)
2	4.0	19	5.757	2304	3.01	–	–	–	2	5	180° (II)
3	4.0	7	5.757	1488	3.01	–	–	02	06	13	90° (II)
				1200		–	–	04	02	17	180° (II)
4	4.0	3	5.757	576	3.01	–	10	04	06	25	90° (II)
				528		–	10	04	02	33	180° (II)
*5	5.757	272	–	–	4.59	60	24	20	06	11	180° (II)
*6	5.757	80	–	–	4.59	060	050	006	002	101	180° (II)

$\gamma_4 = 0\ dB$, $\gamma_F = 0\ dB$

Table 10: 8-D (L=4) Trellis Coded 8PSK

10(a) $R = 11/12$, $R_{\text{eff}} = 2.75$ bit/T, $d_u^2 = 1.172$

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^3	h^2	h^1	h^0	Transparency ($2^d\Psi$)
1	1.172	54	1.757	14580	0.0	–	–	2	3	45°
2	1.757	1944	–	–	1.76	–	6	4	5	45°
3	2.0	26	2.343	2916	2.32	–	04	02	11	45°
4	2.343	963	–	–	3.01	10	06	04	21	45°
5	2.343	27	2.929	110376	3.01	34	16	10	45	45°

$\gamma_4 = -0.94\ dB$, $\gamma_F = 1.60\ dB$

10(b) $R = 10/11$, $R_{\text{eff}} = 2.5$ bit/T, $d_u^2 = 1.172$.

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^3	h^2	h^1	h^0	Transparency ($2^d\Psi$)
1	2.0	26	2.343	5832	2.32	–	–	2	3	45°
2	2.343	2043	–	–	3.01	–	4	6	7	45°
3	2.343	27	3.172	312	3.01	–	04	02	11	45°
4	3.172	78	–	–	4.33	14	04	02	21	45°
5	3.515	4779	–	–	4.77	14	30	02	41	45°
		4428	–	–		16	24	10	47	90°
*6	4.0	52	4.343	9828	5.33	030	050	026	101	45°

$\gamma_4 = -1.35\ dB$, $\gamma_F = 0.23\ dB$

10(c) $R = 9/10$, $R_{\text{eff}} = 2.25$ bit/T, $d_u^2 = 2.0$

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^4	h^3	h^2	h^1	h^0	Transparency ($2^d\Psi$)
1	2.343	27	3.172	1092	0.69	–	–	–	2	3	45°
2	3.172	156	–	–	2.00	–	–	6	4	5	45°
3	4.0	52	4.343	1404	3.01	–	–	06	02	11	45°
			4.686	1458		–	–	02	06	11	90°
4	4.0	16	4.686	648	3.01	–	04	06	16	21	45°
*5	4.0	4	4.343	24	3.01	34	10	14	02	41	45°
			4.686	306		24	14	04	06	41	90°

$\gamma_4 = 0.51\ dB$, $\gamma_F = 1.23\ dB$

10(d) $R = 8/9$, $R_{\text{eff}} = 2.0$ bit/T, $d_u^2 = 2.0$

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^4	h^3	h^2	h^1	h^0	Transparency ($2^d\Psi$)
0	2.343	27	–	–	0.69	–	–	–	–	–	45°
1	4.0	52	4.686	1458	3.01	–	–	–	2	3	90°
2	4.0	16	4.686	648	3.01	–	–	2	6	7	45°
3	4.0	4	4.686	288	3.01	–	04	02	12	11	45°
4	5.172	12	–	–	4.13	06	20	02	06	21	45°
*5	7.024	51	–	–	5.46	22	30	24	02	43	90°

$\gamma_4 = 0\ dB$, $\gamma_F = 0\ dB$

Table 11: 4-D (L=2) Trellis Coded 16PSK

11(a) $R = 7/8$, $R_{\text{eff}} = 3.5$ bit/T, $d_u^2 = 0.304$

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^2	h^1	h^0	Transparency $(2^d\Psi)$
1	0.457	256	–	–	1.76	–	2	3	45°
2	0.586	12	0.761	1024	2.84	–	2	5	45°
3	0.761	864	–	–	3.98	02	06	13	22.5°
		672	–	–		04	02	17	45°
4	0.913	15472	–	–	4.77	16	12	23	22.5°
		13296	–	–		16	04	23	45°
5	0.913	1296	–	–	4.77	32	22	57	22.5°
		528	–	–		26	04	53	45°
6	1.066	48480	–	–	5.44	004	030	127	22.5°
		17088	–	–		060	004	127	45°

$$\gamma_8 = -2.17\ dB, \quad \gamma_F = 0.06\ dB$$

11(b) $R = 6/7$, $R_{\text{eff}} = 3.0$ bit/T, $d_u^2 = 0.586$

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^2	h^1	h^0	Transparency $(2^d\Psi)$
1	0.890	144	–	–	1.82	–	2	3	22.5°
2	1.172	9	1.476	576	3.01	–	2	5	22.5°
3	1.476	324	–	–	4.01	04	02	17	90°
4	1.757	27	–	–	4.77	14	06	23	45°
5	1.781	432	–	–	4.83	06	16	53	45°
6	2.0	6	2.085	11988	5.33	022	052	133	45°
			2.062	162		046	014	103	90°

$$\gamma_8 = 0\ dB, \quad \gamma_F = 0\ dB$$

Table 12: 6-D (L=3) Trellis Coded 16PSK

12(a) $R = 11/12$, $R_{\text{eff}} = 3.67$ bit/T, $d_u^2 = 0.304$

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^3	h^2	h^1	h^0	Transparency ($2^d\Psi$)
1	0.304	40	0.457	8320	0.00	–	–	2	3	22.5° (I)
2	0.457	2304	–	–	1.76	–	6	4	5	22.5° (I)
3	0.586	45	0.609	1600	2.84	–	04	02	11	22.5° (I)
4	0.609	304	–	–	3.01	14	04	02	21	45° (I)
		208	–	–		10	06	04	21	90° (I)
5	0.761	30720	–	–	3.98	14	24	02	77	45° (I)
*6	0.890	180	–	–	4.66	050	024	006	103	45° (I)

$\gamma_8 = -1.97\ dB$, $\gamma_F = 1.04\ dB$

12(b) $R = 10/11$, $R_{\text{eff}} = 3.33$ bit/T, $d_u^2 = 0.457$

v	d^2_{free}	$N(d^2_{\text{free}})$ $\leq$	d^2_{next}	$N(d^2_{\text{next}})$ $\leq$	γ (dB)	h^3	h^2	h^1	h^0	Transparency ($2^d\Psi$)
1	0.586	45	0.609	1168	1.08	–	–	2	3	45° (II)
2	0.738	180	–	–	2.08	–	6	4	7	45° (II)
3	1.043	1125	–	–	3.58	–	04	02	11	90° (II)
4	1.172	57	1.195	4500	4.09	–	14	02	33	45° (II)
5	1.172	45	–	–	4.09	34	16	06	41	22.5° (III)
*6	1.218	920	–	–	4.26	010	046	060	105	22.5° (III)

$\gamma_8 = -0.62\ dB$, $\gamma_F = 0.84\ dB$

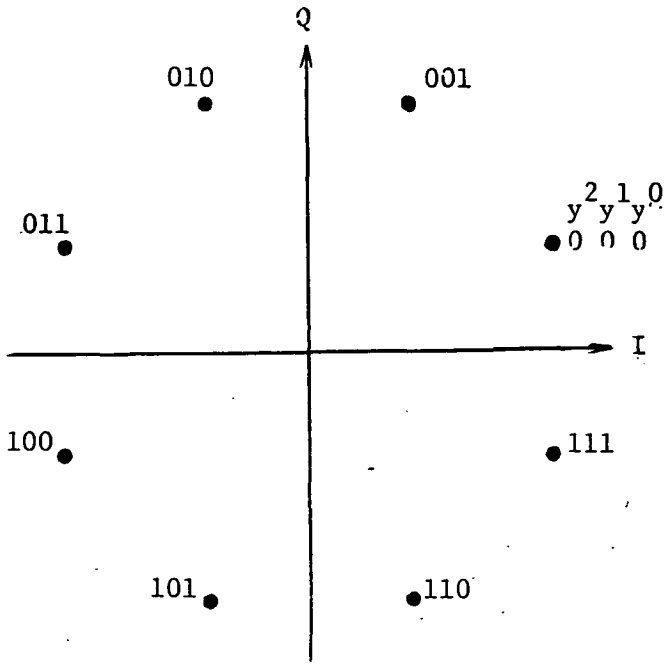


Fig. 1. 2-Dimensional 8PSK Signal Set with natural mapping.

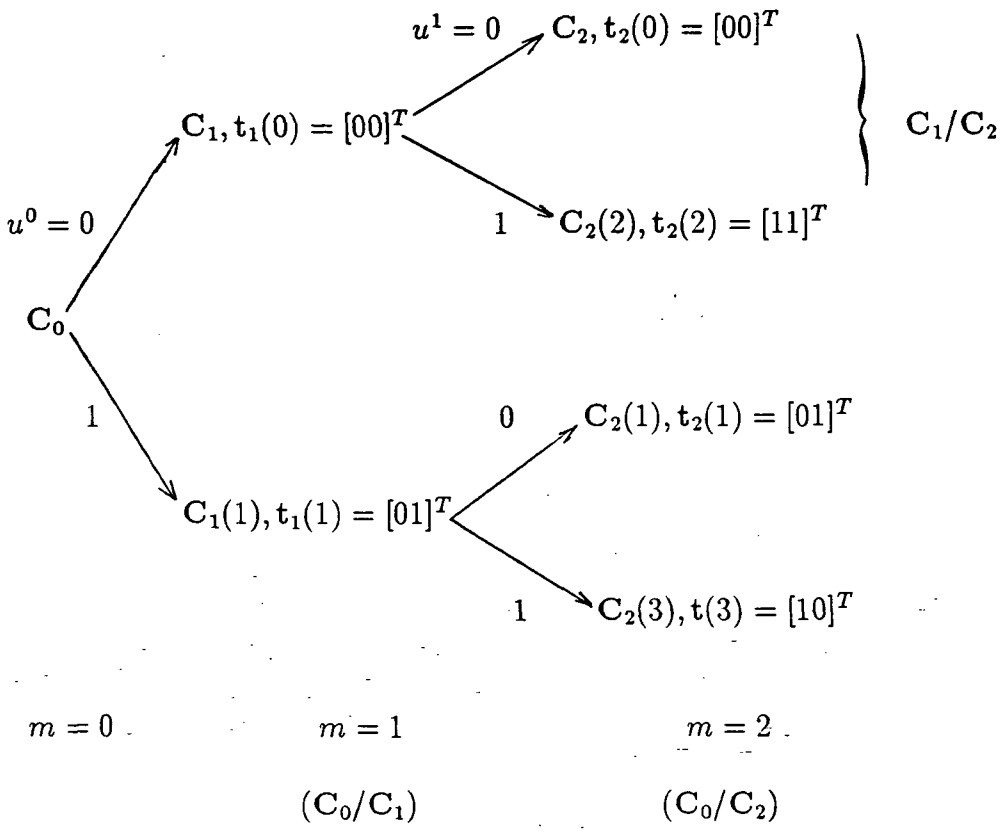
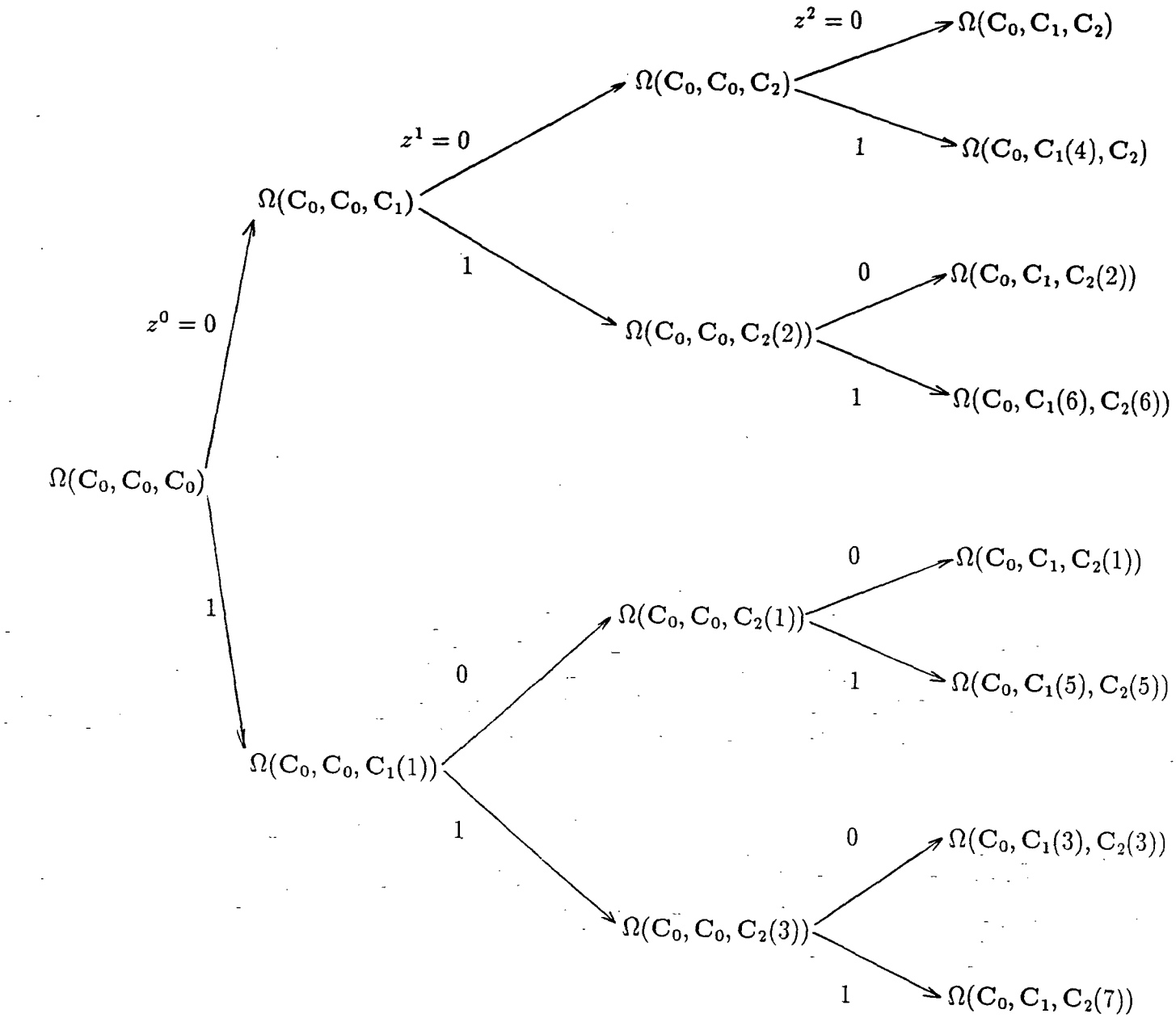
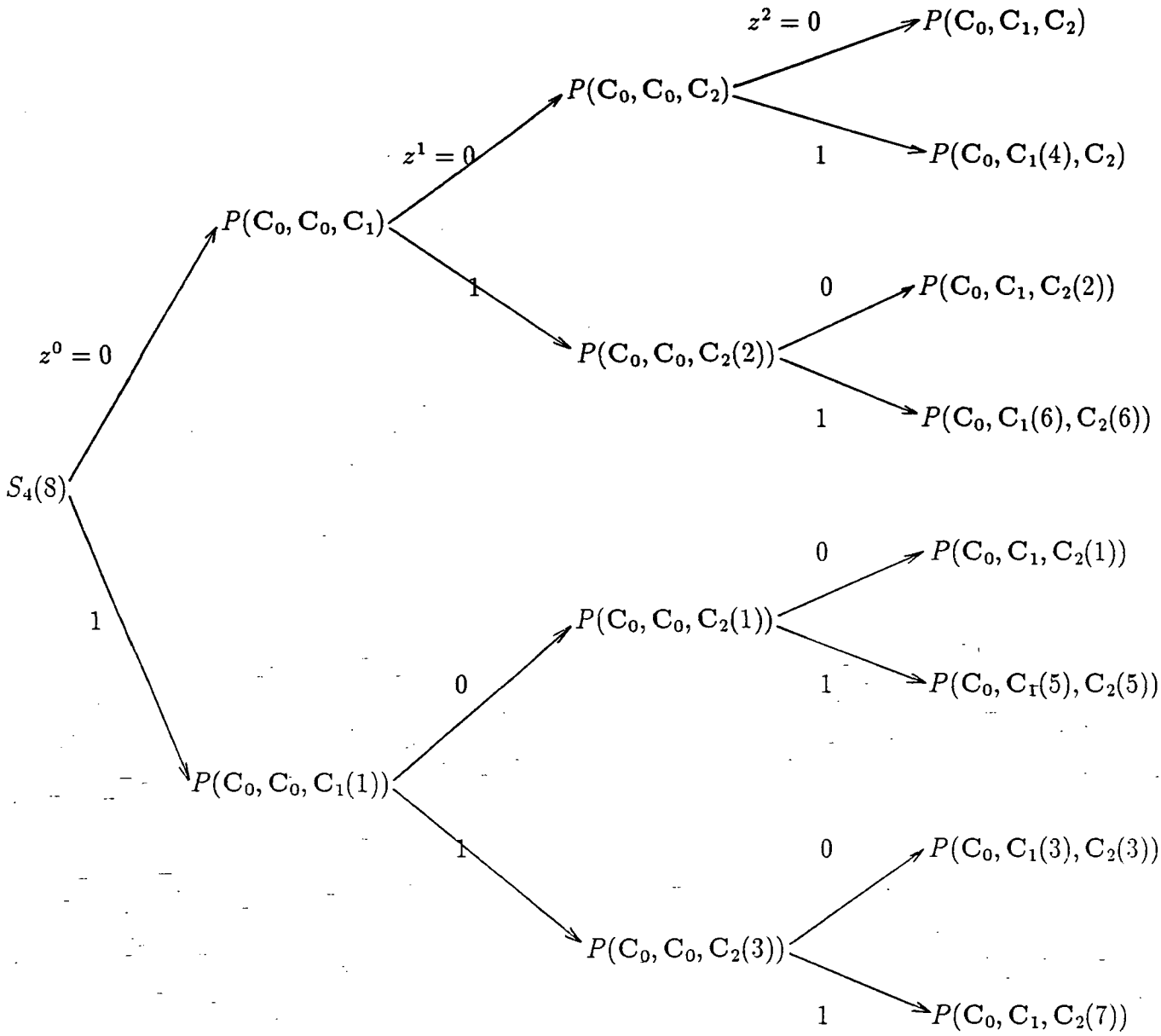


Figure 2: A 2/2-way partition chain $\mathbf{C}_0/\mathbf{C}_1/\mathbf{C}_2$ in the 2-D binary vector space.



Note: for $p = 3$ $m_2 = 0$; $C_{m_2}(z) = C_0$,
$m_1 = 1$; $C_{m_1}(z) = C_1 \oplus [0, z^2 \oplus z_1^0 \cdot z^1]^T$,
$m_0 = 2$; $C_{m_0}(z) = C_2 \oplus [z^1, z^0 \oplus z^1]^T$.

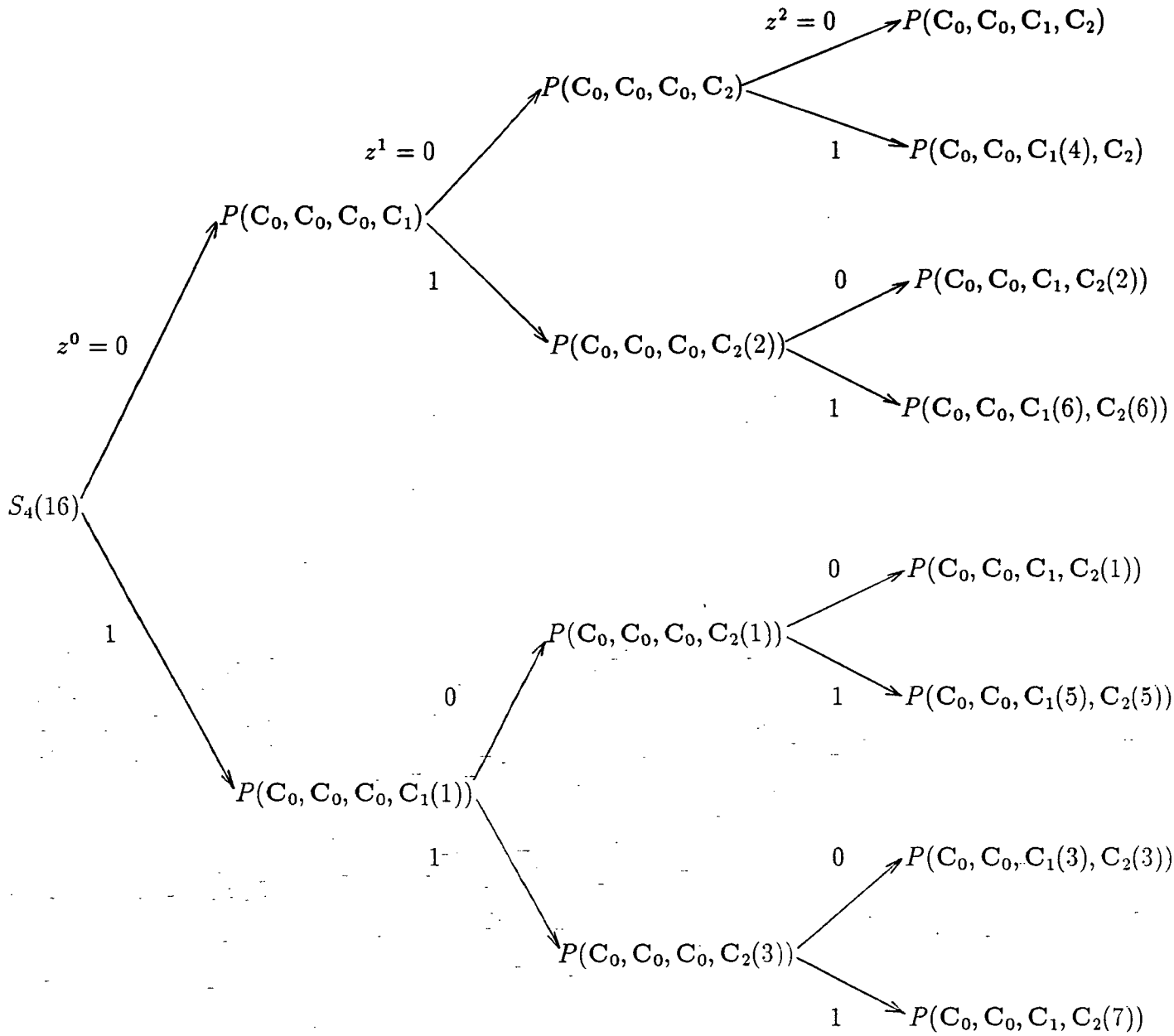
Figure 3: A 2/2/2 partition chain $\Omega(C_0, C_0, C_0)/\Omega(C_0, C_0, C_1)/\Omega(C_0, C_0, C_2)/\Omega(C_0, C_1, C_2)$ in the 2×3 binary matrix space.



	$p=0$	$p=1$	$p=2$	$p=3$
Δ_p^2	0.586	1.172	2	4

Note: At $p=3$ $\quad m_2 = 0;\ \mathbf{C}_{m_2}(z) = \mathbf{C}$,
$m_1 = 1;\ \mathbf{C}_{m_1}(z) = \mathbf{C}_1 \oplus [0, z^2 \oplus z^0 \cdot z^1]^T$,
$m_0 = 2;\ \mathbf{C}_{m_0}(z) = \mathbf{C}_2 \oplus [z^1, z^0 \oplus z^1]^T$.

Figure 4: A three level 4-D 8PSK signal set partition.



p	0	1	2	3
Δ_p^2	0.152	0.304	.586	1.172

Note: For $p = 3$ $m_3 = 0$; $C_{m_3}(z) = C_0$
$m_2 = 0$; $C_{m_2}(z) = C_0$,
$m_1 = 1$; $C_{m_1}(z) = C_1 \oplus [0, z^2 \oplus z^0 \cdot z^1]^T$,
$m_0 = 2$; $C_{m_0}(z) = C_2 \oplus [z^1, z^0 \oplus z^1]^T$.

Figure 5: A three level 4-D 16PSK signal set partition.

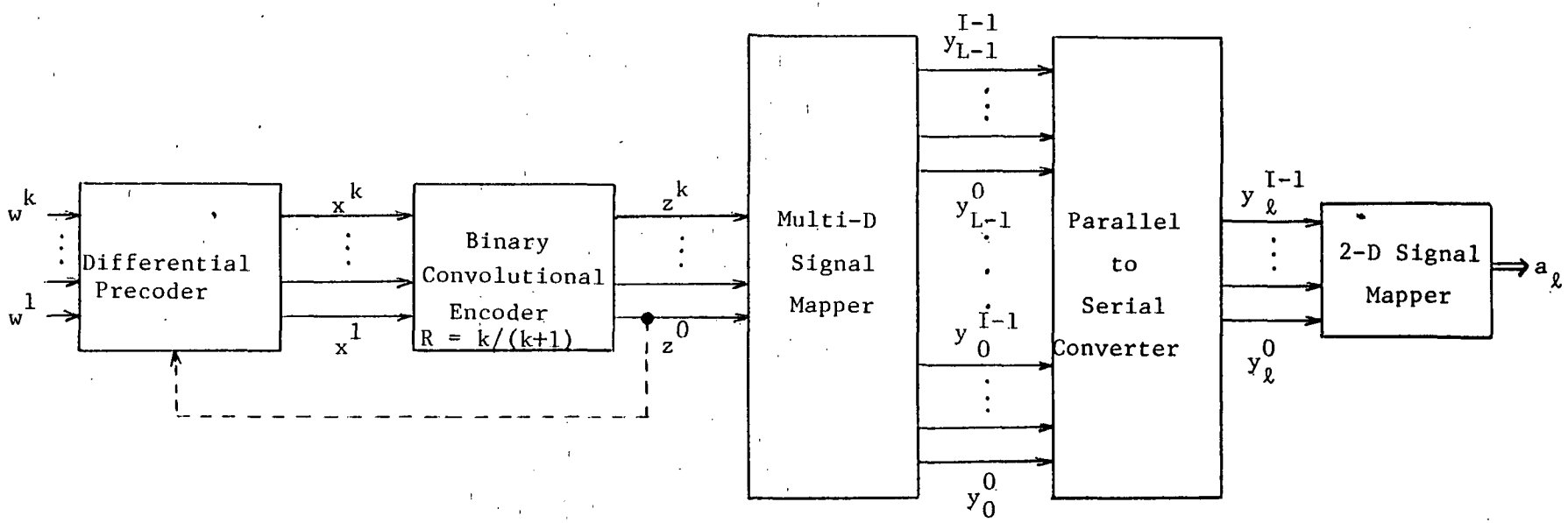


Figure 6: General encoder system for coding with multi-D signal sets.

