Trellis Coded Modulation Techniques

By

Faramarz Shayan Arani, M.Sc.

Submitted for the Degree of Doctor of Philosophy

to the Higher Degrees Committee

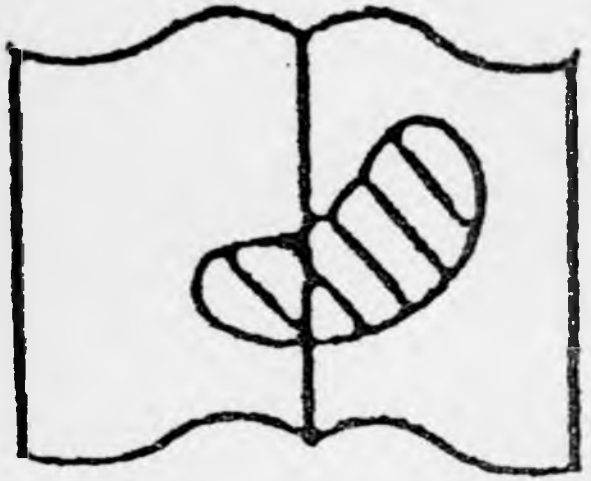
University of Warwick

Department of Engineering

University of Warwick.

August 1993

VARIABLE PRINT QUALITY



To

My Father and Brother;

My Wife Maryam and My Son Behrod.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank my supervisor Professor Bahram Honary for his professional guidance throughout this period of research; for the numerous discussions which lead to inspiring thoughts; for his patience and encouragement and for the opportunity to work in Hull-Lancaster Communications Research Group (HLCRG) (former Hull-Warwick).

I also like to express my appreciation to my colleagues in HLCRG for their friendship, help and support.

Finally, I must thank my mother by being my strength and silent inspiration throughout my life, my father, my brother and my wife Maryam for their love, support and constant encouragement.

Declaration

This thesis is presented in accordance with the regulations for the degree of Doctor of Philosophy by the Higher Degree Committee at the University of Warwick. The thesis has been composed and written based on the research undertaken by the author. The research materials have not been submitted in any previous application for a higher degree. All sources of information are specifically acknowledged in the content.

Abbreviations

A4PSK ..... Asymmetrical four Phase Shift Keying

ACS ..... Add Compare Select

ARQ ..... Automatic Repeat reQuest

AWGN ..... Additive White Gaussian Noise

BCH ..... Bose-Chauduri-Hocquenghen

BCM ..... Block Coded Modulation

BER ..... Bit Error Rate

BFSK ..... Binary Frequency Shift Keying

BPSK ..... Binary Phase Shift Keying

ED ..... Euclidean Distance

ETCM ..... Embedded Trellis Coded Modulation

MLD ..... Maximum Likelihood Detection

PSK ..... Phase Shift Keying

QPSK ..... Quadrature Phase Shift Keying

SNR ..... Signal to Noise Ratio

TCM ..... Trellis Coded Modulation

List of Symbols

$\lfloor x \rfloor$ ..... largest integer less than or equal to x

(n, k) ..... convolutional encoder (n coded bits, k message bits)

b_{ij} ..... trellis branch label

c ..... code vector

d ..... Euclidean distance

d_{min} ..... minimum Euclidean distance

d_{free} ..... free Euclidean distance

d_{ref} ..... reference free Euclidean distance

d^H_{min} ..... minimum Hamming distance

D_b ..... Bhattacharyya distance

D ..... Delay operator

E_s ..... symbol energy

E ..... signal energy

E_b ..... bit energy

g_i ..... generator polynomial

G ..... coding gain

I ..... Inphase signal component

$J(j)$ ..... set of indexes j

K ..... encoded bits

k ..... information bits

L ..... length of shift register (constraint length)

l_n ..... length of the n^{th} branch in trellis

L_i ..... search length

M ..... number of messages

m ..... messages (information)

$n(t)$ ..... noise waveform

N_0 ..... one-sided noise power spectral density

N_a ..... maximum number of stored nodes (states)