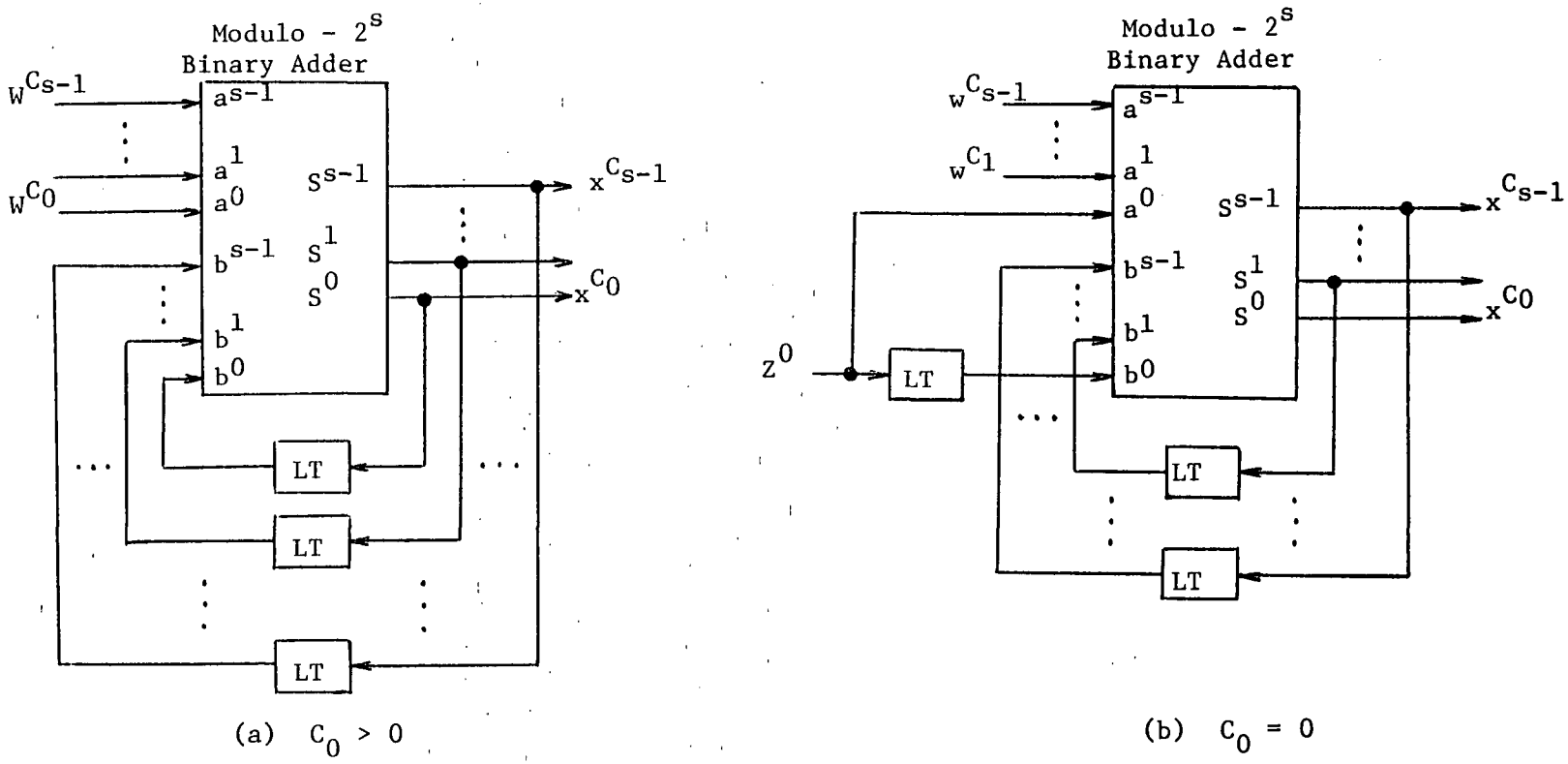


Figure 7: Differential precoders for general encoder.

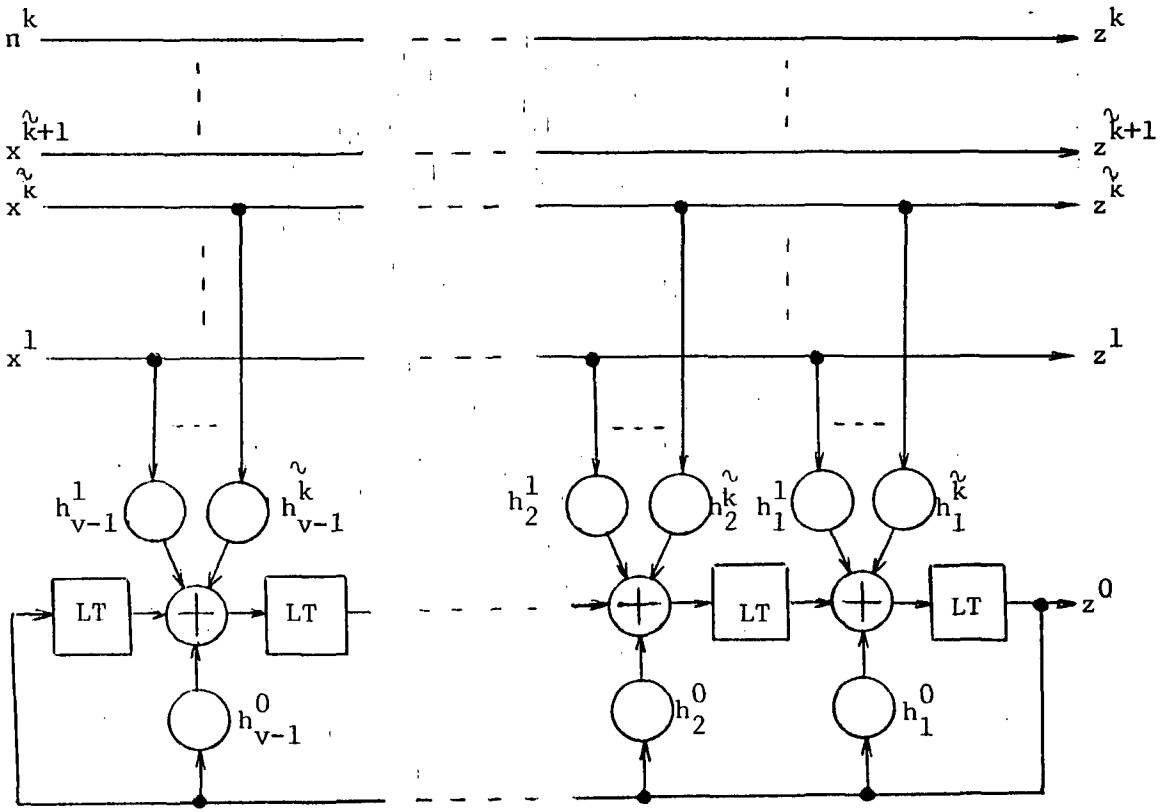
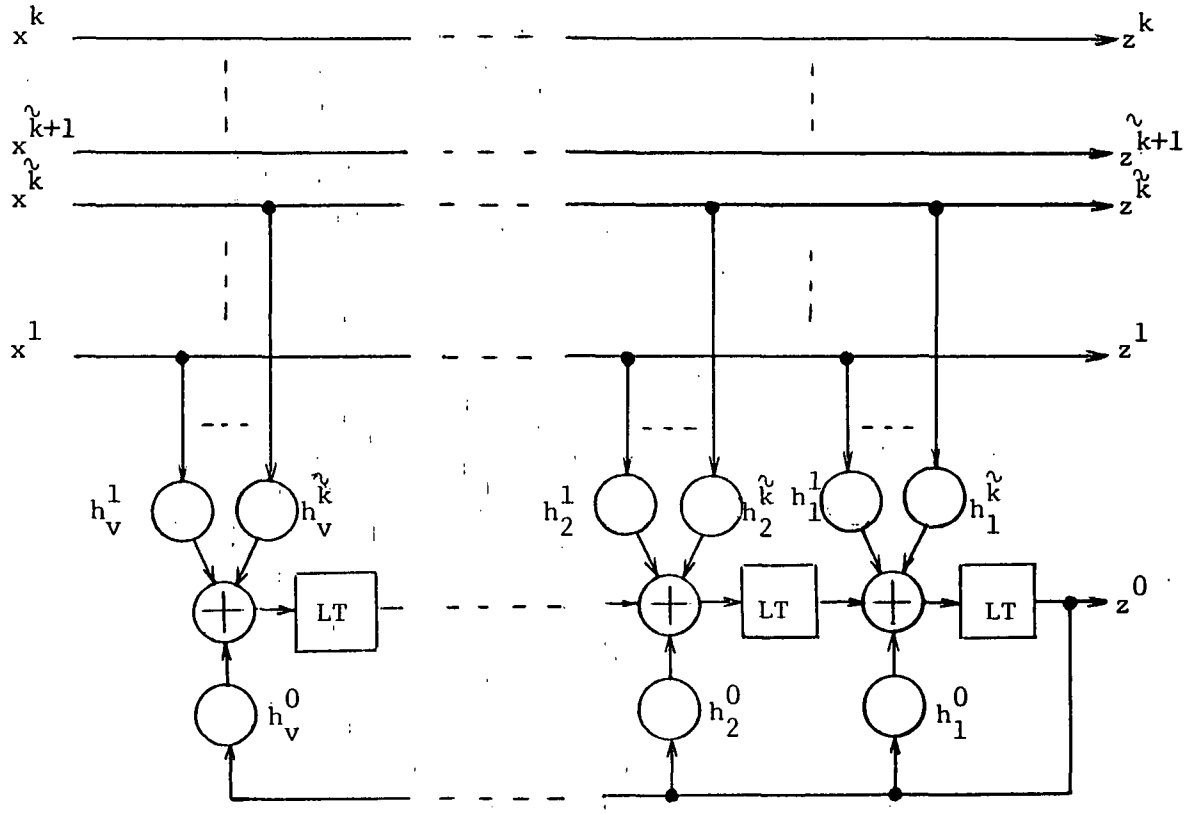


Figure 8(a): Systematic convolutional encoder for $\tilde{k}$ checked bits and $\tilde{k} < v$.



Firgure 8(b): Systematic convolutional encoder with $\tilde{k}$ checked bits and $\tilde{k} = v$.

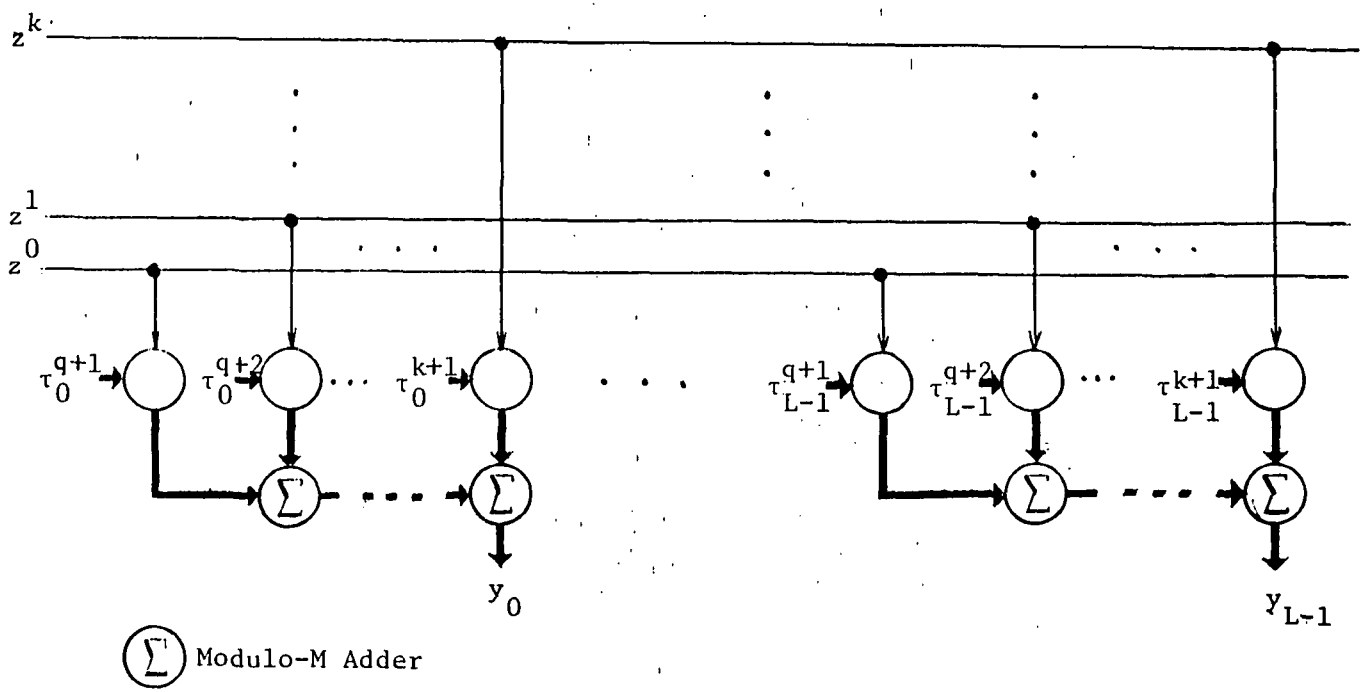


Figure 9: Multi-D Signal mapper.

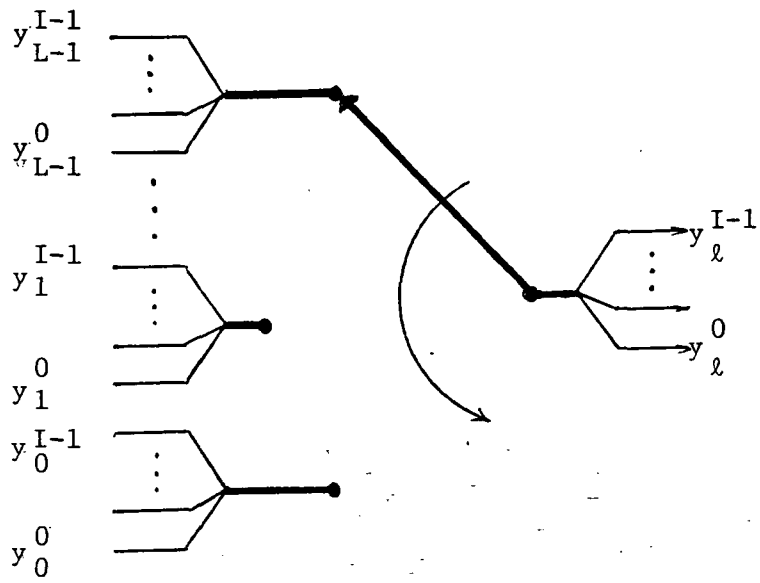


Figure 10: Parallel to serial converter.

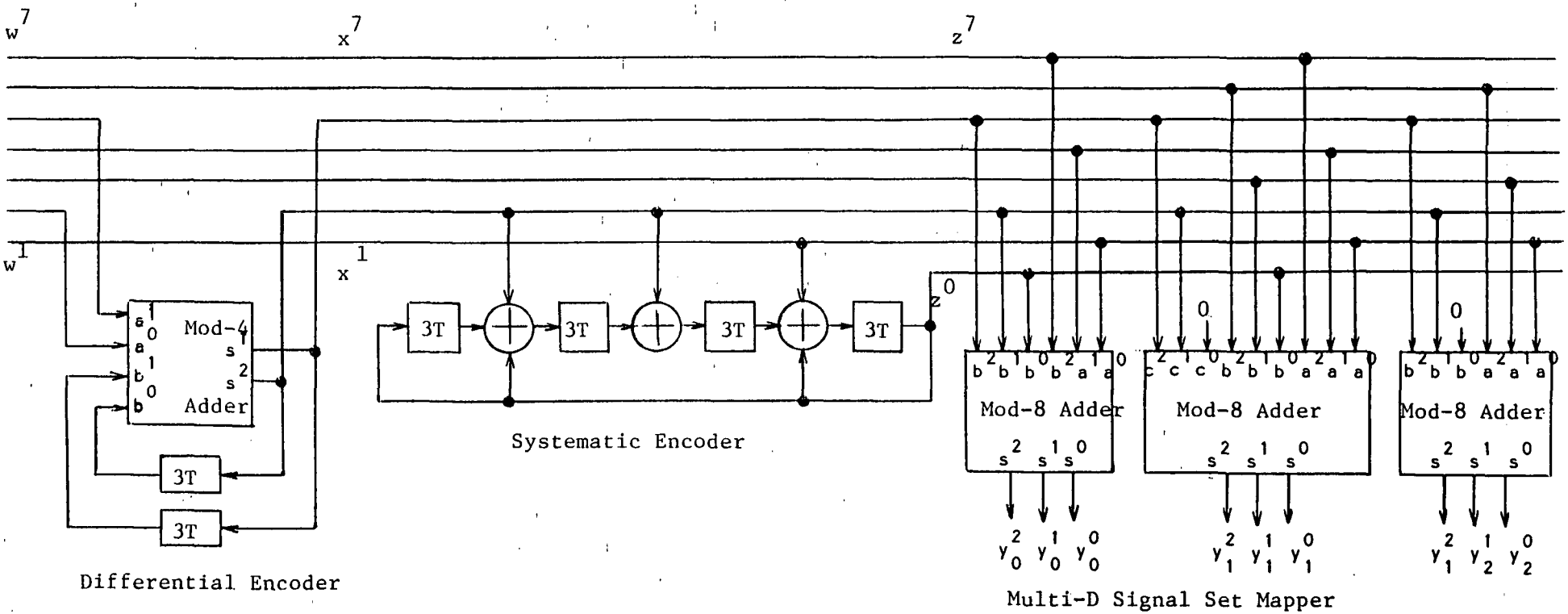


Figure 11: Encoder System for a rate 7/8, 6-D 8PSK signal set (2·33 bit/T) and 90^{o} transparent code with 16 states and $\tilde{k} = 2$.

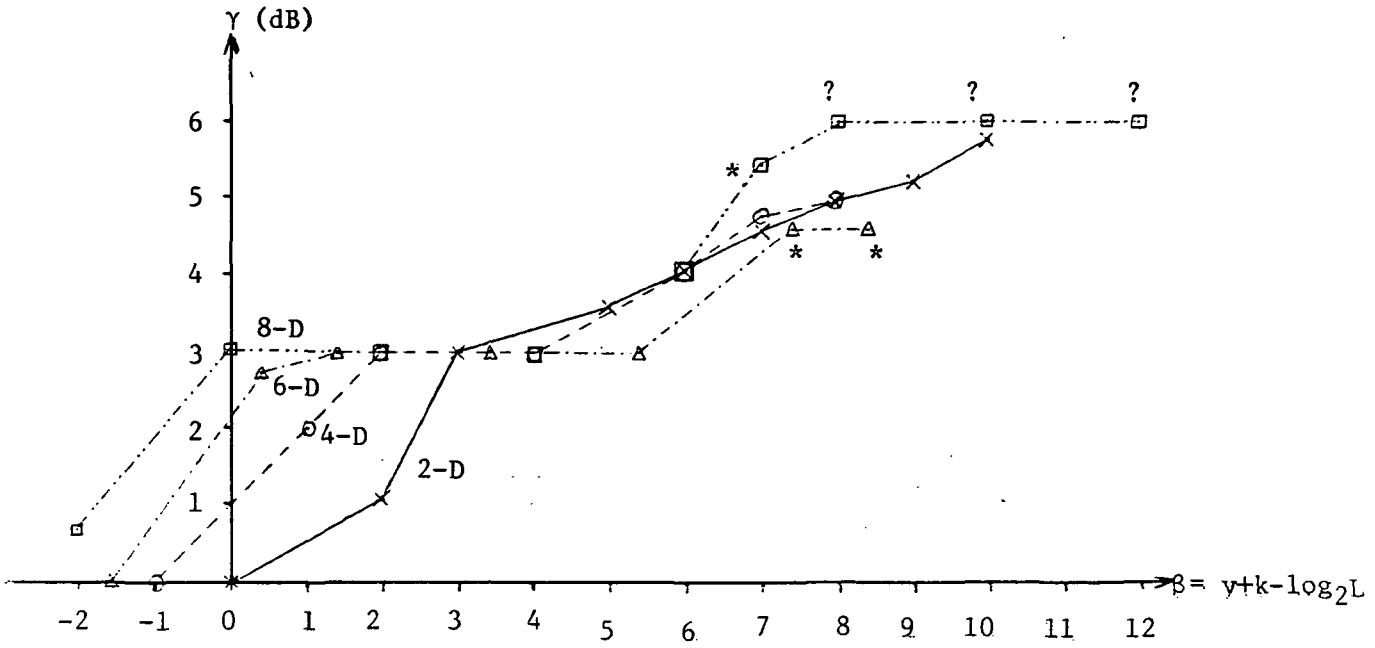


Figure 12(a): Coding gain verses complexity for multi-D 8PSK, and 2-D bit/T.

* Code search incomplete

? Predicted codes

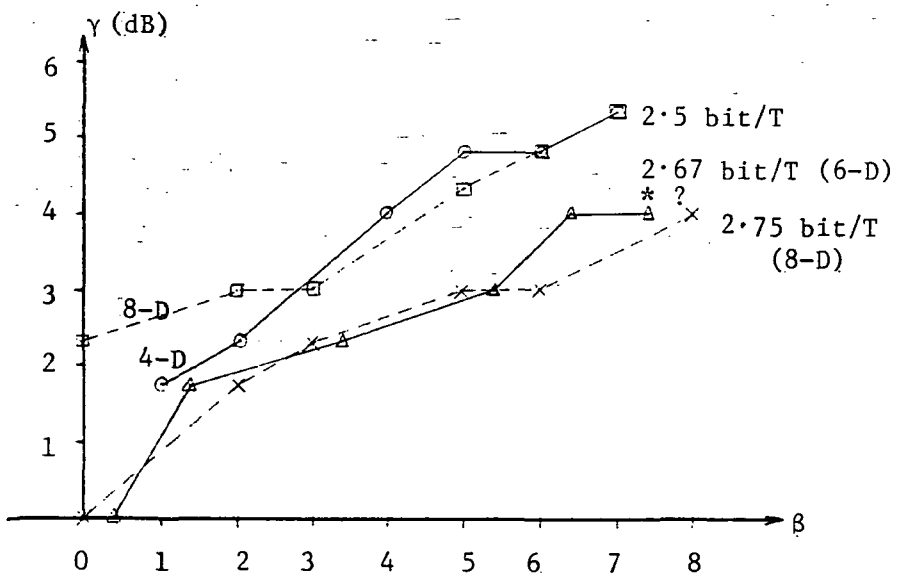


Figure 12(b): Coding gain verses complexity for multi-D 8PSK.

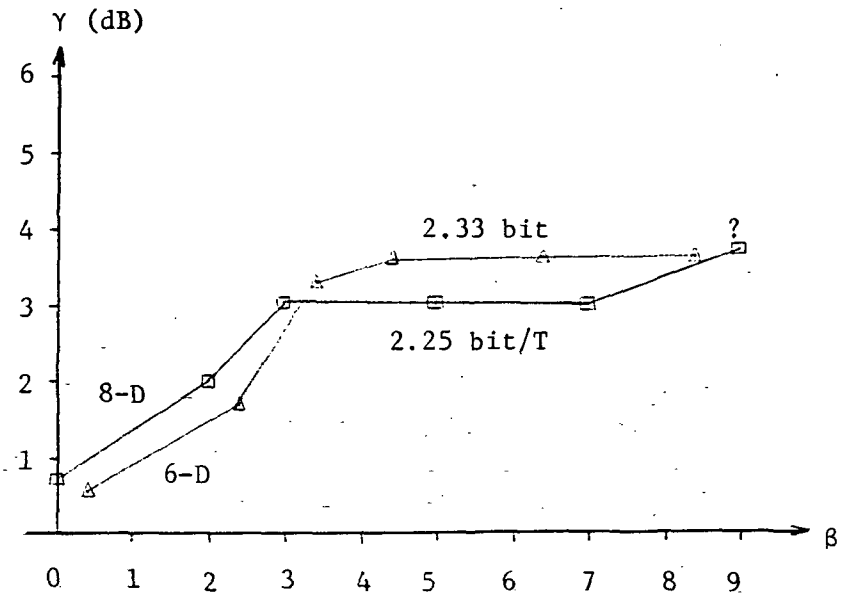


Figure 12(c): Coding gain verses complexity for multi-D 8PSK.

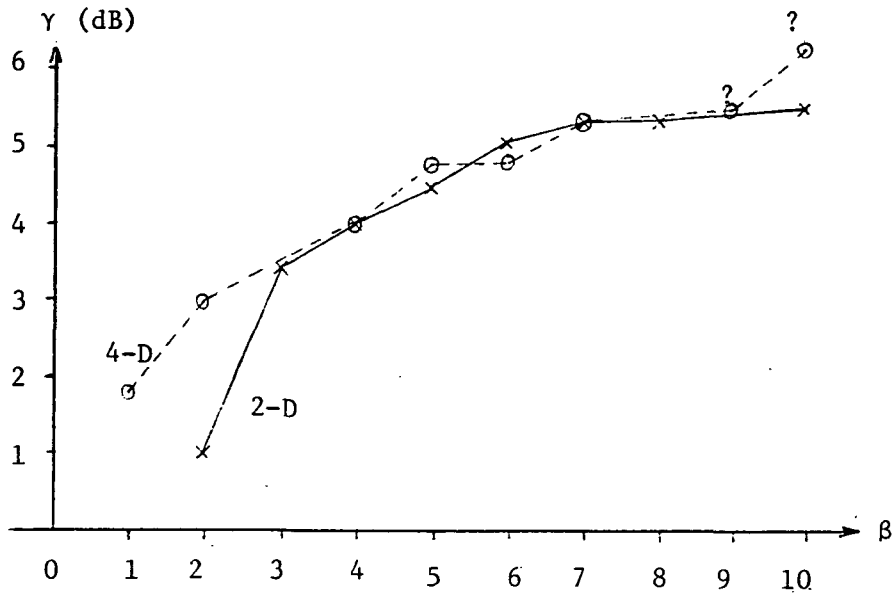
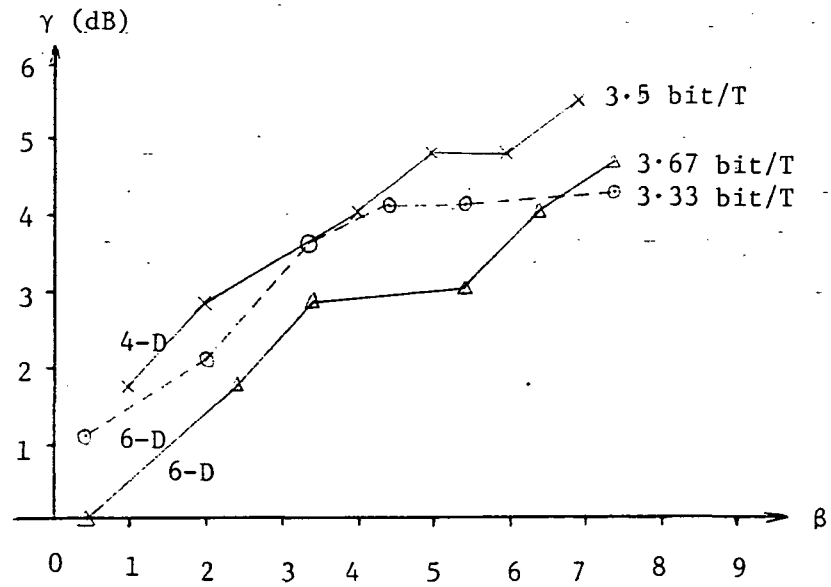


Figure 13(a): Coding gain verses complexity for multi-D 16PSK and 3-D bit/T.



Appendix B

An Algorithm for Computing the Distance Spectrum of Trellis Codes

An Algorithm For Computing the Distance Spectrum of Trellis Codes*

Marc Rouanne and Daniel J. Costello, Jr.
Department of Electrical and Computer Engineering,
University of Notre Dame, Notre Dame, IN 46556

May 1, 1988

Submitted to the IEEE Journal on Selected Areas in Communications

Abstract

The performance of a trellis code can be accurately predicted from its distance spectrum. A class of quasi-regular codes is defined for which the distance spectrum can be calculated from the codeword corresponding to the all-zero information sequence. An algorithm to compute the distance spectrum of linear, regular, and quasi-regular trellis codes is presented. In particular, it can calculate the weight spectrum of convolutional (linear trellis) codes and the distance spectrum of most of the best known trellis codes. The codes do not have to be linear or regular. The algorithm is a bidirectional stack algorithm. We use the algorithm to calculate the beginning of the distance spectrum of some of the best known trellis codes and to compute tight estimates on the first event error probability and on the bit error probability.

1 Introduction

The performance of a trellis code depends on the decoding algorithm employed and on the distance properties of the code, i.e., the distances between codewords. The exact error probability of a coded system cannot be calculated, even for simple trellis codes. However, trellis code error probabilities can be estimated using simulations and performance bounds. Simulations often require long running times and are only useful for short constraint length codes. Performance bounds are the most common means of estimating the error probability

*This work was supported by NASA Grant NAG5-557. Part of this material was presented at the Conference on Information Sciences and Systems, Princeton, NJ, March 1988.

of codes and of designing new coding schemes. The distance spectrum can be used to compute performance bounds.

A *trellis code*, or *trellis coded modulation* (TCM), consists of a convolutional encoder followed by a mapper. Figure 1 shows a typical trellis code as originally designed by Ungerboeck [10]. This schematic representation was introduced by Forney et. al. [6]. An encoder state is characterized by the values of the past information bits stored in the shift registers of the convolutional encoder. The incoming information bits determine the transitions or branches connecting one state to the other. During each signaling interval τ, the $\tilde{k}$ information bits $u_\tau^1, \ldots, u_\tau^{\tilde{k}}$ enter the convolutional encoder and an n-bit subset selector $v_\tau = (v_\tau^1, \ldots, v_\tau^n)$ leaves the encoder. Subset selectors depend on the incoming $\tilde{k}$ information bits and on ν past information bits only, where ν is called the constraint length of the trellis code. The mapper transforms the subset selector into a subset of channel signals. The uncoded information bits $u_{\tilde{k}+1}, \ldots, u_k$ then select one particular channel signal from the selected subset. The set of possible channel signals is denoted by S.

A *topological trellis* is a trellis with no labels on the branches. A *topological path* $\mathbf{Y} \triangleq \{\ldots, Y_\tau, Y_{\tau+1}, \ldots\}$ through a topological trellis is a sequence of consecutive branches Y_τ which have not yet been assigned a signal. The *topological branch* Y_τ is the τ^{th} branch in $\mathbf{Y}$ and corresponds to the τ^{th} signaling interval since the beginning of $\mathbf{Y}$. A *channel path* $\mathbf{y}$ is defined by a topological path $\mathbf{Y}$ and a sequence of labels: $\mathbf{y} \triangleq \{\ldots, y_\tau, y_{\tau+1}, \ldots\}$, where y_τ is a branch in $\mathbf{Y}$ labeled with a channel signal in S. A channel path is a path through a labeled trellis. The *length* l of a path is the number of consecutive branches that form the path, and it can be finite or infinite. We will sometimes call y_τ a signal although it is more properly called a labeled branch, and we say that y_τ is the output signal during the τ^{th} signaling interval. The context should make clear when we mean a labeled branch or a signal.

A *labeling* of a topological trellis associates a signal with each branch in the trellis. A trellis labeling can be seen as the combination of a binary encoder and a mapper. A *trellis code* is uniquely characterized by a topological trellis and a labeling. The most general way to define a trellis code is by using a table which assigns a signal to each branch in the trellis. In other words, a trellis code is the set of all labeled trellis branches for all signaling intervals (to construct a trellis code, one only needs to label trellis branches with channel signals).

Given two signal or subset sequences $\mathbf{y} = (\ldots, y_1, y_2, \ldots, y_l, \cdots)$ and $\hat{\mathbf{y}} = (\ldots, \hat{y}_1, \hat{y}_2, \ldots, \hat{y}_l, \cdots)$, $(\mathbf{y}, \hat{\mathbf{y}})$ is a *first event error* of length l if $Y_\tau = \hat{Y}_\tau$ for $\tau < 1, Y_\tau \neq \hat{Y}_\tau$ for $1 \leq \tau \leq l$,

and $Y_\tau = \hat{Y}_\tau$ for $\tau > l$, i.e., the error event starts when the two paths diverge and ends when the two paths remerge for the first time. Figure 2 shows a first event error of length l (the correct and incorrect paths remerge after l branches).

The performance of a trellis code depends on the distribution of distances between encoder output sequences (code words) corresponding to distinct encoder input sequences. In particular, if $\mathbf{y} = \{y_1, \ldots, y_\tau, \ldots\}$ and $\hat{\mathbf{y}} = \{\hat{y}_1, \ldots, \hat{y}_\tau, \ldots\}$ are sequences of signals from S, the squared Euclidean distance $d(\mathbf{y}, \hat{\mathbf{y}})$ satisfies

$$d(\mathbf{y}, \hat{\mathbf{y}}) = \sum_{\tau=1}^{\infty} d(y_\tau, \hat{y}_\tau), \tag{1}$$

where $d(y_\tau, \hat{y}_\tau)$ is the squared Euclidean distance between two channel signals y_τ and $\hat{y}_\tau$. The distance between two code words determines the likelihood of decoding one code word when the other one was sent. For an AWGN channel, the squared Euclidean distance between two signal sequences $\mathbf{y}$ and $\hat{\mathbf{y}}$ determines the likelihood of receiving $\hat{\mathbf{y}}$ given that $\mathbf{y}$ was sent.

A union bound on the *first event error probability* P_e of trellis codes may be obtained by summing the error probability over all possible incorrect paths which remerge with all possible correct paths [12]. At any time unit, P_e is bounded by

$$P_e \leq \sum_{d=d_{free}}^{\infty} A_d Q\left(\frac{d}{\sqrt{2N_0}}\right), \tag{2}$$

where d represents the squared Euclidean distance between signal sequences, A_d is the average number (multiplicity) of code words at distance d from a specific code word, where the average is taken over all code words in the code, d_{free} is the minimum free squared Euclidean distance of the code, N_0 is the one-sided noise power spectral density, and $Q(.)$ is the Gaussian integral function $Q(\beta) \triangleq \int_\beta^\infty e^{-x^2/2} \frac{dx}{\sqrt{2\pi}}$. Equation (2) can be rewritten as

$$P_e < \sum_{d=d_{free}}^{\infty} A_d P_d,$$

where $P_d \triangleq Q(d/\sqrt{2N_0})$ is the two code word error probability for distance d.

The *bit error probability* P_b is the average number of bit errors per decoded information bit. Equation (2) can be modified to provide a bound on P_b by weighting each term P_d by the average number B_d of information bits on all paths at distance d from the correct path [12]. Hence, at any time unit, P_b is bounded by

$$P_b < \sum_{d=d_{free}}^{\infty} B_d P_d. \tag{3}$$

A *spectral line* is defined by a distance d and its average multiplicity A_d. The set of all spectral lines is called the *distance spectrum* of the code. If the code is linear, A_d is the number of code vectors of weight d in the code, and the distance spectrum is commonly called the weight spectrum of the code [8].

The first event error probability can be estimated in terms of the free distance d_{free} of the code. Trellis codes with a large free distance are optimum at large signal-to-noise ratios (SNR). At moderate SNR, the optimality of the code depends on the first few spectral lines of the code, especially for non-binary, non-regular trellis codes whose distance spectra are relatively dense. This means that the codes with the best free distance may not be the codes that perform the best for moderate SNR.

2 Quasi-Regularity

Given a signal set S and an equivalence relation (R) defined between elements of S, two elements x and y in S belong to the same equivalence class of (R) if and only if they satisfy (R), which is denoted xRy. An *equivalence sequence* $(R_1)/(R_2)/\ldots/(R_r)$ on S is a set of r equivalence relations defined on elements of S which satisfy

$$\forall \quad i,j, \ 1 \leq i \leq j \leq r, \ \forall \quad x,y \epsilon S, \quad xR_jy \Rightarrow xR_iy, \tag{4}$$

where r is the number of *levels* in the sequence. The equivalence classes generated by (R_i) are called *subsets of S at level i and form a partition chain* of the signal set.

A trellis code is *linear* for an operation called a sum iff the sum of any two codewords is a codeword. For example, convolutional codes are linear because the modulo-two sum of any two codewords gives a binary codeword [8]. Linearity can be defined with respect to an equivalence relationship defined on signals [5]. The equivalence classes defined by a relation (R) partition the signal set into subsets. Let a sum be defined on these subsets and a trellis code be labeled with subsets. The codewords of such a code are sequences of subsets, and the code is *linear with respect to* (R) iff the sum of any two codewords (sequences of subsets) is another codeword (sequence of subsets).

When the trellis is labeled with subsets, the sum of signals need not be defined and linearity with respect to (R) is less stringent than linearity with respect to signals. When the signal set is given the structure of a group and partitioned into cosets, linearity with respect to cosets is the same as linearity with respect to signals, because then the sum is defined from the group structure. This can be shown as follows. Suppose that a code is linear with respect to the cosets of a subgroup of the signal set. Consider two sequences of signals through the trellis: they define two sequences of cosets whose sum is a codeword because the code is linear with respect to cosets. The sum of the two sequences of signals corresponds to that codeword and must be a channel path through the trellis, which proves that the code is linear with respect to signals.

A trellis code is *regular* iff the distance between two codewords that correspond to distinct information sequences depends only on the binary sum of the two information sequences (we assume that the distance is an additive metric). Once again, codewords can be sequences of subsets and regularity can be defined with respect to an equivalence relation [3]. In such a case, the distance between subsets is the minimum distance between the signals in the subsets.

Regularity makes the calculation of the distance properties of a code feasible. For regular codes, the set of distances of incorrect paths from a correct path does not depend on the correct path. This means that the distance properties of regular codes can be calculated by assuming that an arbitrary correct path was sent. In practice, it is assumed that the path generated by the all-zero information sequence was sent. This assumption considerably reduces the complexity of distance spectrum calculations, since only one among many correct paths must be evaluated. Unlike linearity, regularity with respect to cosets is not equivalent to regularity with respect to signals. There are no known bandwidth efficient codes that are regular with respect to signals, and it can be conjectured that none exist. However, there exist many known bandwidth efficient codes that are regular with respect to cosets [3] and [5].

Ungerboeck was the first to show that for certain non-regular codes the free distance can be calculated by assuming that the all-zero information sequence was sent [10]. This can be generalized to any trellis code, regular or not, but leads to far more complex algorithms than the one presented in the next section. Instead, we define the class of quasi-regular codes to be non-regular codes for which the distance spectrum can be calculated with a relatively simple algorithm by assuming that the all-zero information sequence was sent [9].

A mapping of signal selectors onto a signal set is *regular* iff the distance between two signals depends only on the Hamming distance between their signal selectors. For example, there is no regular mapping of eight signal selectors onto an 8-PSK signal set. Similarly, there is no regular mapping from four signal selectors onto 4-PAM or from 16 signal selectors onto 16-QASK. Figure 3 shows a non-regular mapping onto 8-PSK. This particular mapping is known as the natural mapping. The regularity of a mapping can also be defined with respect to subsets.

Let s and $\hat{s}$ be two states in a trellis code, and a *signal selector error vector* e the binary sum of two signal selectors. Then the distance polynomial $P_{s,\hat{s},e}(x)$ represents the set of distances between signals generated from s and $\hat{s}$ and whose signal selectors differ by e. The polynomials $P_{s,\hat{s},e}(x)$ are only defined for those e for which there exists a branch that leaves s and a branch that leaves $\hat{s}$ whose signal selectors differ by e. Let $y(\mathrm{v})$ be the signal in S whose signal selector is the binary n-tuple v. Then the polynomial $P_{s,\hat{s},e}(x)$ is given by

$$P_{s,\hat{s},e}(x) \triangleq \sum_{v|s} p(\mathrm{v}|s) x^{d[y(v),y(v+e)]},$$

where $p(v|s)$ is the probability of the signal selector v given that the encoder is in state

s and $d[y(v), y(v+e)]$ is the squared Euclidean distance between $y(v)$ and $y(v+e)$. For example, for 8-PSK, if $y(000)$, $y(010)$, $y(100)$, $y(110)$ leave state s and $y(001)$, $y(011)$, $y(101)$, $y(111)$ leave state $\hat{s}$, only four error vectors e are possible between the branches that leave s and $\hat{s}$ (Figure 3). These four error vectors are 001, 011, 101, and 111, and the corresponding distance polynomials satisfy

$$\begin{aligned} P_{s,\hat{s},001}(x) &= x^{\delta_0}, \\ P_{s,\hat{s},011}(x) &= 1/2x^{\delta_0} + 1/2x^{\delta_2}, \\ P_{s,\hat{s},101}(x) &= x^{\delta_2}, \\ P_{s,\hat{s},111}(x) &= 1/2x^{\delta_0} + 1/2x^{\delta_2}. \end{aligned}$$

These polynomials look similar to the "weight profiles" defined by Zehavi and Wolf in a paper on the performance of rate $k/(k+1)$ trellis codes mapped by set partitioning [14]. However, weight profile polynomials are defined from a knowledge of the signal set and the mapper only, whereas the above polynomials are code dependent. This allows the definition to apply to a large class of codes of any rate which includes the codes treated by Zehavi and Wolf.

A trellis code is *quasi-regular* iff (i) it consists of a linear binary encoder followed by a mapper and (ii) for all e and all pairs of states $(s_1, \hat{s}_1)$ and $(s_2, \hat{s}_2)$, $P_{s_1,\hat{s}_1,e}(x) = P_{s_2,\hat{s}_2,e}(x)$ (provided that the two polynomials are defined). By definition, for regular codes, the distance between signals depends only on the binary sum e of their signal selectors ($P_{s,\hat{s},e}(x)$ is a monomial which does not depend on s or $\hat{s}$), and regular codes are quasi-regular. Since linear codes are regular, they are also quasi-regular. In the previous example of trellis coded 8-PSK, the two polynomials $P_{s,\hat{s},011}(x)$ and $P_{s,\hat{s},111}(x)$ are not monomials, and the code cannot be regular. However, it is quasi-regular because $P_{s,\hat{s},e}(x)$ does not depend on $(s, \hat{s})$.

Let V be the set of signal selectors generated by the underlying binary code and $y(\mathrm{v})$ be the signal in S selected by some $\mathrm{v} \in V$. To each signal selector error e corresponds a unique set of distances $\{d[y(\mathrm{v}), y(\mathrm{v} \oplus e)], v \epsilon V\}$. We define $w(e)$ as the minimum distance of this set. Originally, Ungerboeck [10] computed the free distance of his codes by assuming that the all-zero information sequence was sent and replacing $d[y(\mathrm{v}), y(\mathrm{v} \oplus e)]$ with a lower bound on $w(e)$ in the computation. In a more recent publication, the free distance was computed directly from the values of $w(e)$ [11]. Similarly, it will be shown later that the distance spectrum of quasi-regular codes can be computed from the all-zero path by using the $w(e)$'s.

The set of signal selector error vectors e for which there exists a v such that $d[y(\mathrm{v}), y(\mathrm{v} \oplus e)] > w(e)$ is denoted E. For example, for the mapping in Figure 3, $E = \{011, 111\}$, which means that the distance between signals whose selectors differ by 011 or 111 is not unique.

A distance spectrum contains all the distances between codewords, even infinite distances between codewords that never remerge. In order to avoid dealing with these infinite distances, we consider only paths of any length l for any finite $l \geq 0$. The distance spectrum $SP^{(l)}(x)$ *at depth* l of a trellis code satisfies

$$SP^{(l)}(x) \triangleq \sum_{\mathbf{y}} p(\mathbf{y}) \sum_{\hat{\mathbf{y}}} x^{d(\mathbf{y},\hat{\mathbf{y}})}, \tag{5}$$

where $\mathbf{y} = \{y_1, \ldots, y_l\}$ is a correct path of length $l, \hat{\mathbf{y}} = \{\hat{y}_1, \ldots, \hat{y}_l\}$ is an incorrect path diverging from $\mathbf{y}$ at time 1 such that $(\mathbf{y}, \hat{\mathbf{y}})$ is a first event error of length l, and $p(\mathbf{y})$ is the probability of the correct path $\mathbf{y}$. $SP^{(l)}(x)$ represents all possible distances between paths of length l which leave the same state at time 1. The distance spectrum is entirely defined by $SP^{(l)}(x)$ for all l.

The *worst case distance spectrum* of a trellis code is derived from the distance spectrum of the code by replacing the distance $d[y(\mathrm{v}), y(\hat{\mathrm{v}})]$ by $w(\mathrm{v} \oplus \hat{\mathrm{v}})$ for all pairs of signals $(y(\mathrm{v}), y(\hat{\mathrm{v}}))$. Two paths have the same worst case distance if they have the same distance calculated using the $w(e)$'s and if they have the same number of occurrences of e for each $e \in E$. This means that the worst case distance spectrum contains some topological information about paths.

Most of the best known trellis codes consist of a binary linear convolutional encoder followed by a mapper. In this case, the worst case distance spectrum has a simple expression.

Lemma 1: The worst case distance spectrum at depth l of a trellis code which consists of a binary linear convolutional encoder followed by a mapper satisfies:

$$SP_w^{(l)}(x) = \sum_{\mathbf{e} \neq \mathbf{0}} \prod_{\tau=1}^{l} x^{w(\mathrm{e}_\tau)},$$

where the sum is over all nonzero signal selector error sequences $\mathbf{e} = \{\mathrm{e}_1, \ldots, \mathrm{e}_l\}$ of length l generated by the underlying code and τ represents a time index.

Proof: Since the underlying code is linear, for any correct path $\mathbf{y}$, the set of error events $(\mathbf{y}, \hat{\mathbf{y}})$ can be described by the nonzero codewords of the code. The Lemma follows as a simple consequence of the definitions of the distance spectrum and of the worst case distance spectrum.

The next two Lemmas are also immediate consequences of the definitions of the distance spectrum and of the worst case distance spectrum.

Lemma 2: The worst case distance spectrum of a trellis code which consists of a binary linear convolutional encoder followed by a mapper can be computed by assuming that the correct codeword corresponds to the all-zero information sequence.

Proof: The set of selector error sequences is exactly the set of binary codewords of the underlying code. Thus, the worst case distance spectrum depends only on the underlying code and on the distance function $w(.)$. The distance function $w(.)$ depends only on the signal set and the mapping and not on the regularity of the code. Since the codewords of a linear code can be calculated by assuming that the all-zero information sequence was sent, the worst case distance distance spectrum of any trellis code generated by a binary linear convolutional encoder can be computed by assuming that the all-zero information sequence was sent.

Lemma 3: The worst case distance spectrum and the distance spectrum of a regular code which consists of a binary linear convolutional encoder followed by a mapper are equal.

Proof: Given a correct path $\mathbf{y}$, to each incorrect path $\hat{\mathbf{y}}$ corresponds a unique nonzero error sequence $\mathbf{e}$, which is a codeword since the sum of any two codewords in a linear code is a codeword. Therefore, the sum over $\hat{\mathbf{y}}$ in (5) can be replaced with a sum over $\mathbf{e} \neq \mathbf{0}$. Since the code is regular, $d(y_\tau, \hat{y}_\tau) = w(e_\tau)$, where e_τ is the binary sum of the signal selectors of y_τ and $\hat{y}_\tau$. Equation (5) can then be rewritten as

$$\begin{aligned} SP^{(l)}(x) &= \sum_{\mathbf{y}} p(\mathbf{y}) \sum_{\mathbf{e}\neq\mathbf{0}} \prod_{\tau=1}^{l} x^{w(e_\tau)} \\ &= \sum_{\mathbf{e}\neq\mathbf{0}} \prod_{\tau=1}^{l} x^{w(e_\tau)} \sum_{\mathbf{y}} p(\mathbf{y}) \\ &= \sum_{\mathbf{e}\neq\mathbf{0}} \prod_{\tau=1}^{l} x^{w(e_\tau)}, \end{aligned} \tag{6}$$

since the sum over $\mathbf{e} \neq \mathbf{0}$ does not depend on the correct path $\mathbf{y}$. Equation (6) is exactly the worst case distance spectrum of the code, which proves the Lemma.

The next Theorem is the backbone of the algorithm for computing the distance spectrum of trellis codes. The proof of the Theorem consists of expressing the distance spectrum of a quasi-regular code as a product of the polynomials $P_e(x)$ defined above. Then this product can be computed from the worst case distance spectrum, provided that the underlying code

is linear. Since the worst case distance spectrum of quasi-regular codes can be computed by assuming that the all-zero information sequence was sent, the distance spectrum can also be computed from the all-zero information sequence. The idea of the proof is to group together all the error events that correspond to the same sequence of signal selector errors. The proof is by induction on the length of this error sequence. For each selector error in a sequence, the distance can be computed using the polynomial $P_e(x)$, provided that the error e is the sum of two signal selectors that leave the correct and incorrect states, respectively.

Theorem 1: The distance spectrum of a quasi-regular code can be computed from its worst case distance spectrum.

Proof: The proof of Theorem 1 is given in Appendix A.

The proof of Theorem 1 shows that to compute $SP^{(l)}(x)$ it is sufficient to replace $x^{w(e)}$ in $SP_w^{(l)}(x)$ with $P_e(x)$ whenever $e \in E$, which can be done by knowing the number of occurrences of each $e \in E$ for each incorrect path. Theorem 1 is similar to Zehavi and Wolf's first Theorem which states that the distance spectrum of Ungerboeck's codes can be computed from a state diagram with 2^ν states, where ν is the constraint length of the code [14]. The proof given in their paper is simpler than the proof given above because it applies to a smaller class of codes.

The main advantage of our approach is that it allows the computation of the performance of trellis codes with significant constraint lengths, whereas Zehavi and Wolf's approach requires the computation of a modified transfer function and is limited only to very small constraint lengths. The algorithm described in the next section computes the worst case distance spectrum of quasi-regular codes by assuming that the all-zero information sequence was sent. Hence, the distance spectrum of quasi-regular codes can be computed by assuming that the all-zero information sequence was sent, although unlike regular codes, all correct paths do not give the same distribution of distances.

Lemma 4: Ungerboeck rate $k/(k+1)$ systematic codes are quasi-regular.

Proof: Because of the rate of the codes, half the signals in the signal set leave each state in the trellis. Two cases can occur: either s and $\hat{s}$ are labeled with the same half of the signal set, or they are labeled with different halves. If $(s_1, \hat{s}_1)$ and $(s_2, \hat{s}_2)$ correspond to the same case, then $P_{s_1,\hat{s}_1,e}(x) = P_{s_2,\hat{s}_2,e}(x)$. If the two pairs correspond to different cases, then one of the two polynomials is not defined because distinct cases correspond to distinct sets

of signal selector errors (distinct least significant bits in e). This proves the Lemma.

Ungerboeck rate $k/(k+1)$ systematic codes are quasi-regular, which allows fast distance computation and code search algorithms [10]. Quasi-regularity does not require that the set of distances of paths from a correct path is the same for all correct paths. It only requires that the distance spectrum is calculable from the set of distances from the all-zero path.

3 The Algorithm

The algorithm is a modified version of Chevillat's stack algorithm for calculating the distance profile of convolutional codes [4]. Bahl and Jelinek [1] and later Larsen [7] used a bidirectional algorithm for calculating the free distance of convolutional codes which extends paths forward and backward simultaneously. Our algorithm is also bidirectional.

These previous algorithms terminate when the free distance is reached, whereas our algorithm continues to compute the higher distance spectral lines. It keeps track of the number of paths with the same distance and of the total information sequence weight along these paths. All the paths that reach a given state are retained, whether they have different distances or not. (In conventional free distance computation algorithms, only the path with the smallest distance is retained.) When a merger is detected, two cases can occur: (1) if no previous merging paths had a distance equal to that of the new merger, a new spectral line is created, or (2) if the distance of the merging paths is equal to the distance of an existing spectral line, the multiplicity of the line is incremented. Naturally, this algorithm requires more computation and storage than conventional free distance algorithms because no paths are discarded.

The complexity of the stack decoding algorithm depends on the number of paths that must be extended and not on the constraint length of the code. The multiplicity of a spectral line of distance d is achieved when all remaining paths have a distance larger than d. In most codes, the longest free distance paths are several constraint lengths long. Similarly, it can be conjectured that the longest paths that have a certain distance are several constraint lengths longer than the shortest paths with the same distance.

For example, for a code whose shortest register length is 3, the shortest mergers are 4 branches long. The longest error event with the free distance will be about 20 branches long. If the code is a rate 2/3 code, the number of paths of length 20 is $4^{20} \sim 10^{12}$. Computing the free distance consists in finding the path among these 10^{12} paths that has

the smallest distance. Computing the multiplicity of the free distance consists in calculating the number of paths whose distance equals the free distance, and so on for each spectral line. Obviously, this can can be an enormous task, even for simple codes.

A *forward path* leaves the all-zero state on a non-zero branch in the forward direction, and once it has remerged with the all-zero path it is discarded. A *backward path* leaves the all-zero state on a non-zero branch in the backward direction, i.e., it consists of a succession of branches that lead to the all-zero state.