N_c ..... number of metric comparisions

N_s ..... number of states in trellis diagram

N_m ..... number of branch metric computations

n_j ..... noise signal component

n ..... number of linear algebraic function generators

N ..... dummy variable

P_Q ..... power in Q channel

P_I ..... power in I channel

$P(r/c)$ ..... conditional probability density function

P_E ..... probability of error

P_s ..... probability of symbol error

P_b ..... probability of bit error

$p_m(i.j.t)$ ..... path metric

$Q(.)$ ..... complementary error function

Q ..... quadrature component of signal

q ..... cardinality of a code symbol

R ..... transmission rate

R_c ..... code rate

D_{cd}^{t+b} ..... i^{th} metric (Euclidean distance) value at time 't+b', state 'cd'

$\mathcal{R}$ ..... metric value in terms of inphase and quadrature components

r ..... parity check vector

$r(t)$ ..... received signal

r_j ..... received signal component

$sm(j,t)$ ..... survivor metric

$S_m(t)$ ..... m distinct transmitted waveforms

s_k ..... transmitted symbol

$\hat{s}_k$ ..... received symbol

T ..... Symbol time duration

$T(D)$ ..... Transfer function

t_e ..... error correcting capability

v ..... number of channel symbols per information bit

$w(i,j)$ ..... valid sequence of transition in trellis diagram

y_i ..... outputs of encoder

x_n ..... input binary data

$X(c)$ ..... sequence of transmitted codeword

$Z(c)$ ..... path metric for codeword c

Z_q ..... set of modulo q integers

α ..... power ratio in I and Q channels

φ_k ..... k orthonormal waveforms

Ω ..... set of channel symbols

ϕ ..... constellation angle

σ^2 ..... noise variance

Abstract

The subject of this thesis is an investigation of various trellis coded modulation (TCM) techniques that have potential for out-performing conventional methods. The primary advantage of TCM over modulation schemes employing traditional error-correction coding is the ability to achieve increased power efficiency without the normal expansion of bandwidth introduced by the coding process. Thus, channels that are power limited and bandwidth limited are an ideal application for TCM.

In this thesis, four areas of interest are investigated. These include: signal constellation design, multilevel convolutional coding, adaptive TCM and finally low-complexity implementation of TCM.

An investigation of the effect of signal constellation design on probability of error has led to the optimisation of constellation angles for a given channel signal to noise ratio and a given code.

The use of multilevel convolutional codes based on rings of integers and multi-dimensional modulation is presented.

The potential benefits of incorporating several modulation schemes with adaptive TCM which require a single decoder are also investigated.

The final area of investigation has been the development of an algorithm for decoding of convolutional codes with a low complexity decoder.

The research described in this thesis investigated the use of trellis coded modulation to develop various techniques applicable to digital data transmis-

sion systems. Throughout this work, emphasis has been placed on enhancing the performance or complexity of conventional communication systems by simple modifications to the existing structures.

Chapter 1

Introduction

Communication theory provides a thorough theoretical background for establishing efficient, reliable, and secure communications systems. It is composed of three important theories, which can be referred to as Nyquist theory, Wiener-Kotelnikov theory, and Shannon theory. The aim of Nyquist's theory is to control distortion in the transmission of bandlimited pulse sequences. Nyquist's achievement in 1928 established the three criteria for distortionless transmission of such sequences long before the beginning of the age of high-speed communications [Nyquist, (1928)]. In the early 1940s, Wiener [Wiener, (1949)] determined the structure of optimum receivers in the presence of noise in the sense of minimum mean-square error, and in the late 1940s Kotelnikov [Kotelnikov, (1959)] investigated optimum demodulation techniques for various modulation formats. The works of Wiener and Kotelnikov contributed much to the field of signal design. Continuing efforts on the investigation of optimum communication system design reached a climax when Shannon published his classic paper, " A Mathematical Theory of Communication" [Shannon, (1948)]. Shannon's random coding arguments established theoretical bounds on the achievable performance of communication systems and launched the field of error control coding.

It seems that, from the late 1950s, the three theories mentioned above have been gradually merging, with the result being the development of sophisticated signalling formats in recent years. In 1967, Viterbi derived an error exponent for convolutional codes using an asymptotically optimum decoding algorithm [Viterbi, (1967)]. This algorithm later became known as the

Viterbi algorithm. In the early 1970s, Forney emphasized the importance of the Viterbi algorithm [Forney, (1972)].