A path is determined by the following information:

	Direction (fw or bw)
s	Terminal state
d	Distance
l	Length
W	Array of occurrences of $e \in E$
$A_{s,d,e,W}$	Multiplicity
$B_{s,d,e,W}$	Information weight

The *terminal state* s is the last state reached by a path of *length* l branches from the all-zero state for a forward path and to the all-zero state for a backward path. The *distance* d is the worst case distance along that path, i.e., the distance calculated from the $w(e)$'s. The *array W of occurrences of* $e \in E$ is the number of branches on the incorrect path for which the signal selector was e for each $e \in E$, because the signal selector on the correct path is always zero. Two paths are *identical* iff they have the same structure, i.e., the same distance d, length l, and array W. The *multiplicity* $A_{s,d,e,W}$ is the number of identical paths ending in state s with distance d, length l, and array W. The *information weight* $B_{s,d,e,W}$ is the average of the information weights of all the identical paths (although the paths are identical, since they correspond to different topological paths, they may have different information weights).

In the stack algorithm, an ordered stack of previously examined paths with different parameters is stored. For the bidirectional algorithm, two stacks are necessary, one for each direction. Each stack entry contains a path with all its information. Paths are ordered by decreasing distances and lengths. Paths with the same metrics and lengths but distinct terminal states or arrays W are stored on a last in, first out basis within the stack. This does not affect the efficiency of the algorithm, but accelerates the searching and sorting in the stack. The top path in the stack is the most likely to give the free distance or the shortest merging distance among all the paths in the stack, which is why it is extended first.

Each complete sequence of steps consists of extending the top path in the stack by computing its $2^{\bar{k}}$ successors. The terminal state cannot be the all-zero state because it must reach the all-zero state through the terminal state of a path from the opposite direction. If one of the successors reaches the all-zero state directly, it is discarded. Assume, without loss of generality, that a forward path is extended. Then three situations may occur:

(i) The state is not a terminal state for any forward path. Then a new entry is created in the stack of forward paths to store the new path and its parameters.

(ii) The state is the terminal state of one or more forward paths. Compare the distance, length, and W of the two or more paths. If the new path is not identical to any old path create a new entry. If it is identical to an old path, increment the multiplicity and information weight of that path.

(iii) The state is the terminal state of one or more backward paths. The path is then merged with these backward paths to form one or more error events.

A forward path can reach the all-zero state only through the terminal state of a backward path, because all the branches that leave a state in the middle of a backward path have been extended. Thus, it is impossible to reach this middle state without following an extended branch.

The spectral lines are stored by decreasing distance. Every time (iii) occurs, an old spectral line is incremented or a new one is created. The distance of an error event is the sum of the forward and backward distances, the length is the sum of the lengths, W is the sum of the W's, the multiplicity is the product of the multiplicities, and the information weight is $B_{fw}+B_{bw}$, where B_{fw} and B_{bw} are the forward and backward average information weights.

There is no specific time for the algorithm to terminate. It depends on the number of spectral lines required by the user, or on specified maximum path lengths or distances. Once the algorithm is terminated, the worst case distance spectrum is converted into the distance spectrum of the code, and each worst case spectral line is expanded into many new spectral lines [9].

Each worst case spectral line (d, A_d, W) is expanded by replacing all possible combinations of the $w(e)$'s for which $e \in E$ by the other possible distances between signals whose selectors differ by e. For example, for 8-PSK, E has two elements, 011 and 111, and if the W of a particular worst case spectral line contains the two occurrences 1 and 3, it means that $e = 011$ occurred once, and $e = 111$ occurred three times along each path represented

by this spectral line. Since each element in E corresponds to two possible distances (for 8-PSK), there are 16 ways of combining these distances on the four (1 + 3) branches that correspond to $e \in E$. This means that this particular worst case spectral line can be broken into 16 lines. The distance of each line is found by replacing $w(e)$ by the other possible distances for all $e \in E$.

The probability of a distance that corresponds to a specific $e \in E$ is given by the coefficient of that distance in $P_e(x)$. For example, if $P_e(x) = 1/2x^{\delta_0} + 1/2x^{\delta_2}$, the probabilities of δ_0 and δ_2 are both 1/2. For each one of the 16 lines in the above example, there are four distances that correspond to a signal selector from E. The probability of having a specific set of four distances is the product of the probabilities of the individual distances computed from the corresponding $P_e(x)$. Then the average multiplicity of each new line among the 16 lines is the product of A_d and the probability of each of the four distances. If the probabilities of the various distances are equal for all $e \in E$, the 16 lines will have the same average multiplicity.

The algorithm

Step 1. Load the stack with the origin node, with distance zero, length 0, and multiplicity 1. All the other parameters are set to zero. Enter the number N of desired spectral lines.

Step 2. Compute the metric, length, multiplicity, information weight, and W's of the successors of the top path in the stack.

Step 3. Delete the top path from the stack.

Step 4. For each successor, check if it merges and update the merger information [(iii)].

Step 5. Insert the new paths in the stack, and rearrange the stack in order of decreasing distance and length [(i) or (ii)].

Step 6. Output all the new spectral lines whose distance is smaller than the sum of the minimum forward and backward distances. If less than N spectral lines have been found, then change direction and repeat steps 1 to 6; otherwise, stop.

Figure 4 shows the first four forward and backward steps. The bold line represents the correct path (the all-zero path). The trellis code has four states. Step 1a: one forward path is extended (no remerging because state s_2 has not yet been reached). Note that the all-zero branch is not extented. Step 1b: one backward path is extended (no remerging because state s_3 has not yet been reached). Again, note that the all-zero branch is not extended. Step 2a: the top forward path which terminates at s_1 is extended to s_2 and s_3. One successor terminates at s_2, which has been reached by a backward path, so a merger is found (dashed line). Step 2b: the backward path that terminates at s_2 is extended to

reach s_1 and s_3 (one merger through s_3). Step 3a: the forward path that terminates at s_3 is extended to reach s_2 and s_3 (one merger through s_3). Step 3b: from s_1, only one backward branch is extended because paths are not allowed to reach the all-zero state directly. A path must remerge with the all-zero path through a terminal state from the opposite direction. This ensures that mergers are not counted several times. Since s_2 terminates two forward paths, two mergers are found. Step 4a: again only one branch is extended because the other one reaches the all-zero state directly (no mergers because s_1 is not the terminal state of any backward path). Step 4b: the backward path that terminates at s_3 is extended to reach s_1 and s_3 (one merger is found per new state).

The beginning of the distance spectrum can be used to upper bound the first error event probability and the bit error probability of trellis codes. We calculate these performance bounds for the some of the best known codes [10]. Figure 5 shows coded 8-PSK for constraint lengths 4, 6, 8, and 10 compared to uncoded QPSK. Figure 6 shows the distance spectrum of 16 state coded 8-PSK. The free distance of this code is 5.13, and the multiplicity of the free distance is 2.25, i.e., on the average, 2.25 incorrect paths are distance 5.13 from the correct path. The next spectral line is relatively far from the free distance, and its multiplicity is still moderate. This means that for high enough SNR, the free distance is a relatively accurate indication of the performance of this particular code. The larger spectral lines are less spread out than the smaller spectral lines. This is a common property of trellis codes, as opposed to convolutional codes where the spectral lines are separated by integer distances. Note that the multiplicities of the large spectral lines are large.

We have also noted that the distance spectrum of regular codes is usually denser than the distance spectrum of non-regular codes for the smaller spectral lines. This is because the free distance and most of the smaller spectral lines in a non-regular code are exceptional events and do not occur for all correct paths. Figure 7 shows a comparison of the simulation results performed by Ungerboeck [10] with the performance bounds computed from the distance spectrum. The practical ranges where these codes may be used are below error probabilities of 10^{-5}. For such low probabilities, simulations become difficult, and the distance spectrum bound is a good alternative. It is much tighter than estimates based on the free distance alone.

4 Conclusion

A class of non-regular codes, called quasi-regular codes, was defined whose distance spectrum can be calculated by assuming that the all-zero information sequence was sent. Convolutional codes and regular codes are both quasi-regular, as well as most of the best known trellis codes. An algorithm to compute the distance spectrum of quasi-regular trellis codes was presented. Tight performance estimates can be calculated from the first few spectral lines of most of the best known codes, and several examples show that this is an attractive alternative to simulations.

5 Acknowledgement

We wish to acknowledge the help of Mr. Christian Schlegel in preparing the program used to compute the distance spectrum of trellis codes.

Appendix A

A Proof of Theorem 1

It is sufficient to prove that the distance spectrum at any depth l of a quasi-regular code can be computed from its worst case distance spectrum at depth l, since the distance spectrum can be computed from the distance spectra for all l. The distance spectrum at depth l satisfies

$$SP^{(l)}(x) = \sum_{\mathbf{y}} p(\mathbf{y}) \sum_{\hat{\mathbf{y}}} \prod_{\tau=1}^{l} x^{d(y_\tau, \hat{y}_\tau)}, \tag{A.1}$$

where $(\mathbf{y}, \hat{\mathbf{y}})$ represents the l branches of a first event error of length l. A path $\mathbf{y}$ of length l defines a unique state sequence $\mathbf{s} = \{s_0, s_1, \ldots, s_l\}$ of $l+1$ states (note that two different paths may have the same state sequence if the trellis has parallel transitions). Let s_{l-1} and $\hat{s}_{l-1}$ be the l^{th} states of a correct and incorrect path, respectively. The paths $\mathbf{y}$ whose l^{th} state is the specific state s_{l-1} are denoted by $\mathbf{y}|s_{l-1}$. In (A.1), the correct paths that reach the same state s_{l-1} are grouped together and the incorrect paths that reach the same state $\hat{s}_{l-1}$ are grouped together. Then,

$$SP^{(l)}(x) = \sum_{s_{l-1}} p(s_{l-1}) \sum_{\hat{s}_{l-1}} \sum_{\mathbf{y}|s_{l-1}} p(\mathbf{y}|s_{l-1}) \sum_{\hat{\mathbf{y}}|\hat{s}_{l-1}} \prod_{\tau=1}^{l} x^{d(y_\tau, \hat{y}_\tau)}.$$

Let $\mathbf{y}_{l-1}$ denote the first $l-1$ branches of $\mathbf{y}$. Then a path $\mathbf{y}$ can be broken into $\mathbf{y}_{l-1}$ and y_l. Given a state s_{l-1}, the probability $p(\mathbf{y}|s_{l-1})$ of a path $\mathbf{y}$ of length l is $p(\mathbf{y}|s_{l-1}) = p(\mathbf{y}_{l-1}|s_{l-1})p(y_l|s_{l-1})$, i.e., the probability of the path is the product of the probability of reaching state s_{l-1} times the probability of the last branch of $\mathbf{y}$ (which leaves state s_{l-1}). Therefore

$$\begin{aligned}
SP^{(l)}(x) &= \sum_{s_{l-1}} p(s_{l-1}) \sum_{\hat{s}_{l-1}} \sum_{\mathbf{y}_{l-1}|s_{l-1}} p(\mathbf{y}_{l-1}|s_{l-1}) \sum_{\hat{\mathbf{y}}_{l-1}|\hat{s}_{l-1}} \sum_{y_l|s_{l-1}} p(y_l|s_{l-1}) \sum_{\hat{y}_l|\hat{s}_{l-1}} \prod_{\tau=1}^{l} x^{d(y_\tau, \hat{y}_\tau)} \\
&= \sum_{s_{l-1}} p(s_{l-1}) \sum_{\hat{s}_{l-1}} \sum_{y_l|s_{l-1}} p(y_l|s_{l-1}) \sum_{\hat{y}_l|\hat{s}_{l-1}} x^{d(y_l, \hat{y}_l)} \sum_{\mathbf{y}_{l-1}|s_{l-1}} p(\mathbf{y}_{l-1}|s_{l-1}) \sum_{\hat{\mathbf{y}}_{l-1}|\hat{s}_{l-1}} \prod_{\tau=1}^{l-1} x^{d(y_\tau, \hat{y}_\tau)} \\
&= \sum_{s_{l-1}} p(s_{l-1}) \sum_{\hat{s}_{l-1}} \sum_{e_l|s_{l-1}, \hat{s}_{l-1}} \sum_{v|s_{l-1}} p(v|s_{l-1}) x^{d[y(v), y(v+e_l)]} \\
&\qquad \sum_{\mathbf{y}_{l-1}|s_{l-1}} p(\mathbf{y}_{l-1}|s_{l-1}) \sum_{\hat{\mathbf{y}}_{l-1}|\hat{s}_{l-1}} \prod_{\tau=1}^{l-1} x^{d(y_\tau, \hat{y}_\tau)},
\end{aligned} \tag{A.2}$$

where $e_l|s_{l-1}, \hat{s}_{l-1}$ is one of the signal selector errors between branches that leave s_{l-1} and $\hat{s}_{l-1}$. The fourth sum can be replaced with the polynomial $P_{s,\hat{s},e_l}(x)$ so that (A.2) becomes

$$SP^{(l)}(x) = \sum_{s_{l-1}} p(s_{l-1}) \sum_{\hat{s}_{l-1}} \sum_{e_l|s_{l-1},\hat{s}_{l-1}} P_{s,\hat{s},e_l}(x) \sum_{\mathbf{y}_{l-1}|s_{l-1}} p(\mathbf{y}_{l-1}|s_{l-1}) \sum_{\hat{\mathbf{y}}_{l-1}|\hat{s}_{l-1}} \prod_{\tau=1}^{l-1} x^{d(y_\tau,\hat{y}_\tau)}.$$

Because the code is quasi-regular, $P_{s,\hat{s},e_l}(x)$ does not depend on s and $\hat{s}$, provided that e_l is a signal selector error between two branches that leave s and $\hat{s}$, respectively (otherwise the polynomial is not defined). This allows us to switch the summations over the states s_{l-1} and $\hat{s}_{l-1}$ with the summation over e_l, so that

$$SP^{(l)}(x) = \sum_{e_l} P_{s,\hat{s},e_l}(x) \sum_{(s_{l-1},\hat{s}_{l-1}|e_l)} p(s_{l-1}) \sum_{\mathbf{y}_{l-1}|s_{l-1}} p(\mathbf{y}_{l-1}|s_{l-1}) \sum_{\hat{\mathbf{y}}_{l-1}|\hat{s}_{l-1}} \prod_{\tau=1}^{l-1} x^{d(y_\tau,\hat{y}_\tau)},$$

where $(s_{l-1}, \hat{s}_{l-1}|e_l)$ is any pair of states generating signal selectors that differ by e_l. All the first event errors of length l that reach states s_{l-1} and $\hat{s}_{l-1}$ and differ by e_l can be grouped together. These error events are denoted by $(\mathbf{y}, \hat{\mathbf{y}})|e_l$ in the next equation. Also $P_{s,\hat{s},e_l}(x)$ can be written as $P_{e_l}(x)$, since it does not depend on s and $\hat{s}$. Therefore

$$SP^{(l)}(x) = \sum_{e_l} P_{e_l}(x) \sum_{(\mathbf{y},\hat{\mathbf{y}})|e_l} p(\mathbf{y}_{l-1}) \prod_{\tau=1}^{l-1} x^{d(y_\tau,\hat{y}_\tau)}. \tag{A.3}$$

From (A.3) and the definition of the distance spectrum at depth l,

$$\sum_{(\mathbf{y},\hat{\mathbf{y}})} p(\mathbf{y}_l) \prod_{\tau=1}^{l} x^{d(y_\tau,\hat{y}_\tau)} = \sum_{e_l} P_{e_l}(x) \sum_{(\mathbf{y},\hat{\mathbf{y}})|e_l} p(\mathbf{y}_{l-1}) \prod_{\tau=1}^{l-1} x^{d(y_\tau,\hat{y}_\tau)}. \tag{A.4}$$

The same procedure can be repeated with the right side of (A.4) to express the distance spectrum at depth l as a function of a sum over error events $(\mathbf{y}, \hat{\mathbf{y}})$ conditioned on two consecutive time intervals (i.e., conditioned on e_l and e_{l-1}). Furthermore, this result can be extended by induction on the number of signal selector errors on which the beginning of the error event is conditioned to obtain for any $1 \leq \kappa \leq l-1$

$$\sum_{(\mathbf{y},\hat{\mathbf{y}})|e_l,\ldots,e_{\kappa+1}} p(\mathbf{y}_\kappa) \prod_{\tau=1}^{\kappa} x^{d(y_\tau,\hat{y}_\tau)} = \sum_{e_\kappa} P_{e_\kappa}(x) \sum_{(\mathbf{y},\hat{\mathbf{y}})|e_l,\ldots,e_\kappa} p(\mathbf{y}_{\kappa-1}) \prod_{\tau=1}^{\kappa-1} x^{d(y_\tau,\hat{y}_\tau)}, \tag{A.5}$$

where $(\mathbf{y}, \hat{\mathbf{y}})|e_l, \ldots, e_{\kappa+1}$ is a first event error of length l whose $(\kappa+1)^{\text{th}} \ldots l^{\text{th}}$ branches are labeled with signals whose selectors differ by $e_l, \ldots, e_{\kappa+1}$. In particular, for $\kappa = 1$, (A.5) becomes

$$\sum_{(\mathbf{y},\hat{\mathbf{y}})|e_l,\ldots,e_2} p(\mathbf{y}_1) \prod_{\tau=1}^{1} x^{d(y_\tau,\hat{y}_\tau)} = \sum_{e_1} P_{e_1}(x). \tag{A.6}$$

Hence from (A.3), (A.4), (A.5), and (A.6), we obtain

$$SP^{(l)}(x) = \sum_{\mathbf{e}\neq\mathbf{0}} \prod_{\tau=1}^{l} P_{e_\tau}(x),$$

where the summation is over all nonzero signal selector error sequences $\mathbf{e}$ of length l. Note that when e_τ does not belong to the set E, then $P_{e_\tau}(x) = x^{w(e_\tau)}$. Assuming that $SP_w^{(l)}(x)$ can be computed, then the distance spectrum can be calculated from $SP_w^{(l)}(x)$ by replacing $x^{w(e_\tau)}$ with $P_{e_\tau}(x)$ for each $e_\tau \epsilon E$. This concludes the proof.

References

[1] L. R. Bahl and F. Jelinek, "Rate 1/2 Convolutional Codes with Complementary Generators", *IEEE Trans. Inform. Theory*, Vol. IT-17, No. 6, pp. 718-727, November 1971.

[2] E. Biglieri, "High-Level Modulation and Coding for Nonlinear Satellite Channels", *IEEE Trans. Commun.*, COM-32, pp. 616-626, May 1984.

[3] A. R. Calderbank and N. J. A. Sloane, "New Trellis Codes", *IEEE Trans. Inform. Theory*, IT-33, pp. 177-195, March 1987.

[4] P. R. Chevillat, "Fast Sequential Decoding and a New Complete Decoding AlgoritIllinois Institute of Technology, Chicago, IL, 1976.

[5] G. D. Forney, Jr., "Coset Codes I: Geometry and Classification," *IEEE Trans. Inform. Theory*, to appear.

[6] G. D Forney, Jr., R. G. Gallager, G. R. Lang, F. M. Longstaff, and S. U. Qureshi, "Efficient Modulation for Band-limited Channels." *IEEE J. on Selected Areas in Communication*, SAC-2, No. 5, pp. 632-647, Sept. 1984.

[7] K. J. Larsen, "Comments on 'An Efficient Algorithm for Computing Free Distance'," *IEEE Trans. Inform. Theory*, Vol. IT-19, pp. 577-579, July 1973.

[8] S. Lin and D. J. Costello, Jr., *Error Control Coding*, Prentice-Hall, Englewood Cliffs, N.J., 1983.

[9] M. Rouanne, "Distance Bounds and Construction Algorithms for Trellis Codes," Ph.D Thesis, University of Notre Dame, Notre Dame, IN, 1988.

[10] G. Ungerboeck, "Channel Coding with Multilevel Phase Signals," *IEEE Trans. Inform. Theory*, IT-28, pp. 55-67, Jan. 1982.

[11] G. Ungerboeck, "Trellis Coded Modulation with Redundant Signal Sets - Part I: Introduction and Part II: State of the Art," *IEEE Communications Magazine*, Vol. 25, no. 2, pp. 5-21, Feb. 1987.

[12] A. J. Viterbi and J. K. Omura, *Principles of Digital Communications and Coding*, McGraw-Hill, New York, 1979.

[13] L. F. Wei, "Trellis-Coded Modulation with Multi-Dimensional Constellations," *IEEE Trans. Inform. Theory*, IT-33, pp. 483-501, July 1987.

[14] E. Zehavi and J. K. Wolf, "On the Performance Evaluation of Trellis Codes," *IEEE Trans. Inform. Theory*, IT-33, pp. 196-202, 1987.

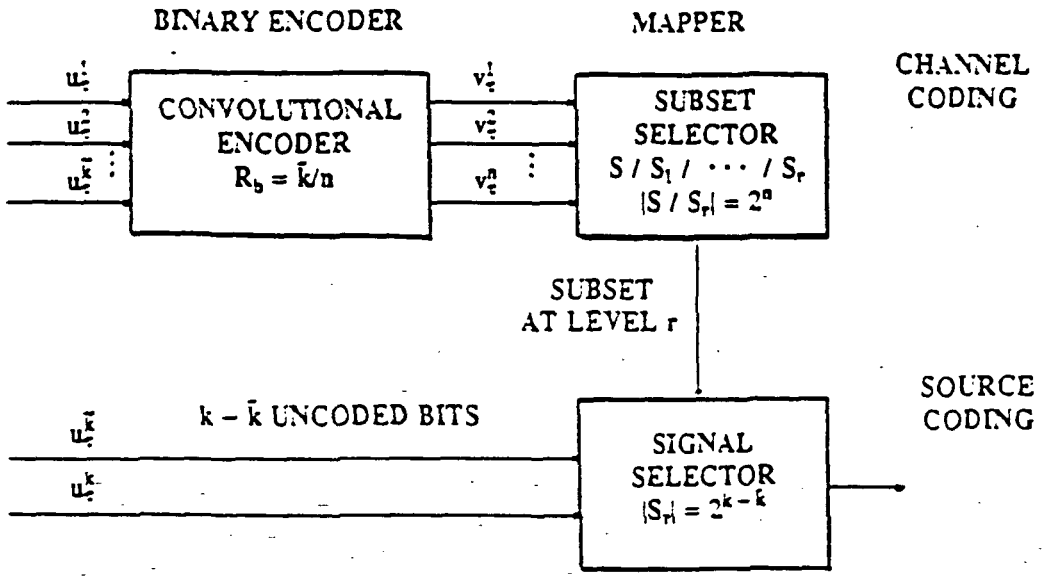


Figure 1: Schematic representation of a typical trellis code.

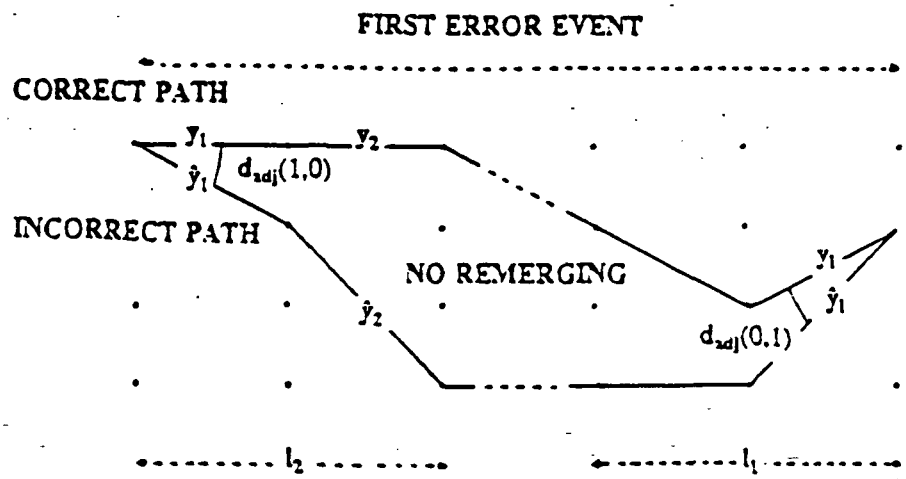


Figure 2: A first event error.

C-2

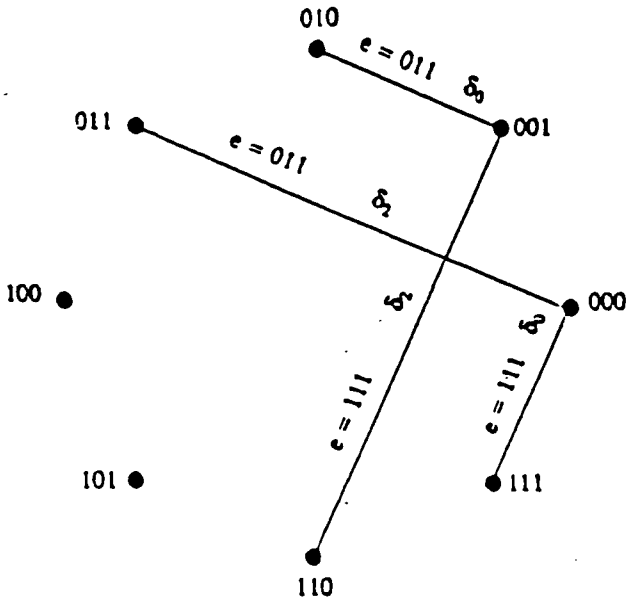


Figure 3: A non-regular mapping onto 8-PSK.

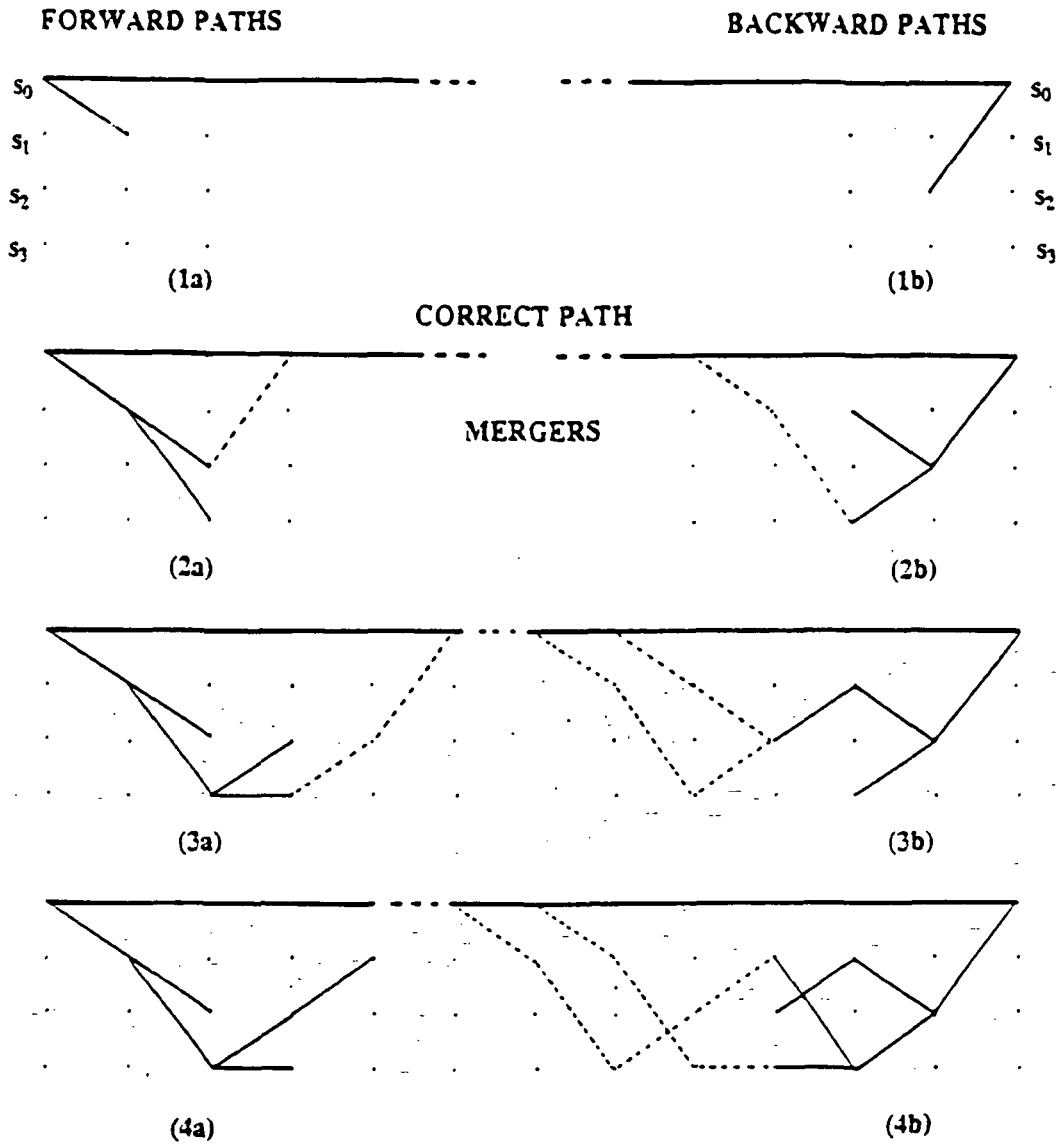


Figure 4: Eight iterations of the algorithm.

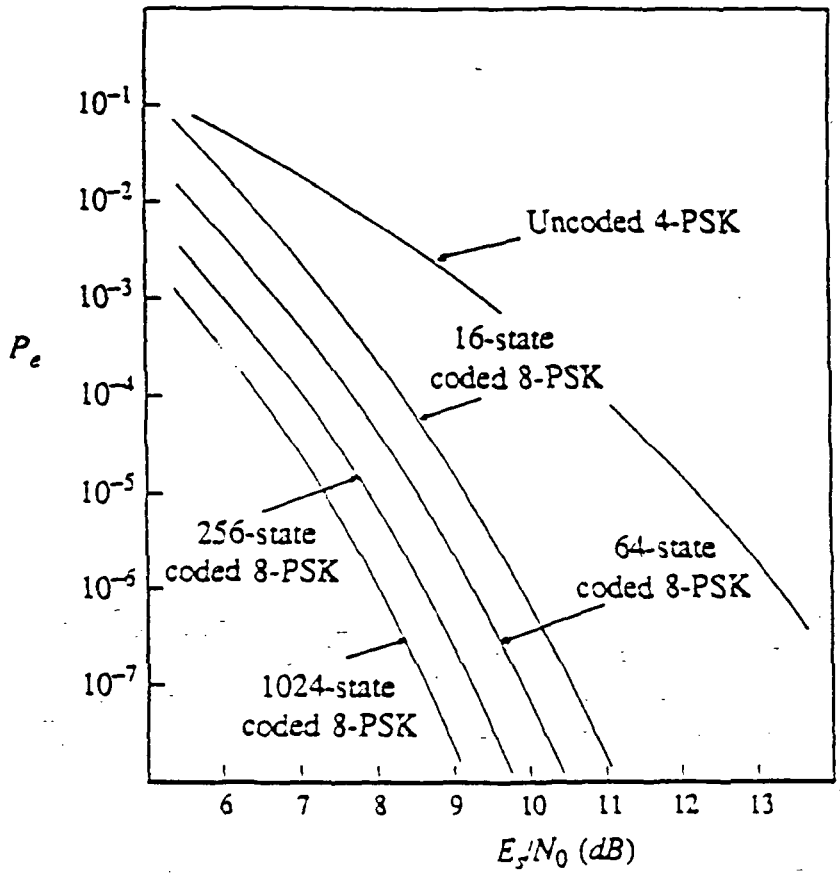


Figure 5: Performance of coded 8-PSK.

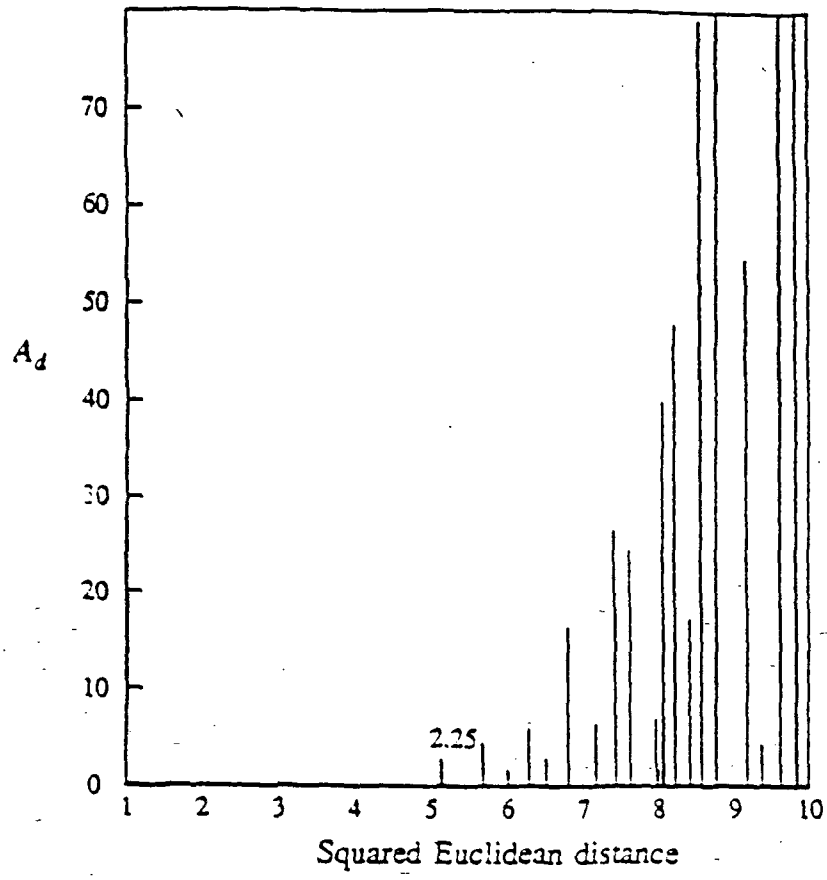


Figure 6: Distance spectrum of 16-state coded 8-PSK.

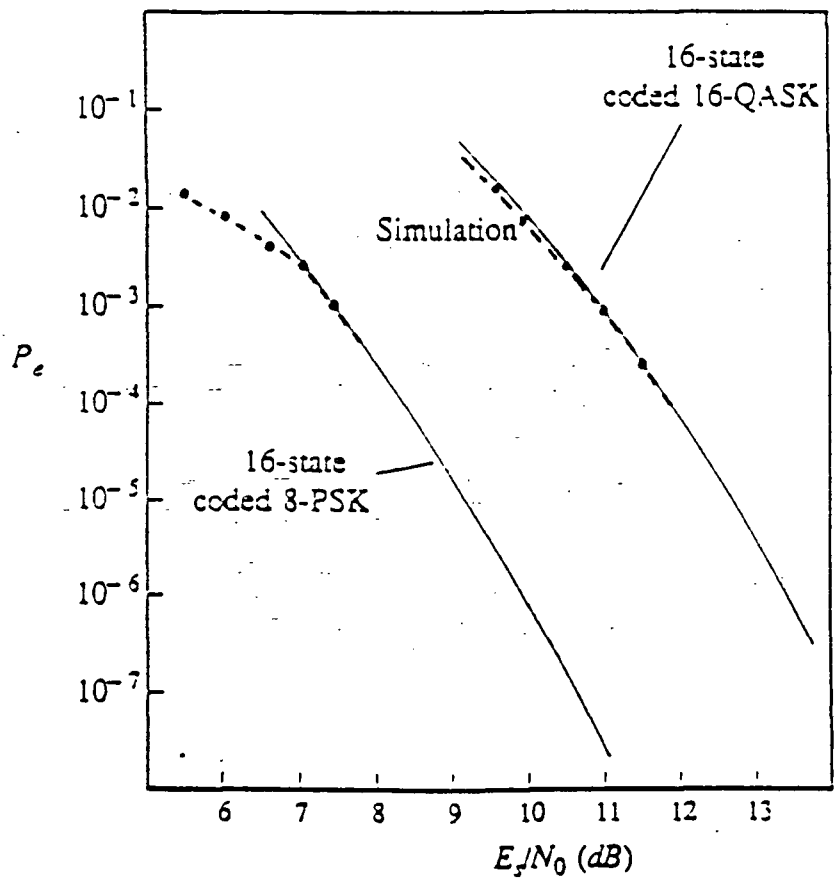


Figure 7: Comparison of performance estimates and simulation results for coded 8-PSK.

Appendix C

Bandwidth Efficient Coding for Fading Channels

Bandwidth Efficient Coding for Fading Channels [1]

Christian Schlegel
Daniel J. Costello, Jr.

Department of Electrical and Computer Engineering
University of Notre Dame
Notre Dame, IN 46556

Submitted to the IEEE Journal on Selected Areas in Communications
May 1, 1988

abstract

Achieving reliable digital communications over fading channels usually requires not only high signal energies, but also large bandwidth expansion factors, in particular for time diversity signaling. It is shown that bandwidth efficient data transmission using trellis coded modulation, introduced for the AWGN-channel, is also feasible on fading channels. The Chernoff bounding technique is used to obtain performance bounds for bandwidth efficient trellis codes on fading channels with various degrees of side information. New design criteria, the effective length and the minimum product distance, are introduced for trellis coded modulation on fading channels. Based on these design criteria, 8-PSK trellis codes for fading channels are constructed. The performance of the new trellis codes is analyzed for fading channels with different degrees of side information, and it is shown that the new codes have a significantly better error performance than codes of the same complexity designed for Gaussian channels.

[1]This work was supported by NASA Grant NAG 5–557.

1 Introduction

In coding theory the most frequently assumed model for a transmission channel is the *additive white Gaussian noise* (AWGN) channel model. However for many communication systems the AWGN-channel is a poor model, and one must resort to more precise and complicated models. One type of non-Gaussian model which frequently occurs in practice is the fading channel. An example of such a fading channel is the mobile satellite communication channel, which has been the subject of several recent articles [1–5]. Mobile satellite communication systems are usually used at low data rates. With *linear predictive coding* (LPC)), digital voice transmission is possible at 2400 bits/s, and it is envisioned that mobile satellite channels can be used up to that data rate.

Fading is caused if the receiving antennas, like the very small antennas used in mobile radio links, pick up multipath reflections. This will cause the channels to exhibit a time varying behavior in the received signal energy, which is called fading. While there are other degradations like time varying dispersion, we will concentrate on the most basic model. We consider *double sideband amplitude modulation* DSB-AM and our receiver uses a DSB demodulator. The transmission channel that arises from such a system is shown in Figure 1. Fading comes about when the communication path is littered with "scattering particles". If the number of scatterers is large, the received signals $I(t)$ and $Q(t)$ will be statistically independent Gaussian processes [2] [6], which translate into statistically independent Gaussian random variables z_i and z_q in signal space. If there is only a diffuse multipath signal, the mean values of z_i and z_q are zero and the amplitude of the signal vector $b\sqrt{E_m} = \sqrt{z_i^2 + z_q^2}$ is Rayleigh distributed with

$$p(b) = 2be^{-b^2}, \tag{1}$$

where $E[b^2 E_m] = E[b^2]E_m = E_m$ is the average energy received via the diffuse multipaths. If there is also a direct line of sight signal, with received signal energy E_d, the amplitude of the total signal is Rician distributed:

$$p(b) = 2b(1+K)e^{-K-b^2(1+K)} I_0\left(2b\sqrt{K(1+K)}\right), \tag{2}$$

where $K = E_d/E_m$ is the ratio of the signal energy received on the direct path to the signal energy received via the diffuse multipaths, and $I_0(\star)$ is the first order modified

spherical Bessel function. Note that (2) reduces to (1) when $K = 0$. We further define the total energy of the received signal $E_S = E_d + E_m$. In the Rayleigh case $E_S = E_m$, since the direct component is zero. With the use of trellis coded modulation, we rely on the feasibility of coherent reception and we assume that the carrier phase can be recovered.

We now discuss the following communication system. The transmitter sends a sequence of 2-dimensional signals $\mathbf{x} = (\underline{x}_1, \cdots, \underline{x}_l)$ over the fading channel, where each signal $\underline{x}_r$ is chosen from some signal set $\mathcal{A} = \{\underline{a}_1, \ldots, \underline{a}_A\}$ of cardinality A. This signal is represented by two analog waveform signals $I(t)$ and $Q(t)$ which modulate the carrier signal $\sqrt{2}\cos\omega_0 t$ and its quadrature component $\sqrt{2}\sin\omega_0 t$. This modulation process translates the signals into the frequency band with center frequency ω_0. This bandpass signal is then transmitted over a bandpass channel with both Gaussian noise and fading. At the receiver the received waveform signal is demodulated into the baseband direct and quadrature components $\hat{I}(t)$ and $\hat{Q}(t)$, which are transformed by the baseband receiver into the sequence of 2-dimensional received signals $\mathbf{y} = (\underline{y}_1, \cdots, \underline{y}_l)$. Each $\underline{y}_r$ is a distorted copy of the transmitted signal $\underline{x}_r$, i.e.,

$$\underline{y}_r = b_r \underline{x}_r + \underline{n}_r, \tag{3}$$

where b_r is the multiplicative distortion introduced by the fading, whose density function is given in (2), and $\underline{n}_r$ is a 2-dimensional Gaussian random variable with variance N_0. For mobile communications the fading usually varies slowly compared to the signal intervals, and therefore we assume that b_r is constant throughout each time interval. The amplitude b_r is called the fading depth at time r. Practical examples of such Rician fading channels are discussed by Hagenauer et. al. in [2] and [3].

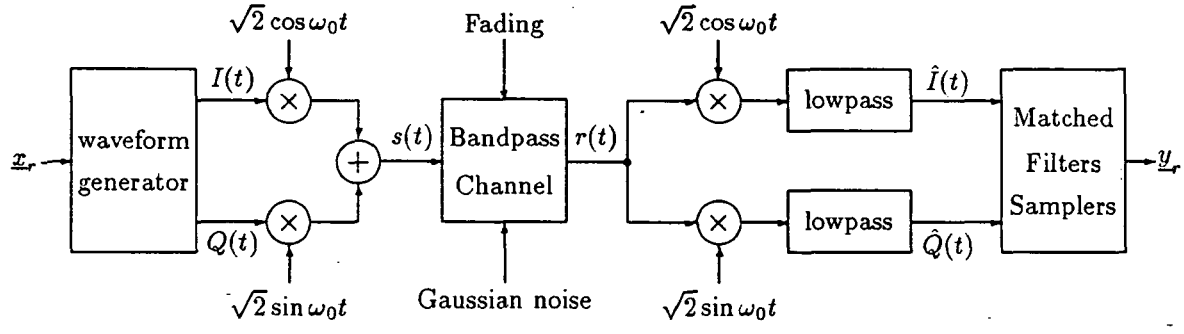


Figure 1: DSB transmission channel model used on the fading channel.

For binary orthogonal signaling on a Rayleigh fading channel, it can be shown that the error probability is given by [6, page 533]

$$P_b = \frac{1}{2 + \frac{E_S}{N_0}}, \tag{4}$$

where E_S is the mean value of the received signal energy and E_S/N_0 is the *channel symbol signal-to-noise ratio* (SNR). In contrast to the AWGN-channel, the error probability on a single transmission decreases only inversely with E_S/N_0. In order to reduce the error probability on a Rayleigh/Rician channel, one must get around the high probability of a deep fade on a single transmission. A popular method to achieve this is diversity transmission. One form of diversity transmission, time diversity, involves sending a symbol L times, where the receiver performs some averaging to achieve an error performance that decreases exponentially with the SNR, i.e.,

$$P_b < e^{-0.149\frac{E_S}{N_0}}. \tag{5}$$

For a more detailed discussion on diversity signaling the reader is referred to [6], [7], and [8]. Retransmitting the same signal L times involves bandwidth expansion by a factor L, which is not tolerable in bandwidth limited environments. Another method is binary coding, which can yield an arbitrarily low error probability, but also at the expense of bandwidth. In this paper, we focus on bandwidth efficient trellis coded modulation (TCM) as a means of achieving reliable digital communications over fading channels without bandwidth expansion. Some of our results have been derived independently by Divsalar and Simon in a paper [9] that focuses on a discussion of multiple trellis coded modulation for fading channels.

2 Performance Bounds

2.1 Chernoff Factors

In this section we present a general method for deriving error performance bounds for coded systems used on memoryless channels[2]. We will apply these results to TCM communication systems whose structure is shown in Figure 2. A TCM communication system consists of a trellis encoder, a signal interleaver, the transmission channel, a signal

[2]This method can be extended to also include finite state channels.

deinterleaver, and a trellis decoder. A rate $R = k/n$ trellis code is generated by a binary convolutional encoder followed by a mapper. The convolutional encoder is a finite state automaton with 2^ν possible states, where ν is the memory order of the encoder. At each time interval r, the encoder accepts $\tilde{k}$ binary input bits $(u_r^{\tilde{k}}, u_r^{\tilde{k}-1}, \ldots, u_r^1)$ and makes a transition from its state S_r at time r to one of $2^{\tilde{k}}$ possible successor states S_{r+1}. The $\tilde{n} = n - (k - \tilde{k})$ output bits of the convolutional encoder and the $k - \tilde{k}$ uncoded information bits $(u_r^k, \ldots, u_r^{\tilde{k}+1})$ form one of 2^n binary n-tuples $v_r = (v_r^n, v_r^{n-1}, \ldots, v_r^1)$, which is translated by the mapper into one of $A = 2^n$ channel signals from a signal set $\mathcal{A} = \{\underline{a}_1, \underline{a}_2, \ldots, \underline{a}_A\}$. The uncoded information bits do not affect the state of the convolutional encoder and cause $2^{k-\tilde{k}}$ parallel transitions between the encoder states S_r and S_{r+1}. Since the coherent DSB-AM system transmits two dimensions in one analog waveform signal, it is sensible to design the trellis encoder for 2-dimensional signal sets. A rate $R = k/n$ trellis code transmits k bits/channel signal, where the channel signal set contains $A = 2^n$ signals. If such a TCM communication system replaces an uncoded system that uses a signal set with $A' = 2^k$ signals, the overall transmission rate is preserved, and we call such a TCM system bandwidth efficient.

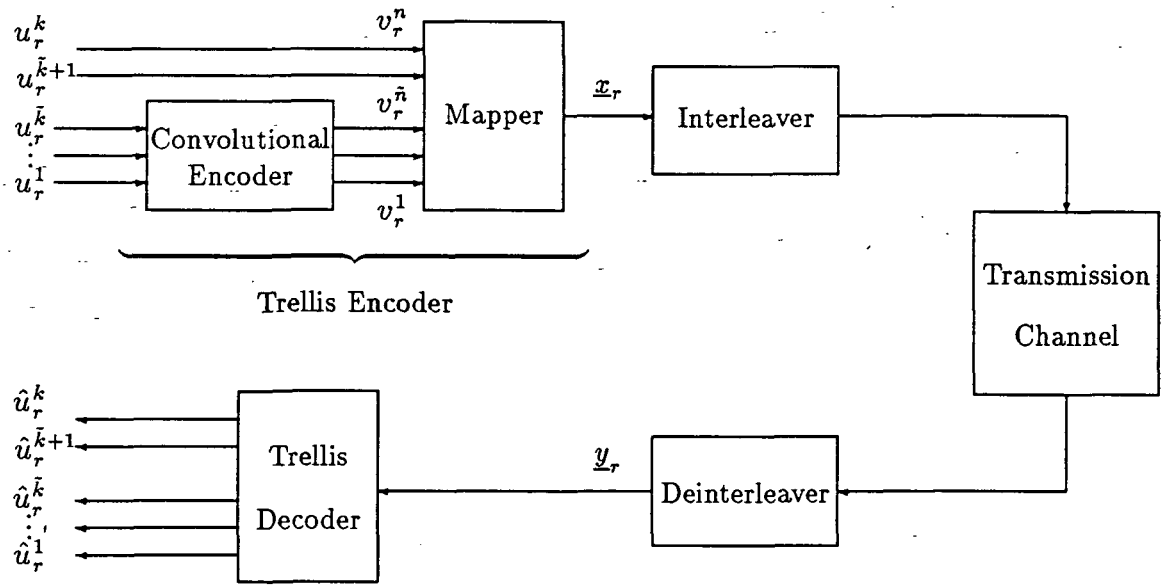


Figure 2: Trellis coded communication system.

The particular transmission channel discussed in this paper is the Rayleigh/Rician fading channel introduced in section 1. The interleaver/deinterleaver converts the channel to a memoryless channel and insures that the signals in the received sequence are independent. They are decoded by a maximum likelihood sequence estimator (usually using the Viterbi algorithm). The Viterbi algorithm finds the signal sequence that most closely corresponds to the sequence of received signals. It achieves this by calculating a decoding metric $m(\mathbf{x}, \mathbf{y})$ between $\mathbf{x}$ and $\mathbf{y}$, where $\mathbf{x} = (\underline{x}_1, \ldots, \underline{x}_l)$ is a possible sequence of 2-dimensional transmitted signals and $\mathbf{y} = (\underline{y}_1, \ldots, \underline{y}_l)$ is the received sequence. $m(\mathbf{x}, \mathbf{y})$ is some non-negative function of $\mathbf{x}$ given $\mathbf{y}$, which is inversely related to the conditional probability that $\mathbf{x}$ was transmitted if $\mathbf{y}$ was received. The decoder will then choose the message sequence $\mathbf{x}$ for which this metric is minimized. It makes an error if it decodes a sequence $\mathbf{x}'$, given that the correct sequence, i.e., the transmitted sequence, was $\mathbf{x}$. This will happen if $m(\mathbf{x}', \mathbf{y}) \leq m(\mathbf{x}, \mathbf{y})$. In the case of channel state side information the decoder will use $m(\mathbf{x}, \mathbf{b}, \mathbf{y})$ as its metric, where $\mathbf{b}$ is the side information obtained from the channel.

The two code word error probability, i.e., the probability that $\mathbf{x}'$ is erroneously decoded if $\mathbf{x}$ is sent is given by

$$P(\mathbf{x} \to \mathbf{x}') = \Pr\{m(\mathbf{x}', \mathbf{y}) - m(\mathbf{x}, \mathbf{y}) \leq 0\}. \tag{6}$$

We use the Chernoff bounding technique [7] to upper bound the above expression and obtain

$$\Pr\{m(\mathbf{x}', \mathbf{y}) - m(\mathbf{x}, \mathbf{y}) \leq 0\} \leq \underset{\mathbf{y}|\mathbf{x}}{E}\left[\exp(-\lambda\{m(\mathbf{x}', \mathbf{y}) - m(\mathbf{x}, \mathbf{y})\})\right], \tag{7}$$

where $\underset{\mathbf{y}|\mathbf{x}}{E}$ denotes conditional expectation and λ is a non-negative real valued parameter over which we minimize the right hand side of (7) to obtain the tightest possible exponential bound, i.e.,

$$\begin{aligned} P(\mathbf{x} \to \mathbf{x}') &\leq \min_{\lambda} \underset{\mathbf{y}|\mathbf{x}}{E}\left[\exp(-\lambda\{m(\mathbf{x}', \mathbf{y}) - m(\mathbf{x}, \mathbf{y})\})\right] \\ &= \min_{\lambda} C(\mathbf{x}, \mathbf{x}', \lambda), \end{aligned} \tag{8}$$

where $C(\mathbf{x}, \mathbf{x}', \lambda)$ is called the Chernoff bound between the signal sequences $\mathbf{x}$ and $\mathbf{x}'$.

Restricting attention to decoders using additive metrics, i.e.,

$$m(\mathbf{x},\mathbf{y}) = \sum_{r=1}^{l} m(\underline{x}_r, \underline{y}_r), \tag{9}$$

we may rewrite (8) as

$$\begin{aligned} P(\mathbf{x} \to \mathbf{x}') &\leq \min_{\lambda} C(\mathbf{x},\mathbf{x}',\lambda) \\ &= \min_{\lambda} \prod_{r=1}^{l} \underset{\underline{y}_r|\underline{x}_r}{E} \left[\exp(-\lambda\{m(\underline{x}'_r,\underline{y}_r) - m(\underline{x}_r,\underline{y}_r)\})\right] \\ &= \min_{\lambda} \prod_{r=1}^{l} C(\underline{x}_r,\underline{x}'_r,\lambda), \end{aligned} \tag{10}$$

where $C(\underline{x}_r,\underline{x}'_r,\lambda)$ is called the *Chernoff factor* of the signals $\underline{x}_r$ and $\underline{x}'_r$. The two code word error probability bound is now given by

$$P(\mathbf{x} \to \mathbf{x}') \leq \min_{\lambda} \prod_{r=1}^{l} C(\underline{x}_r,\underline{x}'_r,\lambda). \tag{11}$$

The Chernoff factors are important because they not only streamline the expression for the two code word error probability but also apply to the transfer function bound for trellis codes introduced later and to the cutoff rate calculations. In particular, it can be shown [10] that R_0, the channel cutoff-rate in bits/transmitted signal, is given by

$$R_0(p) = -\log_2 \min_{\lambda} \sum_{m=1}^{A} \sum_{p=1}^{A} p(\underline{a}_m)p(\underline{a}_p)C(\underline{a}_m,\underline{a}_p,\lambda), \tag{12}$$

where $p(\underline{a}_j)$ is the probability of choosing the signal $\underline{a}_j \in \mathcal{A}$. Note that R_0 is dependent on the particular metric $m(\underline{y}_r,\underline{x}_r)$ that is used by the decoder. If the decoder uses the maximum likelihood (ML)-metric for a memoryless channel, i.e.,

$$\begin{aligned} m(\mathbf{y},\mathbf{x}) &= -\log(\Pr(\mathbf{y}|\mathbf{x})) = -\log \prod_{r=1}^{l} \Pr(\underline{y}_r|\underline{x}_r) \\ &= \sum_{r=1}^{l} \left(-\log(\Pr(\underline{y}_r|\underline{x}_r))\right) = \sum_{r=1}^{l} m(\underline{x}_r,\underline{y}_r), \end{aligned} \tag{13}$$

(12) becomes the channel cutoff-rate for the optimum receiver, which is the usual definition of R_0 [6]. We will denote the value of λ which maximizes the cutoff-rate (12) by λ_{R_0}. In this case, the Chernoff factors will be written as $C(\underline{a}_m,\underline{a}_p) \triangleq C(\underline{a}_m,\underline{a}_p,\lambda_{R_0})$.