During the last twenty years a new area of research in modulation namely coded modulation or trellis coded modulation (TCM) has found tremendous interest and immediate applications. New modems and codecs that are already on the market, which use this coded modulation. A coded signal is one in which only a set of coded signal patterns are allowed. This often means that increased memory is introduced in the signal compared to uncoded transmission. After passage through a noisy channel, the signal is detected by a receiver that knows the set of patterns and chooses the pattern, or code word, that lies in some sense closest to what it receives. Bandwidth efficient coding is achieved by combined coding and modulation. Coding can reduce energy consumption or bandwidth, or both at once. The key to successful coded modulation is to jointly design the "coding" and the "digital modulation". Ungerboeck made various efforts [Ungerboeck, *et al*, (1976)] to improve the signal-to-noise ratio in channels with additive noise in the late 1970s by proposing TCM.

Further improvement of performance in terms of signal-to-noise ratio, particulary for multi- level signal constellations, requires an increase in the Euclidean distance between signal points. To increase the Euclidean distance, one of the best solutions is to encode the data sequences themselves. However, combined coding and modulation is not simple since large Hamming distance between differing data sequences does not necessarily imply

large Euclidean distances between modulated data sequences, unless the assignment of coded signals to modulated signals is cleverly made. In a paper appearing in 1982, Ungerboeck proposed [Ungerboeck, (1982)] a very interesting method of set partitioning which does not only provide a method for the assignment of coded signals to channel signals, but also provide a simple formula for the lower bound of the Euclidean distance between modulated data sequences [Ungerboeck, (1982)].

This thesis concentrates on the design of new trellis coded modulation techniques applicable to modern communication systems. The investigation considers signal constellation design, channel coding and TCM implementation.

The second chapter of this thesis reviews some of the theoretical background relevant to this thesis. In the first section, various forms of decoding for convolutional codes are described and their merits are compared. Chapter 3 considers the concept of asymmetrical signal constellation points for both convolutional and block codes.

Since the early 1960s, various types of multi-dimensional signal constellations were proposed by many researchers to more closely approach Shannon's bound. As expected from Shannon results, it has been shown that a performance improvement can be effectively obtained if higher dimensional signal spaces are allowed [Slepian, (1963), Wilson *et al*, (1984), Gersho *et al*, (1984), Welti *et al*, (1974)]. In chapter 4, convolutional codes over rings, in conjunction with multi-dimensional signal spaces, are consid-

ered.

Chapter 5 introduces a new concept of embedded trellis coded modulation which is an extension of embedded coding [Darnell *et al.* (1988a), Zolghadr *et al.* (1988)]. Embedded TCM employs a combination of modulation schemes, and forward error correction and detection in an automatic repeat request (ARQ) environment. Error detection is performed via real time channel evaluation [Zolghadr *et al.* (1988)] which is a by-product of the soft decision decoding of convolutional codes.

Chapter six of this thesis describes a novel implementation of the Viterbi decoding algorithm. This technique does not require as much computation per decoded data symbol as does the conventional Viterbi algorithm. The particular application studied is that of a bandwidth efficient data transmission system, transmitting a rate 1/2 convolutional coded 4PSK signal over a linear channel in the presence of additive white gaussian noise.

Finally, chapter seven discusses the main results of the research, and suggests ideas for future investigation and development in each area.

Chapter 2

Theoretical Background

2.1 Introduction

The use of coding for transmission of digital data over a noisy channel is a well established fact and was demonstrated by Shannon in 1948 [Shannon, (1948)]. Shannon proved that if the data rate is less than a quantity called the channel capacity, completely reliable communication over a noisy channel is possible with proper encoding and decoding. Following Shannon's work, the use of error-control coding to improve the error performance of digital communication systems has found numerous applications.

The error-control capabilities of the encoder/decoder employing either linear block codes [Peterson *et al.* (1972)], or convolutional codes [Viterbi *et al.* (1979)], or a combination of both [Forney, (1966)], derive from the fact that the redundancy introduced by the encoder can be exploited by the decoder to detect, locate, and correct a certain number of errors that may have occurred during transmission.

The presence of redundancy therefore has the effect of improving the average error rates by decreasing the effective information rate per transmission bandwidth. If the communication channel is primarily power limited, then error-correcting codes provide a powerful mean for improving the reliability of transmission. However, if the channel is bandwidth as well as power limited, different coding techniques that are bandwidth efficient must be sought.