2.2 Fading Channel with Side Information

In this section we calculate the Chernoff factors under the assumption that the decoder is furnished with perfect side information, i.e., at each symbol interval r the fading depth b_r is known. We choose our decoding metric as $m(\underline{y}_r, b_r, \underline{x}_r) = |\underline{y}_r - b_r\underline{x}_r|^2$, which would be the maximum likelihood metric on an AWGN-channel if the transmitter sent the signal $b_r\underline{x}_r$ during time interval r. Indeed, the decoder will not know whether the transmitter modulated the signal amplitude by multiplying it by the constant b_r or whether the channel distorted the symbols in that way. Due to the additional stochastic process b, the two code word error probability becomes

$$P(\mathbf{x} \to \mathbf{x}') = \underset{\mathbf{b}}{E}\left[\underset{\mathbf{y}|\mathbf{x}}{E}\left[\Pr\{m(\mathbf{x}', \mathbf{b}, \mathbf{y}) - m(\mathbf{x}, \mathbf{b}, \mathbf{y}) \leq 0\}\right]\right], \tag{14}$$

where $\mathbf{b} = (b_1, \ldots, b_l)$ is the sequence of fading depths. The above probability can be expressed in terms of the Chernoff factors, i.e.,

$$\begin{aligned} P(\mathbf{x} \to \mathbf{x}') \leq \min_{\lambda} C(\mathbf{x}, \mathbf{x}', \lambda) &= \min_{\lambda} \underset{\mathbf{b}}{E}\left[\prod_{r=1}^{l} \underset{\underline{y}_r|\underline{x}_r}{E}\left[\exp\left(-\lambda\left(|\underline{y}_r - b_r\underline{x}'_r|^2 - |\underline{y}_r - b_r\underline{x}_r|^2\right)\right)\right]\right] \\ &= \min_{\lambda} \underset{\mathbf{b}}{E}\left[\prod_{r=1}^{l} \exp\left(-\lambda b_r^2(\underline{x}_r - \underline{x}'_r)^2(1 - \lambda N_0)\right)\right]. \end{aligned} \tag{15}$$

The product inside the expectation is minimized by setting $\lambda = \lambda_{R_0} = 1/(2N_0)$ and the Chernoff factors become independent of λ, i.e.,

$$\begin{aligned} P(\mathbf{x} \to \mathbf{x}') &\leq \underset{\mathbf{b}}{E}\left[\prod_{r=1}^{l} \exp\left(-\frac{b_r^2}{4N_0}(\underline{x}_r - \underline{x}'_r)^2\right)\right] \\ &= \prod_{r=1}^{l} \underset{b_r}{E}\left[\exp\left(-\frac{{b_r}^2}{4N_0}(\underline{x}_r - \underline{x}'_r)^2\right)\right], \end{aligned} \tag{16}$$

and

$$C(\underline{x}_r, \underline{x}'_r) = \underset{b_r}{E}\left[\exp\left(-\frac{{b_r}^2}{4N_0}(\underline{x}_r - \underline{x}'_r)^2\right)\right]. \tag{17}$$

Due to the interleaving the fading depths b_r are governed by independent identically distributed probability distributions and the subscript r on b in the above equation can be dropped. Then we have

$$C(\underline{x}_r, \underline{x}'_r) = \underset{b}{E}\left[\exp\left(-\frac{b^2}{4N_0}(\underline{x}_r - \underline{x}'_r)^2\right)\right] = \int_0^\infty p(b) e^{-b^2\frac{(\underline{x}_r - \underline{x}'_r)^2}{4N_0}} db, \tag{18}$$

where $p(b)$ is the Rayleigh-Rician probability distribution given in (2). After some manipulations (18) becomes

$$C(\underline{x}_r, \underline{x}'_r) = \frac{1+K}{1+K+\frac{(\underline{x}_r - \underline{x}'_r)^2}{4N_0}} e^{-\frac{K\frac{(\underline{x}_r - \underline{x}'_r)^2}{4N_0}}{1+K+\frac{(\underline{x}_r - \underline{x}'_r)^2}{4N_0}}}. \tag{19}$$

Figure 3 shows R_0 from (12) for the Rayleigh ($K = 0$) fading channel with side information for several two dimensional signal constellations. In this case, R_0 is 3–6dB smaller than on the AWGN channel [6], [10], but the relative performance of each signal constellation is preserved. It is worth noting from (5) that if diversity signaling is used with binary symbols on a Rayleigh fading channel, there is a loss of 5.25dB in SNR compared to the AWGN-channel with the same noise power spectral density N_0, at the expense of considerable bandwidth expansion. The R_0-curves for the fading channel on the other hand assure that good error performance (i.e., comparable error probabilities to the AWGN-channel), without the bandwidth expansion introduced by diversity signaling, is possible, albeit at higher values of the SNR E_S/N_0.

2.3 Fading Channel Without Side Information

With no information on the fading depth available, the decoder uses the maximum likelihood metric for the AWGN-channel, i.e., $m(\underline{x}_r, \underline{y}_r) = |\underline{y}_r - \underline{x}_r|^2$, and we obtain for the Chernoff bound

$$\begin{aligned} P(\mathbf{x} \to \mathbf{x}') &\leq \min_\lambda C(\mathbf{x}, \mathbf{x}', \lambda) = \min_\lambda \underset{\mathbf{b}}{E}\left[\prod_{r=1}^{l} \underset{\underline{y}_r|\underline{x}_r}{E}\left[\exp(-\lambda(|\underline{y}_r - \underline{x}'_r|^2 - |\underline{y}_r - \underline{x}_r|^2))\right]\right] \\ &= \min_\lambda \prod_{r=1}^{l} \underset{b}{E}\left[\exp(\lambda(\underline{x}_r^2 - \underline{x}_r'^2) - 2\lambda b_r(\underline{x}_r^2 - \underline{x}_r\underline{x}'_r) + \lambda^2 N_0(\underline{x}_r - \underline{x}'_r)^2)\right], \end{aligned} \tag{20}$$

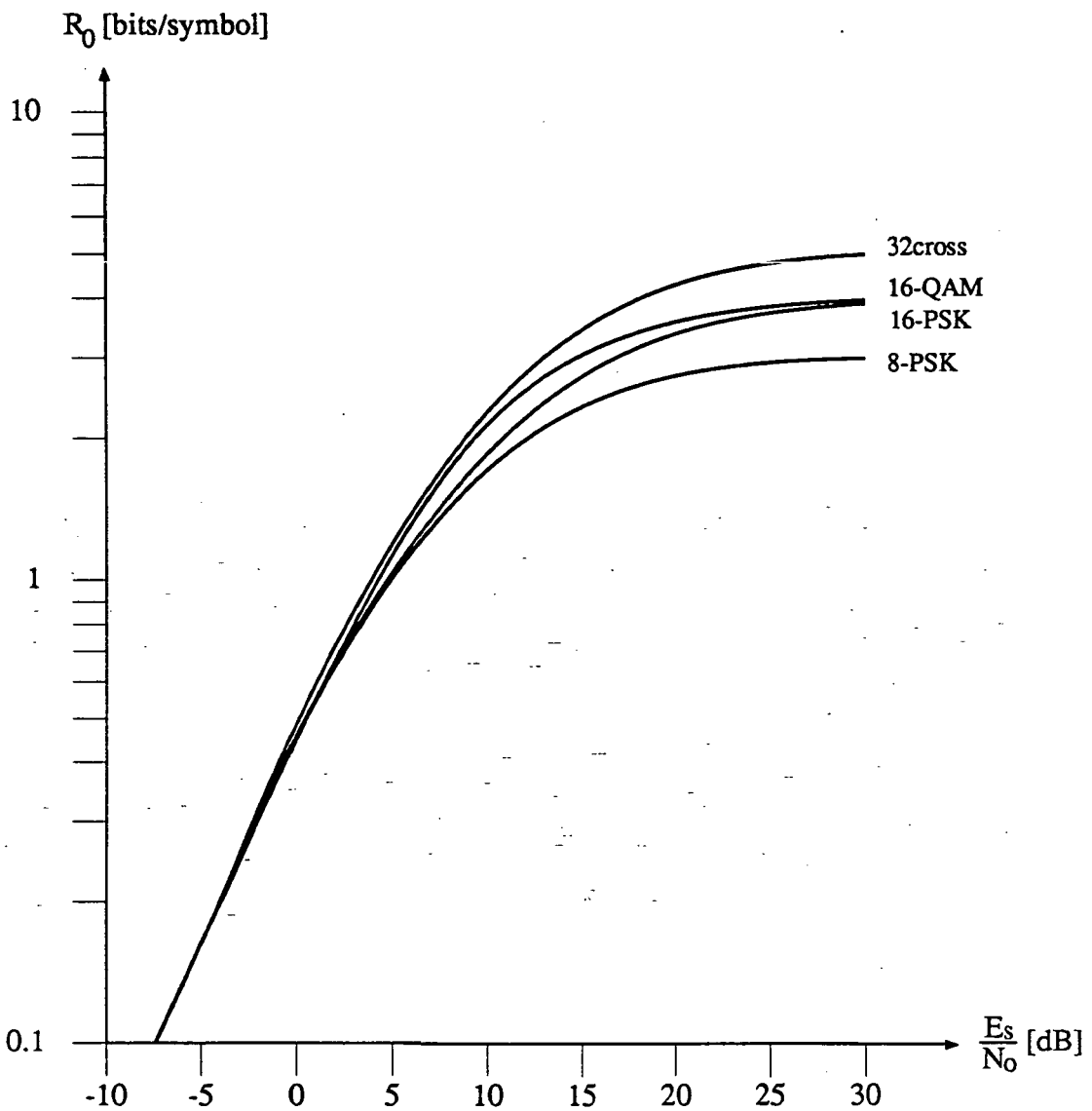


Figure 3: Cutoff-rates for the Rayleigh (K=0) fading channel with side information.

which gives us the following expression for the Chernoff factors for the fading channel without side information,

$$C(\underline{x}_r', \underline{x}_r, \lambda) = e^{\lambda(\underline{x}_r^2 - \underline{x}_r'^2) + \lambda^2 N_0 (\underline{x}_r - \underline{x}_r')^2} \underset{b}{E}\left[e^{-2\lambda b(\underline{x}_r^2 - \underline{x}_r \underline{x}_r')}\right]. \tag{21}$$

$\underset{b}{E}\left[e^{-2\lambda b(\underline{x}_r^2 - \underline{x}_r \underline{x}_r')}\right]$ can be evaluated as follows:

$$\underset{b}{E}\left[e^{-2\lambda b(\underline{x}_r^2 - \underline{x}_r \underline{x}_r')}\right] = \frac{e^{-K}}{\pi} \int_0^{\pi} \left(1 - \sqrt{\pi}\,\mathrm{erfc}\,(\vartheta)\, e^{\vartheta^2} \vartheta\right) d\theta, \tag{22}$$

where

$$\vartheta = \frac{\lambda(\underline{x}_r^2 - \underline{x}_r \underline{x}_r')}{\sqrt{1+K}} - \sqrt{K}\cos(\theta) \tag{23}$$

and $\mathrm{erfc}(\vartheta) = 2/\sqrt{\pi} \int_{\vartheta}^{\infty} e^{-\gamma^2} d\gamma$ is the complementary error function. (22) must be evaluated numerically and substituted into (21) to yield the expression for the Chernoff factors of a Rayleigh/Rician fading channel with no side information available at the receiver. If we consider the limiting case of a Rayleigh channel with $K = 0$, the expectation in (22) can be evaluated in closed form to give

$$\underset{b}{E}\left[e^{-2\lambda b(\underline{x}_r^2 - \underline{x}_r \underline{x}_r')}\right] = 1 - \sqrt{\pi}\mathrm{erfc}(\lambda(\underline{x}_r^2 - \underline{x}_r \underline{x}_r'))e^{\lambda(\underline{x}_r^2 - \underline{x}_r \underline{x}_r')}(\lambda(\underline{x}_r^2 - \underline{x}_r \underline{x}_r')). \tag{24}$$

The Chernoff factors in (21) can then be used in (12) to calculate the cutoff-rate R_0 for the fading channel with no side information. Figure 4 shows R_0 curves for the fading channel without side information for several signal constellations. It is worth noting that, unlike the case of the AWGN-channel and the fading channel with side information, constant envelope schemes fare considerably better than rectangular constellations for the fading channel without side information. The reason for this superiority of constant envelope schemes is the fact that fading radially shrinks or expands the decision region boundaries. Because the decoder does not know the fading depth b_r, it cannot adjust these boundaries. In the case of constant envelope signal constellations, the decision boundaries are radially symmetric and therefore independent of the fading depth [6, chapter 5].

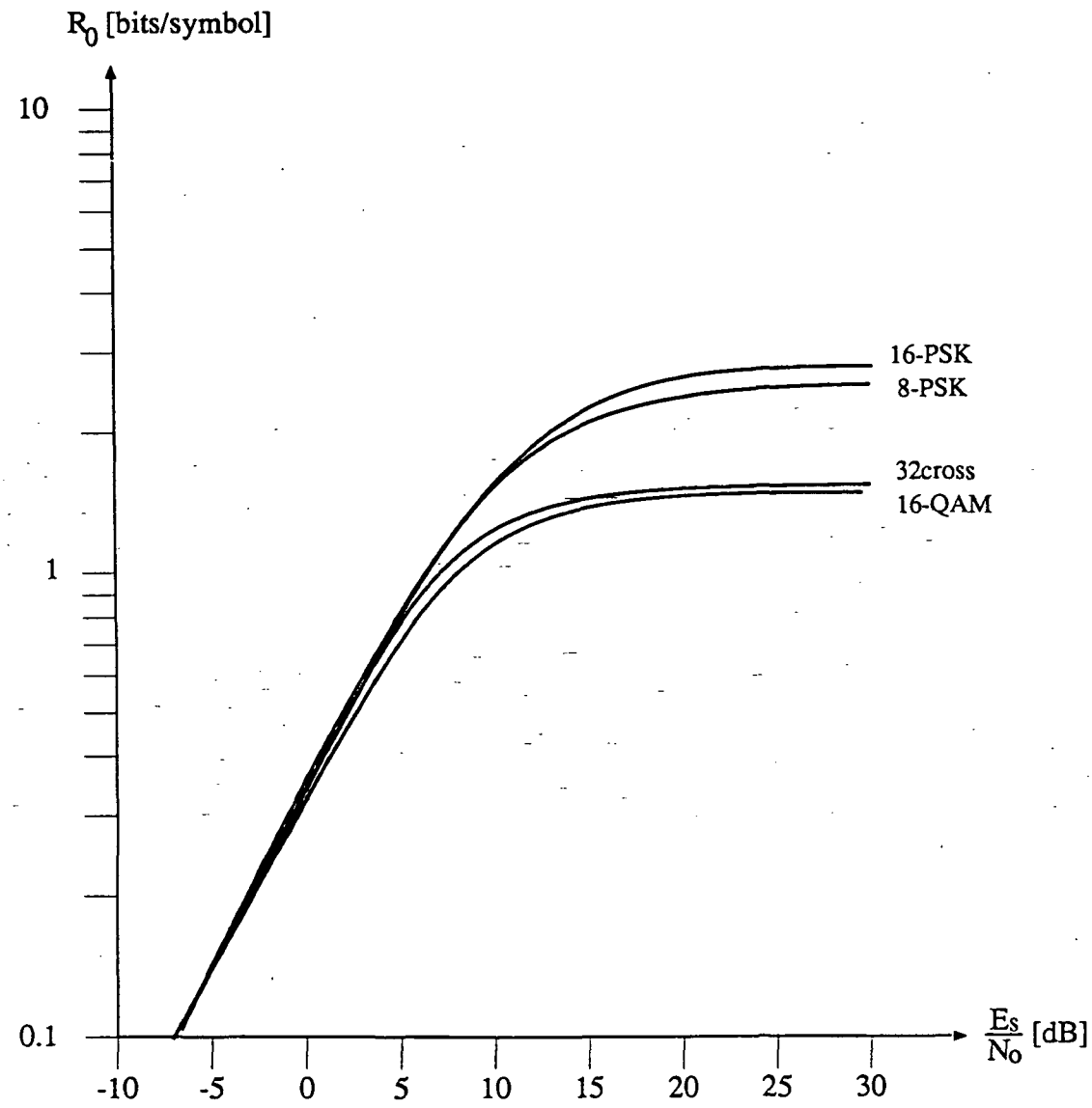


Figure 3: Cutoff-rates for the Rayleigh (K=0) fading channel without side information.

3 Coding Schemes

3.1 8-PSK Trellis Codes

A rate $R = k/n$ trellis code is generated by a binary convolutional encoder followed by a mapper as discussed in section 2. Figure 5 shows a rate 2/3 convolutional encoder and a mapping to 8-PSK signals without parallel transitions, i.e., $\tilde{k} = k$. The trellis codes for fading channels discussed in this paper are based on systematic convolutional codes of rate 2/3. The output bits of the binary encoder are mapped into the set of 8-PSK signals. This coded system transmits 2 bits/modulation signal, maintaining the same data rate as uncoded QPSK modulation.

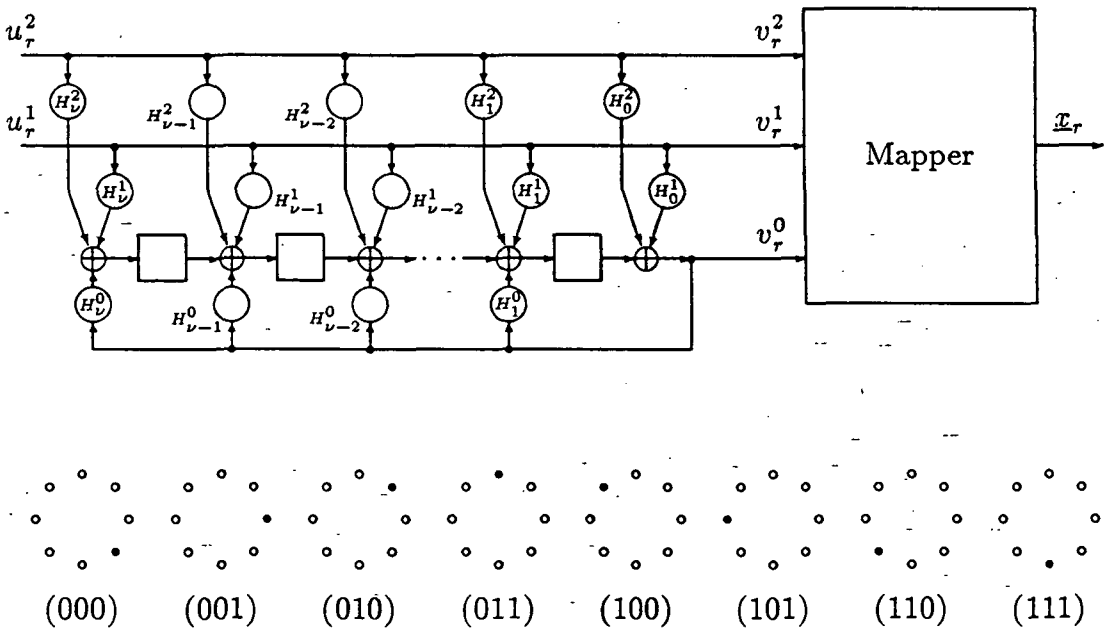


Figure 5: Rate 2/3 convolutional encoder with mapping from the binary output triples into 8-PSK signals.

Using the delay operator D, we may express the binary input sequences $u_0^1, u_1^1, u_2^1, \ldots$ and $u_0^2, u_1^2, u_2^2, \ldots$ as polynomials in D, i.e., $u^1(D) = u_0^1 + u_1^2 D + u_2^1 D^2 + \cdots$ and $u^2(D) = u_0^2 + u_1^2 D + u_2^2 D^2 + \cdots$. Similarly, the encoder connections may be expressed as polynomials in D such that $H^0(D) = H_0^0 + H_1^0 D + \ldots + H_\nu^0 D^\nu$, $H^1(D) = H_0^1 + H_1^1 D + \ldots + H_\nu^1 D^\nu$,

and $H^2(D) = H_0^2 + H_1^2 D + \ldots + H_\nu^2 D^\nu$, where ν is the encoder memory and $H_0^0 = 1$. The encoding operation can then be expressed in matrix notation as

$$[v^2(D), v^1(D), v^0(D)] = [u^2(D), u^1(D)] \left[I_2 \left| \begin{array}{c} H^2(D)/H^0(D) \\ H^1(D)/H^0(D) \end{array} \right. \right], \tag{25}$$

where $v^0(D)$, $v^1(D)$, and $v^2(D)$ are the polynomials of the three binary output sequences entering the mapper and I_2 is the 2×2 identity matrix.

We will usually give the encoder polynomials $H^0(D)$, $H^1(D)$, and $H^2(D)$ in octal form, i.e., the code $H^0(D) = 23\,(10011), H^1(D) = 04\,(00100), H^2(D) = 16\,(01110)$ means $H^0(D) = D^4 + D + 1, H^1(D) = D^2$, and $H^2(D) = D^3 + D^2 + D$.

3.2 Transfer Function Bound

In this section we develop the transfer function bound on the performance of trellis codes over fading channels. This will lead to a design criterion for good codes. Although the bit error probability P_b is the quantity of ultimate interest, a closely related and more readily determined quantity, the *event error probability* P_e, will be used to characterize the performance of trellis codes.

If $\mathbf{x}$ and $\mathbf{x}'$ are two symbol sequences corresponding to two paths through the trellis which are distinct for l branches starting at node j, and the decoder chooses the encoded sequence $\mathbf{x}'$ over the correct sequence $\mathbf{x}$, this is called an error event of length l starting at node j. An error event starts where the two paths diverge and ends where the two paths remerge. A union bound on P_e for a trellis code may be obtained by summing the probabilities of the error events of all possible lengths given a particular correct sequence $\mathbf{x}$ and averaging this quantity over all possible correct sequences $\mathbf{x}$.

With each incorrect path we may associate a sequence of incorrect trellis states S_r', while the sequence of correct states is denoted S_r. Any error event of length l can then be described by l state pairs, $(S_0, S_0'), \ldots, (S_l, S_l')$, with $S_0 = S_0'$, $S_l = S_l'$, and $S_r \neq S_r'$ for $0 < r < l$, i.e., the incorrect path must not touch the correct path during the error event. Associated with these paths are the two symbol sequences $\mathbf{x} = (\underline{x}_0, \underline{x}_1, \ldots, \underline{x}_l)$ and $\mathbf{x}' = (\underline{x}_0', \underline{x}_1', \ldots, \underline{x}_l')$, where $\underline{x}_r, \underline{x}_r' \in \mathcal{A}$. The probability of an error event may be upper bounded using the Chernoff factors (with $\lambda = \lambda_{R_0}$) between the individual signals of the two code sequences which the paths generate. We may therefore write

$$\Pr[(S_0,\ldots,S_l)\rightarrow(S'_0,\ldots,S'_l)]\leq C(\mathbf{x},\mathbf{x}')=\prod_{r=1}^{l}C(\underline{x}_r,\underline{x}'_r). \tag{26}$$

We now introduce the transfer function matrix $\mathbf{T}$ as the $2^{2\nu}\times 2^{2\nu}$ matrix whose rows and columns are labeled with state pairs (S_r,S'_r) and whose elements at the intersection of row (S_i,S_j) and column (S_k,S_l) are given by

$$t_{ij,kl}=\begin{cases}\sum_{\underline{x}_{ik}}\frac{1}{m}\sum_{\underline{x}'_{jl}}C(\underline{x}_{ik},\underline{x}'_{jl}) & \\ 0 & \text{if the transition } S_i\rightarrow S_k \\ & \text{or } S'_j\rightarrow S'_l \text{ does not exist,}\end{cases} \tag{27}$$

where $\underline{x}_{ik}$ is the signal the encoder transmits when it changes from state S_i to S_k and $\underline{x}'_{jl}$ is the signal on the trellis branch connecting state S'_j to S'_l. $t_{ij,kl}$ represents the Chernoff factors averaged over a pair of branches corresponding to the state pairs (S_i,S_k) and (S'_j,S'_l). Note that the sum over $\underline{x}'_{jl}$ is over all parallel transitions leading from state S'_j to state S'_l, the sum over $\underline{x}_{ik}$ is over all parallel transitions that lead the encoder form state S_i to S_k, and $m=2^{k-\bar{k}}$ is the number of parallel transitions. The factor $1/m$ reflects the fact that one of these transitions is selected with probability $1/m$ by the transmitter following the correct path.

Rearranging the state pairs in the matrix $\mathbf{T}$, we can write

$$\mathbf{T}=\begin{bmatrix}T_{CC} & T_{CI}\\ T_{IC} & T_{II}\end{bmatrix}, \tag{28}$$

where C is the set of state pairs such that $S_r=S'_r$, called a correct state pair, and I is the set of state pairs such that $S_r\neq S'_r$, called an incorrect state pair. The $2^{\nu}\times 2^{\nu}$ submatrix T_{CC} then contains all branch pairs in the trellis that diverge from a particular state and immediately remerge again, i.e., all the parallel transitions. T_{CI} and T_{IC} represent diverging and remerging branch pairs, respectively, while T_{II} represents all branch pairs that do not touch at either end. All error events start in a correct state pair and return to a correct state pair. An error event may thus occur in two distinct ways:

- One step error events consist of immediate transitions from one correct state pair to another. These are the parallel transitions in the trellis and are given by the submatrix T_{CC}.

- Error events extending over more than one branch are given by state pair sequences starting in the subset C and returning to it via one or more visits to the subset I, i.e., in matrix notation,

$$\begin{aligned} & T_{CI}T_{IC} + T_{CI}T_{II}T_{IC} + T_{CI}T_{II}^2T_{IC} + \ldots \\ \text{or} \quad & T_{CI}\left[\mathbf{I} + T_{II} + T_{II}^2 + \ldots\right]T_{IC} \\ \text{or} \quad & T_{CI}\left[\mathbf{I} - T_{II}\right]^{-1}T_{IC}, \end{aligned} \tag{29}$$

where $\mathbf{I}$ denotes the $(2^{2\nu} - 2^{\nu}) \times (2^{2\nu} - 2^{\nu})$ identity matrix. The event error probability may now be upper bounded by the transfer function bound, i.e.,

$$P_e \leq \frac{1}{2^{\nu}} \underline{1}^T \left\{T_{CC} + T_{CI}\left[\mathbf{I} - T_{II}\right]^{-1} T_{IC}\right\} \underline{1}, \tag{30}$$

where $\underline{1}$ is the 2^{ν}-dimensional all-one vector. The post multiplication by $\underline{1}$ represents the union of the error events from one state to any other state, while the premultiplication by $\underline{1}^T$ sums all 2^{ν} starting states, each one of which is assumed to have probability $1/2^{\nu}$.

Equation (30) can be written in the following form:

$$P_e \leq \sum_t A_t P_t, \tag{31}$$

where the sum is over all code sequence pairs $\mathbf{x}, \mathbf{x}'$ whose two codeword error probability has a specific value of the Chernoff bound $P_t = \prod_{r=1}^{l} C(\underline{x}_r, \underline{x}'_r)$ and A_t is the average number of code sequence pairs with Chernoff bound P_t.

3.3 Effective Length

In order to gain insight into the problem, we will approximate the expression in (30). In the submatrix T_{CC}, one term will usually be dominant. This is the nearest neighbor to the correct path among all the parallel transitions. Let this dominant term be P_C, i.e.,

$$P_C = C(\underline{x}_r, \underline{x}'_r), \tag{32}$$

where the signal pair $\underline{x}_r, \underline{x}'_r$ is the one with the smallest value of the Chernoff factor. Similarly, the second term inside the bracket of (30) will have a dominant term, which

we denote by P_I. This is an error event of length $l > 1$, that extends over more than one branch. According to (26), P_I can be written as

$$P_I = \prod_{r=1}^{l} C(\underline{x}_r, \underline{x}'_r). \tag{33}$$

We now look at two particular cases, the additive white Gaussian noise channel and the Rayleigh fading channel with side information. For the Gaussian channel

$$P_C = C(\underline{x}_r, \underline{x}'_r) = e^{-\frac{(\underline{x}_r - \underline{x}'_r)^2}{4N_0}} = e^{-\frac{\Delta^2}{4N_0}}, \tag{34}$$

where the dominant signal pair $(\underline{x}_r, \underline{x}'_r)$ is the one with the smallest value of $(\underline{x}_r - \underline{x}'_r)^2$, i.e., the one with the smallest squared Euclidean distance, denoted by Δ^2. Similarly P_I may be evaluated as

$$P_I = \prod_{r=1}^{l} C(\underline{x}_r, \underline{x}'_r) = e^{-\frac{1}{4N_0}\sum_{r=1}^{l}(\underline{x}_r - \underline{x}'_r)^2} = e^{-\frac{d_f^2}{4N_0}}, \tag{35}$$

where the dominant path pair is the one with the smallest cumulative squared Euclidean distance, denoted by d_f^2, the minimum free squared Euclidean distance of the code. In the Gaussian case it is unimportant over how many branches this minimum free squared Euclidean distance is accumulated, i.e., the right hand side of (35) is independent of l.

For the Rayleigh fading channel with $K = 0$, we obtain the following expression for P_C from (19):

$$P_C = C(\underline{x}_r, \underline{x}'_r) = \frac{1}{1 + \frac{(\underline{x}_r - \underline{x}'_r)^2}{4N_0}} \approx \frac{4N_0}{(\underline{x}_r - \underline{x}'_r)^2}, \tag{36}$$

where the approximation is tight for $E_S/N_0 > 6dB$, which is usually the case for transmission over a fading channel. Hence in the Rayleigh fading case the dominant term for parallel transitions is also the one with the smallest squared Euclidean distance Δ^2 between two signals. P_I on the other hand may be evaluated as

$$\begin{aligned} P_I &= \prod_{r=1}^{l} C(\underline{x}_r, \underline{x}'_r) \\ &= \prod_{r=1}^{l} \frac{1}{1 + \frac{(\underline{x}_r - \underline{x}'_r)^2}{4N_0}} \approx \frac{(4N_0)^{l'}}{\prod_{r=1}^{l}{}_{\underline{x}_r \neq \underline{x}'_r} (\underline{x}_r - \underline{x}'_r)^2}, \end{aligned} \tag{37}$$

where l' equals l, the length of the path pair, less the number of *matches*, i.e., the number of branches where $\underline{x}_r = \underline{x}'_r$. We call l' the *effective length* of the path pair and $l'_m = \min(l')$ the *effective length* of the code, where the minimum is taken over all path pairs. The approximation is again tight for $E_S/N_0 > 6dB$. Equation (37) is dominated by the paths with the shortest effective length, i.e., $l' = l'_m$, and among those by the one having the smallest product in the denominator, i.e., the one with the smallest squared *product distance* $d_p^2 = \prod_{\substack{r=1 \\ \underline{x}_r \neq \underline{x}'_r}}^{l} (\underline{x}_r - \underline{x}'_r)^2$. Hence the total event error probability in the fading case can be approximated by

$$P_e \approx k_1 \frac{4N_0}{\Delta^2} + k_2 \frac{(4N_0)^{l'_m}}{d_p^2}, \tag{38}$$

where k_1 is the average number of parallel transitions with the smallest squared Euclidean distance Δ^2 and k_2 is the average number of trellis path pairs with the smallest squared product distance d_p^2. It is clear from (38) that parallel transitions are most harmful in the Rayleigh case and should be avoided, unless Δ^2 is considerably larger than d_p^2 and the code is used at a low SNR E_S/N_0. In that case it is questionable if the necessary synchronization can be maintained in order to decode any but the most simple constellations. In Table 1 we list the effective length (l'_m) and the *minimum squared product distance* (d_p^2), i.e., the smallest squared product distance of those path pairs with $l' = l'_m$, for the set of rate 2/3 trellis codes designed by Ungerboeck [11], [12] for the AWGN-channel.

i.d.	ν	$H^0(D)$	$H^1(D)$	$H^2(D)$	l'_m	d_p^2
g2	2	5	2	-	1	4
g3	3	11	02	04	2	8
g4	4	23	04	16	3	4.68
g5	5	45	16	34	2	8
g6	6	103	030	066	3	16
g7	7	277	054	122	4	2.75
g8	8	435	072	130	3	16
g9	9	1007	164	260	3	16
g10	10	2003	164	770	4	32

Table 1: Ungerboeck's 8-PSK codes.

These codes use an 8-PSK signal set and are among the best codes known for the AWGN-channel. However they suffer a significant performance degradation on the Rayleigh

fading channel due to their small l'_m and the slow increase in l'_m with code complexity, which explains their poor performance discussed in section 4.

Table 2 shows a list of 8-PSK codes designed for fading channels, i.e., designed for a large effective length l'_m. The codes were found using either an exhaustive search or one of the construction methods presented in [13] and [14].

i.d.	ν	$H^0(D)$	$H^1(D)$	$H^2(D)$	l'_m	d_p^2
f2	2	5	2	-	1	4
f3	3	11	02	04	2	8
f4	4	23	04	16	3	4.68
f5	5	43	14	36	3	16
f6	6	103	036	154	4	8
f7	7	223	076	314	4	8
f8	8	673	336	164	5	5.49
f9	9	1413	756	244	5	18.75
f10	10	3303	1676	504	5	128
f11	11	6403	3436	1264	6	10.98

Table 2: Codes designed for fading channels.

Note that the codes in Table 2 have, with the exception of the very short codes f2=g2, f3=g3, and f4=g4, a larger effective length l'_m, which should give them superior performance on fading channels, especially at high values of E_S/N_0. The Gaussian codes designed for a large free squared Euclidean distance d_f^2 achieve this distance over only a few branches in most cases. This proves detrimental on fading channels, where the distance should be spread more evenly over all the branches of a trellis path pair.

3.4 Binary Signaling

Due to the synchronization problem, binary signaling is usually used on fading channels. Let us assume for the sake of discussion that the trellis is generated by a rate $R = k/n$ binary convolutional encoder. Instead of the two dimensional multilevel/phase signals $\underline{x}_r$ on the branches of the trellis in the case of bandwidth efficient coding, n binary signals are used. These signals are antipodal if coherent reception is possible (BPSK-signaling) and orthogonal otherwise. We also assume that the binary signals are interleaved to make the channel memoryless. The Chernoff factor for the two signals $\underline{x}_r$ and $\underline{x}'_r$ is then given by

$$C(\underline{x}_r, \underline{x}'_r) = \prod_{i=1}^{n} C(x_{ri}, x'_{ri}), \tag{39}$$

where x_{ri} is the i-th bit on the r-th branch of $\mathbf{x}$ and the path error probability bound of (26) becomes

$$\Pr[(S_0, \ldots, S_l) \to (S'_0, \ldots, S'_l)] \leq \prod_{r=1}^{l} \prod_{i=1}^{n} C(x_{ri}, x'_{ri}) = \prod_{k=1}^{nl} C(x_k, x'_k), \tag{40}$$

where x_k and x'_k, $1 \leq k \leq nl$, are the two binary signal sequences associated with the error event $(S_0, S'_0), \ldots, (S_l, S'_l)$. Substituting the Chernoff factors for binary signals, we obtain

$$\begin{aligned} \Pr[(S_0, \ldots, S_l) \to (S'_0, \ldots, S'_l)] &\leq \prod_{k=1}^{nl} \frac{1}{1 + \frac{(x_k - x'_k)^2}{4N_0}} \\ &\approx \prod_{\substack{k=1 \\ x_k \neq x'_k}}^{nl} \frac{4N_0}{(x_k - x'_k)^2}. \end{aligned} \tag{41}$$

Since $(x_k - x'_k)^2$ equals 0 or Δ^2, depending on whether $x_k = x'_k$ or $x_k \neq x'_k$, we can simplify (41) to obtain

$$\begin{aligned} \Pr[(S_0, \ldots, S_l) \to (S'_0, \ldots, S'_l)] &\approx \prod_{\substack{k=1 \\ x_k \neq x'_k}}^{nl} \frac{4N_0}{\Delta^2} \\ &= \left(\frac{4N_0}{\Delta^2}\right)^{d(\mathbf{x}, \mathbf{x}')}, \end{aligned} \tag{42}$$

where $d(\mathbf{x}, \mathbf{x}')$ denotes the Hamming distance between the two binary sequences $\mathbf{x}$ and $\mathbf{x}'$ that make up the two paths in question. Equation (42) shows that the well known Hamming distance remains the design criterion for binary signaling on the fading channel. It is worth noting that a 3dB coding gain may be achieved if phase synchronization is possible, since the binary signals may then be chosen to be antipodal, i.e., $\Delta^2 = 4E_S$, where E_S is the signal energy. If phase synchronization is not possible, the signals must be orthogonal, giving $\Delta^2 = 2E_S$, reflecting a 3dB loss. For a more extensive discussion on binary signaling for the fading channel see [6] and [16].

3.5 Rician Fading Channels

The Rayleigh fading channel is the limiting case of a more general channel, the Rician fading channel. In the Rician case the expression for P_C is, from (19),

$$P_C = C(\underline{x}_r, \underline{x}'_r) = \frac{1+K}{1+K+\frac{(\underline{x}_r-\underline{x}'_r)^2}{4N_0}} e^{-\frac{K\frac{(\underline{x}_r-\underline{x}'_r)^2}{4N_0}}{1+K+\frac{(\underline{x}_r-\underline{x}'_r)^2}{4N_0}}}. \tag{43}$$

P_I on the other hand is given by

$$P_I = \prod_{r=1}^{l} C(\underline{x}_r, \underline{x}'_r) = e^{-\sum_{r=1}^{l} \frac{K\frac{(\underline{x}_r-\underline{x}'_r)^2}{4N_0}}{1+K+\frac{(\underline{x}_r-\underline{x}'_r)^2}{4N_0}}} \prod_{r=1}^{l} \frac{1+K}{1+K+\frac{(\underline{x}_r-\underline{x}'_r)^2}{4N_0}}. \tag{44}$$

Comparing (43) and (44), it becomes obvious that the performance degradation of the parallel transitions becomes less and less severe as the energy E_d received on the direct path increases with respect to the energy E_m received via the diffuse multipaths, i.e., with growing K. Hence for Rician fading channels with strong line of sight reception, parallel transitions become feasible. For small values of the Rice factor K, i.e., $K < E_S/N_0$, (43) and (44) can be approximated as

$$P_C = C(\underline{x}_r, \underline{x}'_r) \approx \frac{4N_0(1+K)e^{-K}}{(\underline{x}_r - \underline{x}'_r)^2}, \tag{45}$$

$$P_I = \prod_{r=1}^{l} C(\underline{x}_r, \underline{x}'_r) \approx \frac{(4N_0(1+K)e^{-K})^{l'}}{\prod_{\substack{r=1 \\ \underline{x}_r \neq \underline{x}_r}}^{l} (\underline{x}_r - \underline{x}'_r)^2}. \tag{46}$$

It becomes evident then that for small values of the Rice factor K, the effective length l'_m is once again the dominant code design criterion.

3.6 Distance Spectrum of Fading Channel Codes

Using the approximations (37) or (46) for the Chernoff factors $C(\underline{x}_r, \underline{x}'_r)$, the expression for the transfer function bound (31) can be written in the form

$$P_e \leq \sum_{l',d'} A_{l',d'} P_{l',d'}, \tag{47}$$

where $A_{l',d'}$ is the average number of code sequence pairs $\mathbf{x}, \mathbf{x}'$ with effective length l' and product distance d', $P_{l',d'}$ is the event error probability bound for an event error with effective length l' and product distance d', and the average is taken over all code sequences $\mathbf{x}$ in the code. The parameter $A_{l',d'}$ is called the average *multiplicity* of all code sequence pairs with event error probability bound $P_{l',d'}$. Note that in (47) the error bound $P_{l',d'}$ includes both terms of the form (32) (parallel transition error events) and terms of the form (33) (length $l > 1$ error events).

For fading channels a *spectral line* is defined by an effective length l', a product distance d', and an average multiplicity $A_{l',d'}$. The set of all spectral lines of a code is called the *distance spectrum* of that code. In [15], we present an algorithm that computes the distance spectrum for *quasi-regular* codes, a general class of codes to which all the best known trellis codes belong. This algorithm has been adapted to compute the distance spectrum of codes for fading channels. Figure 6 shows the distance spectrum of the 16-state 8-PSK from Table 2 (code f4), where the factors in the product distances d' have been normalized by the square root of the signal energy $\sqrt{E_S}$. The effective length of this code is $l'_m = 3$, its minimum squared product distance is $d_p^2 = 4.68$, and its multiplicity is $A_{3,4.68} = 2$. Note that Figure 6 is divided into different diagrams for different effective lengths l'. The distance spectrum is used in section 4 to evaluate the performance of codes on fading channels.

3.7 Block Code Performance on a Quantized Fading Channel

In this section we discuss the use of block codes as an alternative to trellis codes on fading channels. The received 2-dimensional signal $\underline{y}$ is mapped into a discrete output alphabet, as illustrated in Figure 7 for an 8-PSK signal set. For MPSK signaling, each received signal $\underline{y}$ is decoded into one of M angular sections, called decision regions, and denoted by Λ_i. Because of the high probability of a deep fade, i.e., the reception of a signal with a very low amplitude b, we have introduced a circular erasure region with radius ρ. This is done with the idea of using Reed-Solomon block codes which can handle erasures. In conjunction with interleaving, this transforms the fading channel into a discrete memoryless erasure channel.

Let q_{ij} be the conditional probability that the received signal $\underline{y}$ lies in Λ_j, given that the signal $\underline{x}_i$ was sent, and let q_e be the probability that $\underline{y}$ falls into Λ_e, i.e.,

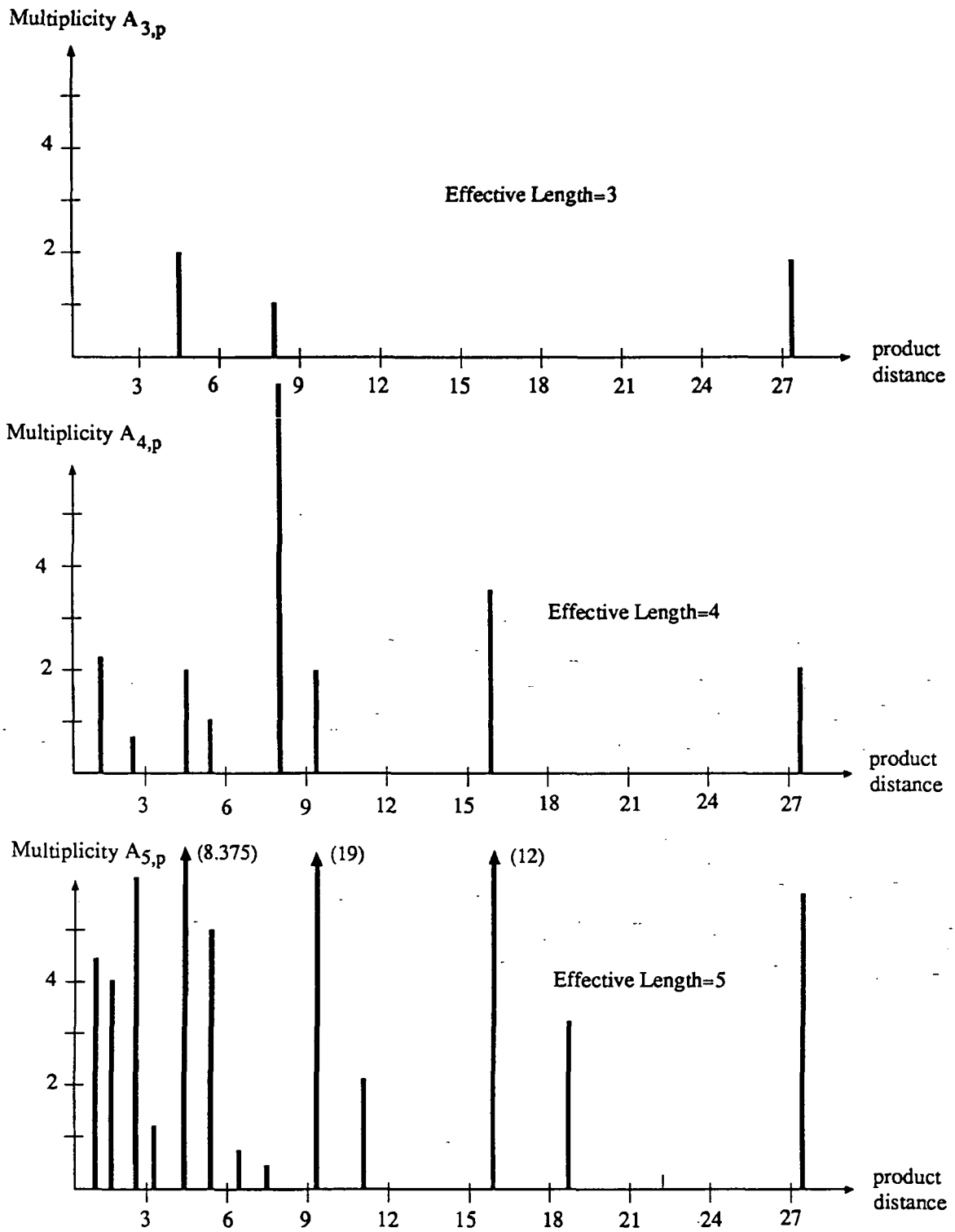


Figure 6: Fading distance spectrum of coded 8-PSK modulation (Code f4).

$$q_{ij} = \int_{\Lambda_i} Pr(\underline{y} \mid \underline{x}_i) d\underline{y} = \frac{1}{\pi N_0} \int_{\Lambda_i} e^{-\frac{y^2}{N_0}} \underset{b}{E} \left\{ e^{2\frac{y x_i}{N_0} b} e^{-b^2 \frac{x_i^2}{N_0}} \right\} d\underline{y} \tag{48}$$

and

$$q_e = 1 - \sum_j q_{ij}. \tag{49}$$

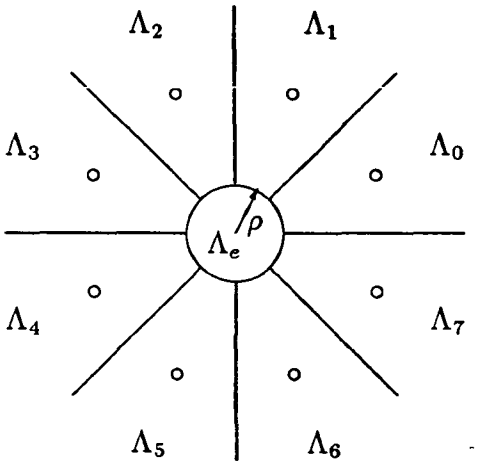


Figure 7: Decision regions for an 8-PSK signal set with an erasure region.

As an example, we will discuss the performance of an RS(63,42) block code using this quantized 8-PSK channel. The symbols of the RS(63,42) block code are 6 bits long, i.e., two concatenated 8-PSK signals. This code has a symbol rate $k/n = 42/63$, which translates into a bit rate of 2 bits/signal, i.e., the same transmission rate as uncoded QPSK, the reference transmission system. The receiver operates such that if any one of the two received vectors $\underline{y}_1, \underline{y}_2$ belonging to the same RS-symbol is decoded into Δ_e, the whole RS-symbol is erased.

Using the Berlekemp-Massey decoding algorithm, the RS-decoder can correct any combination of t errors and e erasures as long as $e + 2t \leq n - k$, where $n - k$ is the number of parity symbols. For this particular code, the symbol erasure probability P_Δ is given by $P_\Delta = q_e q_e + 2q_e(1-q_e)$, the probability of receiving a correct symbol is $P_c = q_{00} q_{00}$, and the probability of receiving a symbol in error is given by $P_i = 1 - P_c - P_\Delta$.

The block error probability P_B of such an RS-code may then be computed as

$$P_B = \sum_{t=0}^{\lfloor \frac{n-k}{2} \rfloor} \binom{n}{t} P_i^t \sum_{e=n-k+1-2t}^{n-t} \binom{n-t}{e} P_\Delta^e P_c^{n-t-e} + \sum_{t=\lfloor \frac{n-k}{2} \rfloor + 1}^{n} \binom{n}{t} P_i^t (1-P_i)^{n-t}. \quad (50)$$

Figure 8 compares the performance of this bandwidth efficient RS-block code to uncoded QPSK. The erasure threshold radius ρ has been optimized for each value of E_S/N_0.

4 Comparison of Code Performance

In this section we present performance curves on the event error probability P_e of several TCM schemes. The 8-PSK trellis codes introduced in Section 3 are *quasi-regular* [3], and we have used a variant of the algorithm reported in [15] to evaluate P_e. The two code word error probability bound in (10) is given by

$$P(\mathbf{x} \to \mathbf{x}') \leq \min_{\lambda} C(\mathbf{x}, \mathbf{x}', \lambda) = \min_{\lambda} \prod_{r=1}^{l} C(\underline{x}_r, \underline{x}'_r, \lambda), \quad (51)$$

where the tightest bound is obtained by individually minimizing (51) over λ for each path pair $\mathbf{x}, \mathbf{x}'$. In view of the large number of code sequence pairs $\mathbf{x}, \mathbf{x}'$, this is computationally unfeasible. However, any value of λ may be used in (51) to obtain a looser bound. We choose $\lambda = \lambda_{R_0}$, i.e., the value of λ which maximizes R_0, and the two code word bound of (51) becomes equivalent to the terms P_t in the event error probability bound of (31), i.e.,

$$P_e \leq \sum_t A_t P_t, \quad \text{where } P_t = \prod_{r=1}^{l} C(\underline{x}_r, \underline{x}'_r). \quad (52)$$

The algorithm in [15] is a stack type algorithm which successively searches code sequence pairs in increasing order of a metric associated with those code sequence pairs. This metric is additive over the individual signal pairs in the code sequences, and in [15] it is the squared Euclidean distance between those signal pairs. For $\lambda = \lambda_{R_0}$, if we write

[3] For a precise definition of quasi-regularity, the reader is referred to [15].

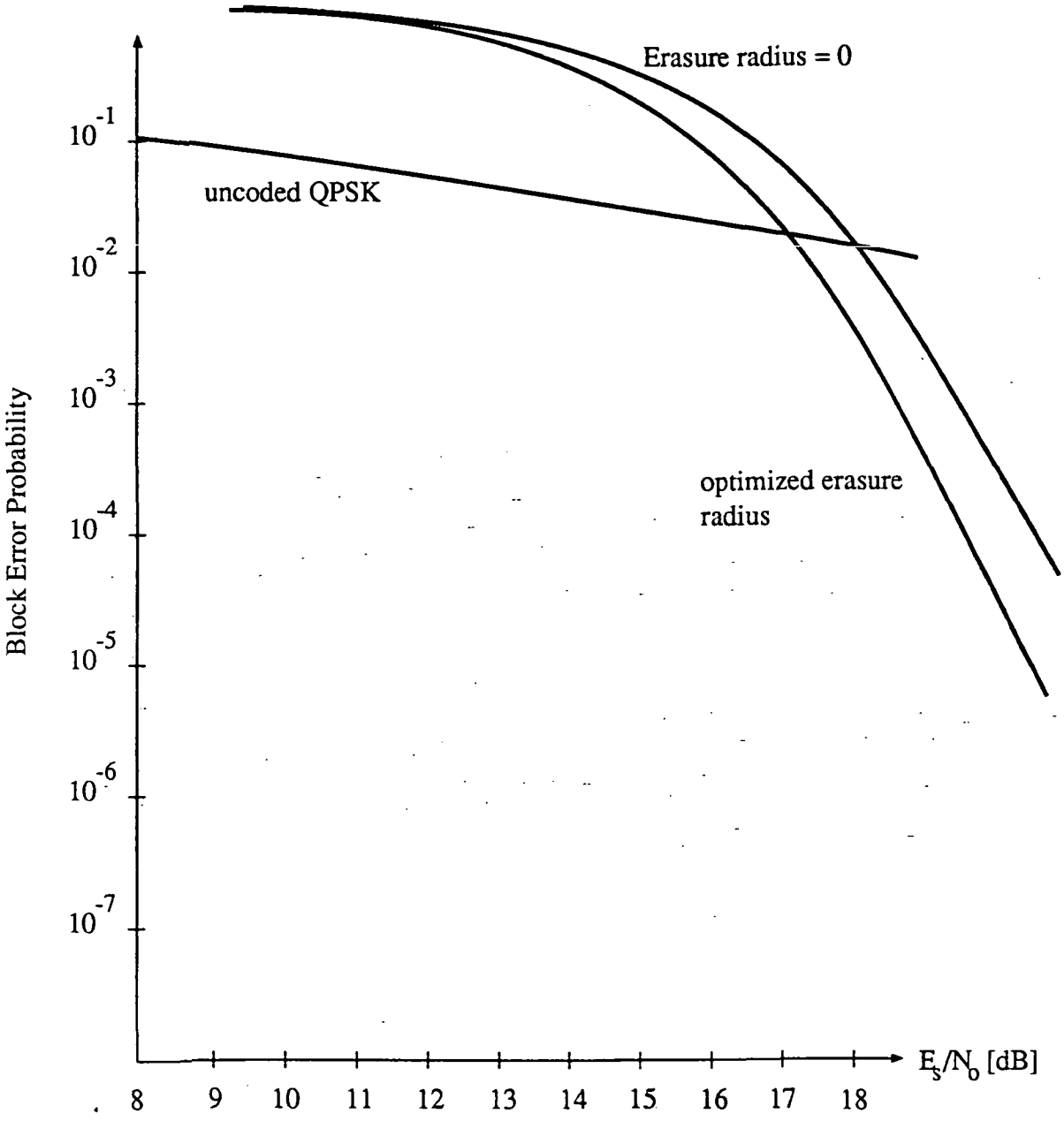


Figure 8: Comparison of an RS (63,42) block code with uncoded QPSK on a hard-quantized Rayleigh fading channel.

$$-\ln P_t = \sum_{r=1}^{l} -\ln C(\underline{x}_r, \underline{x}'_r) \tag{53}$$

and choose $-\ln C(\underline{x}_r, \underline{x}'_r)$ as the branch metric, the algorithm can be used in a slightly altered form to calculate the event error probability of any quasi-regular TCM scheme.