Because of the increasing demand for available spectrum, many channels are indeed becoming both power and bandlimited. Following this fact, in the

past decade a great deal of research effort has been dedicated to the problem of devising coding and modulation techniques that have the advantage of improving error performance without sacrificing the bandwidth.

In the remainder of this chapter, a general model of a communication channel will be described and then a review of convolutional code construction and decoding techniques will be reviewed with particular emphasis on the performance and practical implementation. Finally, a coding technique known as trellis coded modulation (TCM) which allows significant coding gains over conventional uncoded modulation, without sacrificing data rate or requiring extra bandwidth [Ungerboeck, *et al.* (1976), Ungerboeck, (1982)] is described.

2.2 Channel Model

A digital communication system can be modelled as in Fig 2.1. The binary data x_n are transmitted from a digital source to a user at a rate of R bit/sec. The encoder processes the information stream x_n by adding redundant bits, at a rate of $1 - R_c$ redundant bits/information bit, and groups blocks of K encoded bits to form $M = 2^K$ different messages. The digital modulator maps the set of M messages at its input onto a set of M finite energy signals $S_m(t), m = 1, \ldots, M$, of duration T seconds, which are transmitted through the channel every $T = \log_2 M.R_c/R$ seconds.

Since any set of M finite energy signals can be represented as a linear com-

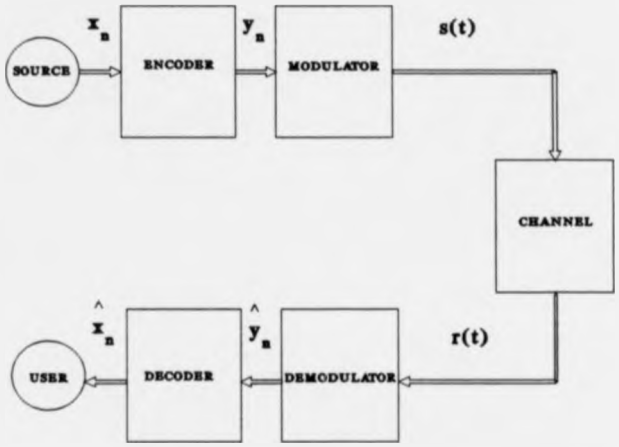


Figure 2.1: Model of a digital communication system

bination of $N \leq M$ orthonormal basis function $\varphi_1, \varphi_2, ..., \varphi_N$ [Franks, (1981)], the channel signals $S_m(t)$ can be conveniently expressed as [Arthurs *et al*, (1962)]

$$S_m(t) = \sum_{j=1}^{N} S_{mj}\varphi_j(t), \qquad m = 1, \ldots, N \tag{2.1}$$

where

$$S_{mj} = \int_0^T S_m(t)\varphi_j(t)dt \tag{2.2}$$

and

$$\int_0^T \varphi_i(t)\varphi_j(t)dt = 0 \quad \text{for } i \neq j \tag{2.3}$$

The energy E of the transmitted signal is defined as

$$E = \int_0^T S_m^2(t)dt \tag{2.4}$$

$$E = \sum_{j=1}^{N} S_{mj}^2 \quad \text{for all m}$$

The channel is modelled as an additive white Gaussian noise (AWGN) channel, which corrupts the transmitted signal by adding white Gaussian noise $n(t)$ of one-sided power spectral density N_0. In this case, the received signal can be expressed as

$$r(t) = S_m(t) + n(t), 0 \leq t \leq T \tag{2.5}$$

At the receiver, the demodulator performs the inverse operation of the modulator by projecting the received random process $r(t)$ onto the set of orthonormal basis function $\varphi_j(t)$, namely

$$r_j = \int_0^T r(t)\varphi_j(t)dt, \quad \text{j= 1, . . . ,N} \tag{2.6}$$

By defining

$$n_j = \int_0^T n(t)\varphi_j(t)dt, \quad \text{j= 1,. . . ,N} \tag{2.7}$$

the output of the demodulator is completely specified by the set of N random variables

$$\underline{r}_j = \underline{S}_{mj} + n_j, \quad \text{j=1,. . . ,N} \tag{2.8}$$

The n_j's are independent random variables with zero mean and variance $N_0/2$. The conditional probability density function of the vector $\underline{r}$, given that

the vector $\underline{S}_m = \{S_{m1}, S_{m2}, \ldots, S_{mN}\}$ was transmitted is then [Viterbi *et al.* (1979)].

$$p(\underline{r}/\underline{s}_m) = \Pi_{j=1}^{N} \frac{1}{\sqrt{\pi . N_0}} e^{-(r_j - S_{mj})^2/N_0} \tag{2.9}$$

The task of the decoder is to obtain the best possible estimate, in the sense of minimum error probability, of the transmitted data, from the set of N observable r_j.

2.3 Convolutional Codes

Convolutional codes were first introduced by Elias [Elias, (1955)] and have been applied over the past 30 years to increase the efficiency of numerous communication systems, where they invariably outperform block codes of the same order of complexity. A binary convolutional code is generated by passing the information sequence to be transmitted through a linear finite-state shift register. In general, the shift register consists of L (k-bit) stages clocked k bits at a time, and n linear algebraic function generators, as shown in Fig 2.2. Fig 2.3 shows a simple binary systematic rate-half convolutional encoder of constraint length $L = 3$ with $k = 1$. The input to this encoder is a binary sequence

$$x = (\ldots, x_{-1}, x_0, x_1, \ldots) \tag{2.10}$$

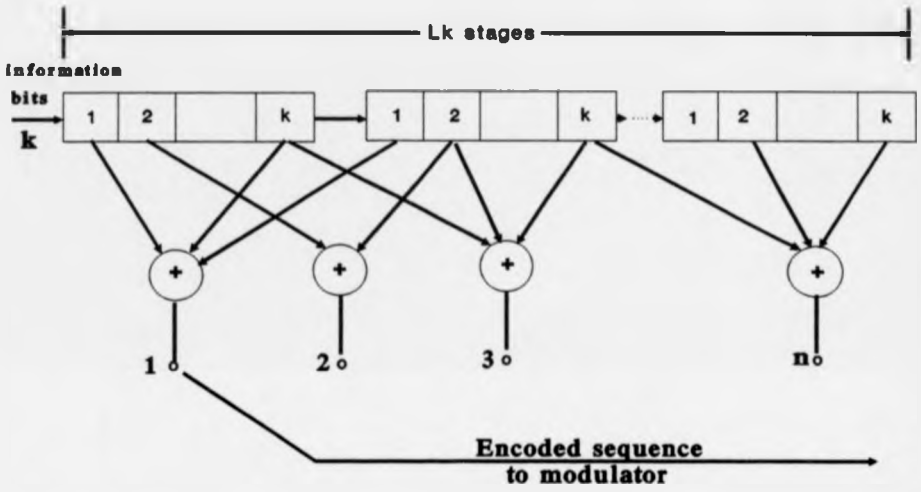


Figure 2.2: General convolutional encoder Model

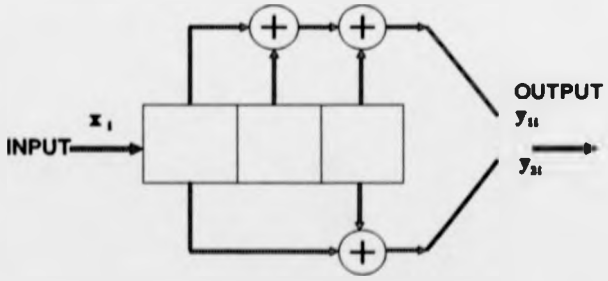


Figure 2.3: Half rate code

The outputs are two binary sequences y_1 and y_2. The i^{th} outputs are given by

$$y_{1i} = x_i \bigoplus x_{i-1} \bigoplus x_{i-2}$$
$$y_{2i} = x_i \bigoplus x_{i-2} \tag{2.11}$$

where $\bigoplus$ denotes modulo 2 addition.