In general, $P_t = \prod_{r=1}^{l} C(\underline{x}_r, \underline{x}'_r)$ will depend on E_S/N_0 as well as on the specific signals $\underline{x}_r$ and $\underline{x}'_r$, $1 \leq r \leq l$, and we must run the algorithm separately for each value of the SNR. If, however, the Chernoff factors are accurately approximated by (37) or (46), where they depend only on E_S/N_0, l', and d' (fading channels with side information), the distance spectrum can be used to evaluate P_e for all values of E_S/N_0 and the algorithm is used only once per code to calculate the distance spectrum. In this case the distance spectrum gives a good measure of code performance. As shown in the following figures codes with a good distance spectrum usually also perform well on channels where the approximations (37) or (46) may not be tight (fading channels with no side information, Rician channels with a large Rice factor).

Figures 9 to 13 show the event error probability bound of some selected codes, where we have used (53) to evaluate P_e. Figure 9 shows the performance on a Rayleigh channel with side information. The superiority of the new codes presented in Table 2 is evident. At an error probability level of $P_e = 10^{-5}$, for example, the code f8 shows a 3.5dB improvement over the code g8. At $P_e = 10^{-6}$, the difference is 4.5dB. Note further that the asymptotic behavior of the Gaussian codes g8 and g6 are identical and it is therefore useless to employ the higher complexity of code g8.

Figure 10 shows the same set of codes on a Rician channel with $K = 7dB$, a typical value of the Rice factor. Again the fading codes from Table 2 outperform the Gaussian codes form Table 1 of the same complexity. The code f8, for example, achieves an error probability of $P_e = 10^{-6}$ at an E_S/N_0 of $12.2dB$, while the code g8 achieves the same error performance at $13.7dB$. At $P_e = 10^{-7}$, the gain of f8 over g8 is $2.5dB$.

Figure 11 shows the same set of codes on a $K = 15dB$ Rician channel. This value of K indicates strong line of sight reception. In this environment, the Gaussian codes fare slightly better due to their superior minimum free squared Euclidean distance. f8 and g8 have almost identical error performance, while f6 loses $1.2dB$ compared to g6.

Figures 12 and 13 show the same set of codes on a Rayleigh fading channel and a Rician channel with $K = 7dB$ and without side information. While the relative performance of the codes with respect to each other is preserved, the lack of side information causes

a rather severe degradation in performance, particularly for the Rayleigh channel. At an error probability of $P_e = 10^{-5}$, for instance, f8 gains $3dB$ over g8 on the Rayleigh channel and gains $2dB$ on the $K = 7dB$ Rician channel. It is fortunate that for the Rayleigh channel side information can be extracted relatively easily from the receiver by monitoring a pilot tone, or in the case of constant envelope signaling by simply estimating the received signal amplitude.

Comparing the performance of the RS-block code to TCM, we see that the RS-code is very poor for $E_S/N_0 < 20dB$. The error curves for the RS-code have a rather sharp cutoff and catch up with the trellis codes at an E_S/N_0 of about $30dB$ for the Rayleigh channel with side information. With no side information, the RS-code performance curve crosses the f8 performance curve at $E_S/N_0 \approx 25dB$. This behavior is typical when comparing the performance of block codes and trellis codes.

5 Conclusions

We have presented a general method of bounding the event error probability of TCM schemes and applied this method to the fading channel. We have shown that the effective length and the minimum squared product distance replace the minimum free squared Euclidean distance as a design criterion for Rayleigh fading channels and Rician fading channels with a substantial multipath component. We have presented codes specifically constructed for fading channels that outperform equivalent codes designed for the AWGN-channel. The use of RS-block codes with expanded signal sets becomes interesting only for large SNR's, where they begin to outperform trellis codes.

6 Acknowledgement

We wish to acknowledge the help of Mr. Marc Rouanne in preparing the program used to compute the distance spectrum of the codes presented in this paper.

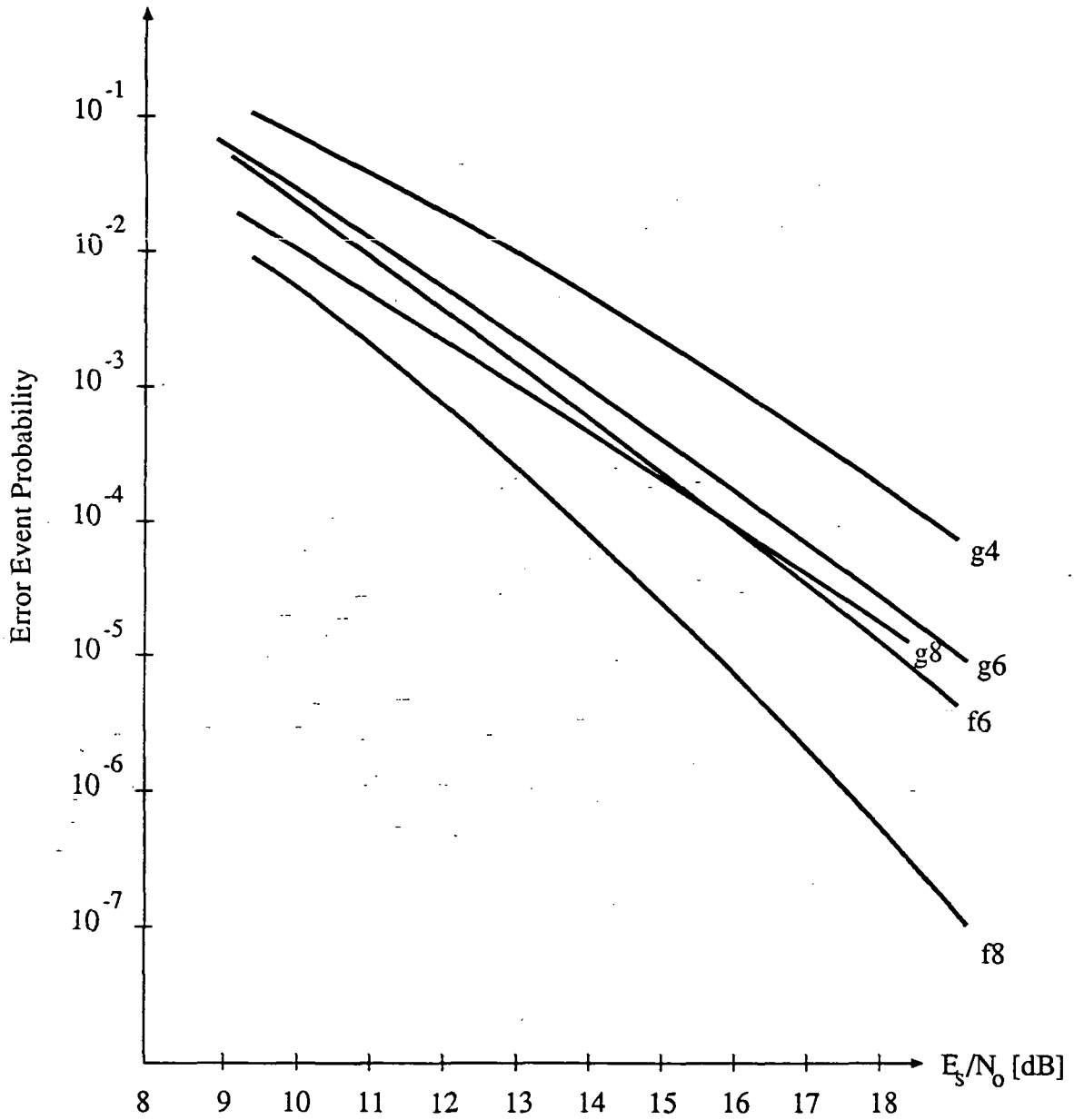


Figure 9: Performance of trellis codes on a Rayleigh channel with side information.

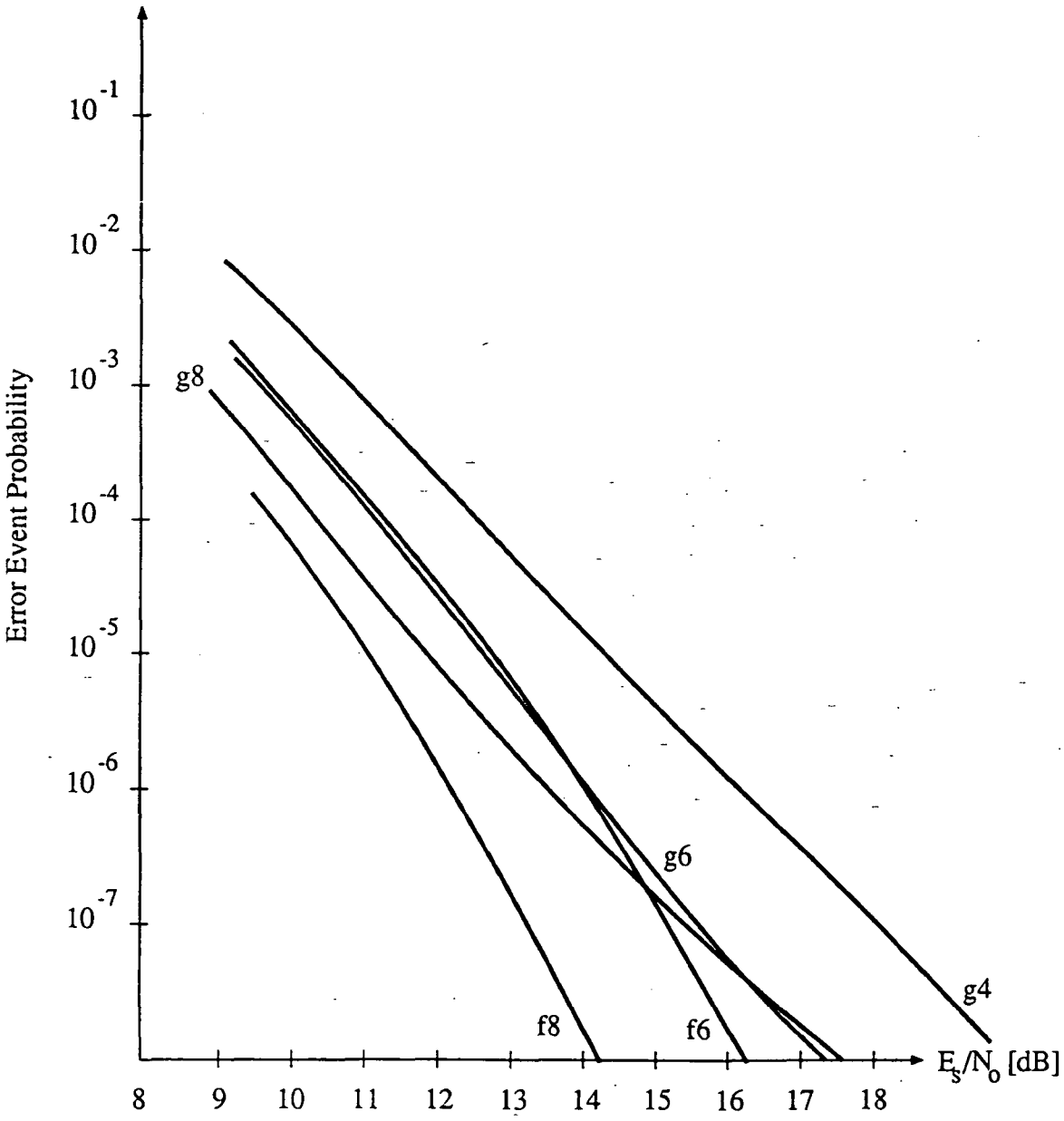


Figure 10: Performance of codes on a Rician channel with $K = 7dB$ and with side information.

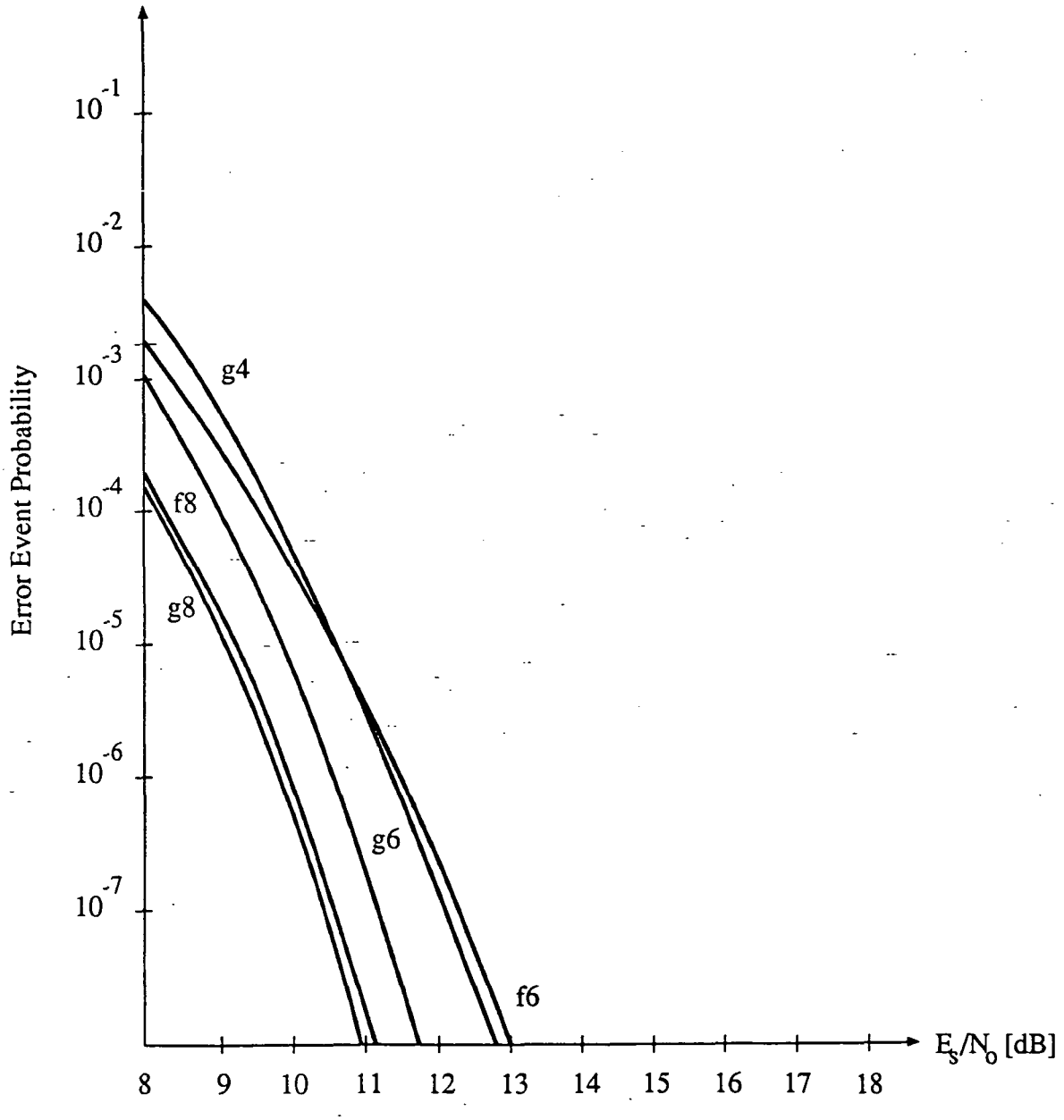


Figure 11: Performance of trellis codes on a Rician channel with $K = 15dB$ and with side information.

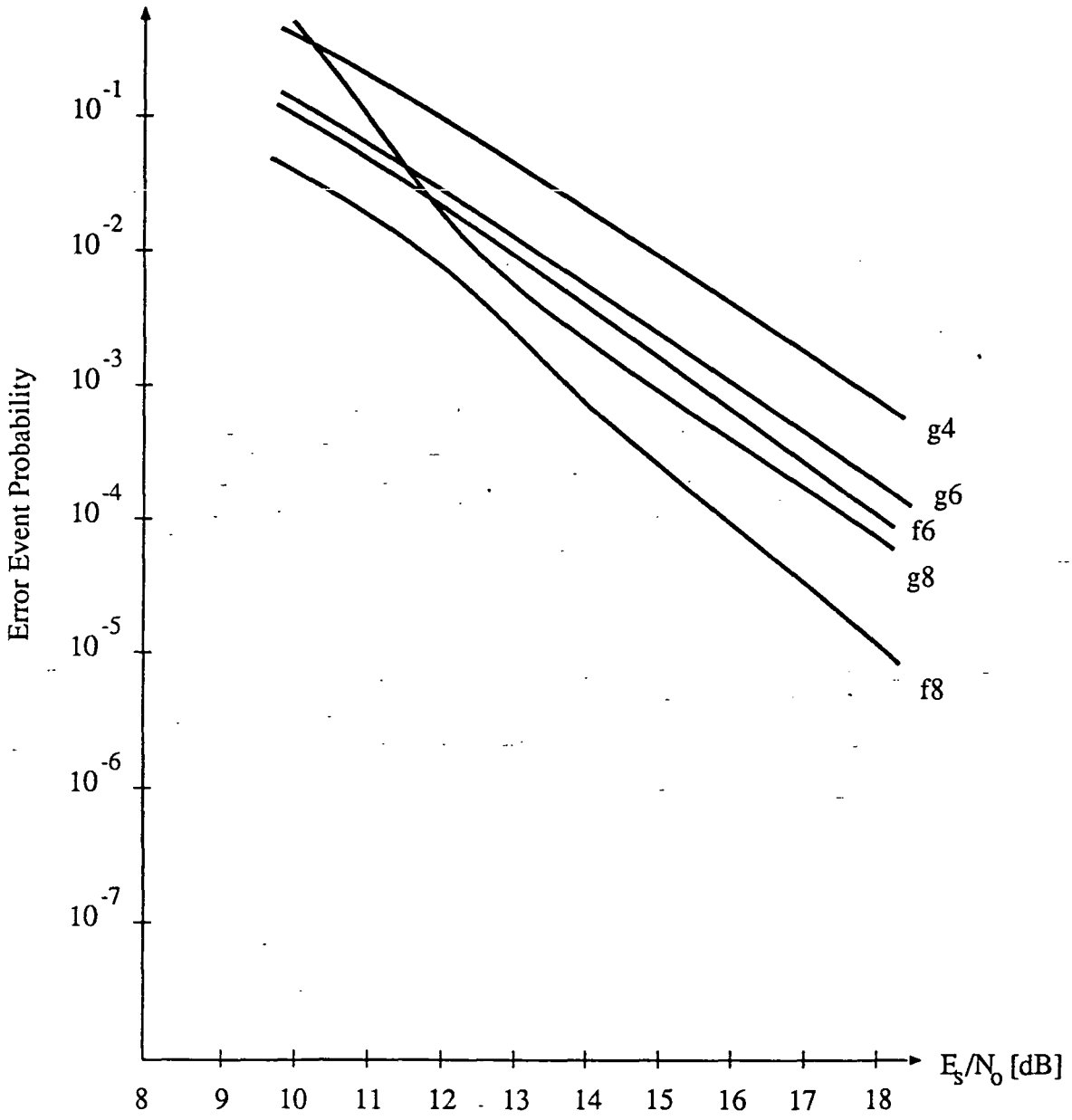


Figure 12: Performance of trellis codes on a Rayleigh channel with no side information.

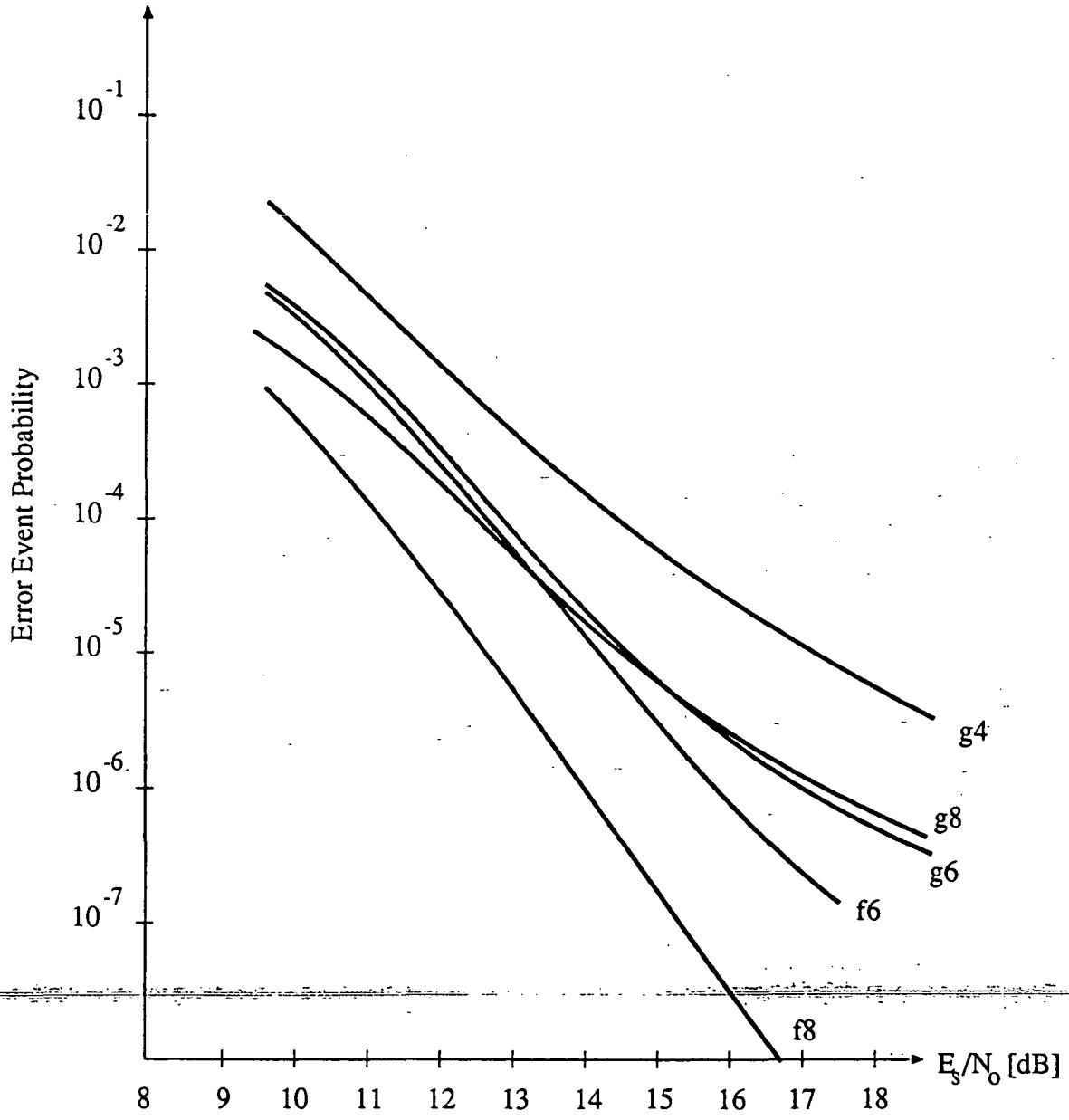


Figure 13: Performance of trellis codes on a Rician channel with $K = 7dB$ and without side information.

References

[1] A. Ghais et. al., "INMARSAT and the Future of Mobile Satellite Services", *IEEE Journal Selected Areas Commun.*, Vol. SAC-5, No.4, May 1987, pp. 592–600.

[2] J. Hagenauer et al., "The Maritime Satellite Communication Channel," *IEEE Journal Selected Areas Commun.*, Vol. SAC-5, No.4, 701–713, May 1987.

[3] J. Hagenauer and E. Lutz, "Forward Error Correction Coding for Fading Compensation in Mobile Satellite Channels," *IEEE Journal Selected Areas Commun.*, Vol. SAC-5, No.2, pp. 215–225, February 1987.

[4] M. K. Simon and D. Divsalar, "The Performance of Trellis Coded Multilevel DPSK on a Fading Mobile Satellite Channel", presented at the IEEE 1987 International Conference on Communications, Seattle, WA, June 7–10, 1987.

[5] M. K. Simon and D. Divsalar, "Trellis Coded Modulation for 4800-9600 bits/s Transmission Over a Fading Mobile Satellite Channel," *IEEE Journal Selected Areas Commun.*, Vol. SAC-5, No.2, pp. 162–177, February 1987.

[6] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering*, New York, Wiley, 1965.

[7] R. G. Gallager, *Information Theory and Reliable Communications*, New York, Wiley, 1968.

[8] R. S. Kennedy, *Fading Dispersive Communication Channels*, New York, Wiley, 1969.

[9] D. Divsalar and M. K. Simon, "The Design of Trellis Codes for Fading Channels", submitted for publication.

[10] C. Schlegel, "Bandwidth Efficient Coding for Non-Gaussian Channels", Ph. D. Dissertation, University of Notre Dame, Notre Dame, Indiana, 1988.

[11] G. Ungerboeck, "Channel Coding with Multilevel/Phase Signals," *IEEE Trans. Inform. Theory*, Vol. IT-28, No.1, pp. 55–67, January, 1982.

[12] G. Ungerboeck, "Trellis-Coded Modulation with Redundant Signal Sets Part II: State of the Art," *IEEE Communications Magazine*, Vol. 25, No.2, pp. 12–21, February, 1987.

[13] J. E. Porath and T. Aulin, "Fast Algorithmic Construction of Mostly Optimal Trellis Codes" Technical Report No. 5, Division of Information Theory, Chalmers University of Technology, Göteborg, Sweden, February 1987.

[14] M. Rouanne, "Distance Bounds and Construction Algorithms for Trellis Codes", Ph. D. Dissertation, University of Notre Dame, Notre Dame, Indiana, 1988.

[15] M. Rouanne, D. J. Costello, Jr. and C. Schlegel, "An Algorithm for Computing the Distance Spectrum of Trellis Codes", Proceedings 1988 Conference on Information Sciences and Systems, Princeton, New Jersey, March 16–18, 1988.

[16] I. M. Jacobs, "Probability-of-Error Bounds for Binary Transmission on the Slowly Fading Rician Channel", *IEEE Trans. Inform. Theory*, Vol. IT-12, No.4, October 1966, pp. 431–441.