The term "convolutional" comes from the observation that the output sequences can be regarded as a convolution of the input sequence with certain generator sequences. With the input and output sequences, we associate sequences in the delay operator D:

$$x(D) = \ldots + x_{x-1}D^{-1} + x_0 + x_1 D + x_2 D^2 + \ldots$$
$$y_1(D) = \ldots + y_{1,-1}D^{-1} + y_{10} + y_{11} D + y_{12} D^2 + \ldots \tag{2.12}$$
$$y_2(D) = \ldots + y_{2,-1}D^{-1} + y_{20} + y_{21} D + y_{22} D^2 + \ldots$$

The input/output relationship are expressed as:

$$y_1(D) = g_1(D)x(D)$$
$$y_2(D) = g_2(D)x(D), \tag{2.13}$$

where the generator polynomials $g_1(D)$ and $g_2(D)$ are

$$g_1 = 1 + D + D^2$$

$$g_2 = 1 + D^2 \tag{2.14}$$

We define a general (n, k) convolutional encoder by a matrix of generator

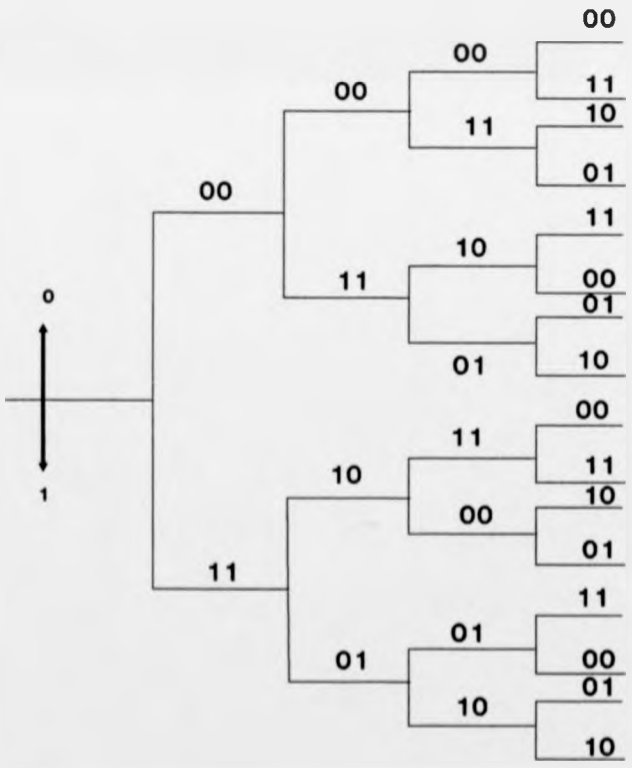


Figure 2.4: Tree diagram for rate 1/2, L=3 convolutional code

polynomials $g_{ij}(D), 1 \leq i \leq k, 1 \leq j \leq n$. There are k input sequences $x_i(D)$, and n output sequences $y_j(D)$. The input/output relations given by

$$y_j(D) = \sum_{i=1}^{k} x_i(D) g_{ij}(D) \tag{2.15}$$

The code rate is defined as $R_c = k/n$.

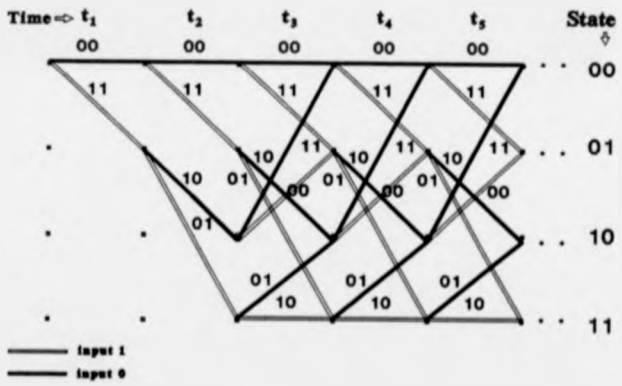


Figure 2.5: Trellis diagram for rate 1/2, L=3 convolutional code

There are three methods that are often used to describe a convolutional code. These are the tree diagram, the trellis diagram, and the state diagram. For example, the tree diagram for the convolutional encoder shown in Fig 2.3 is illustrated in Fig 2.4. Close observation of the tree diagram reveals that the structure repeats itself after the third stage. This behaviour is consistent with the fact that the constraint length $L = 3$. Using this repetition of structure, another diagram which is more compact, called a trellis can be obtained.

This is shown in Fig.2.5. Since the output of the encoder is determined by the input and the state of the encoder, an even more compact diagram than the trellis is the state diagram. The state diagram is simply a graph of the possible states of the encoder and the possible transitions from one state to another. The state diagram for the encoder in Fig 2.3 is shown in Fig 2.6.

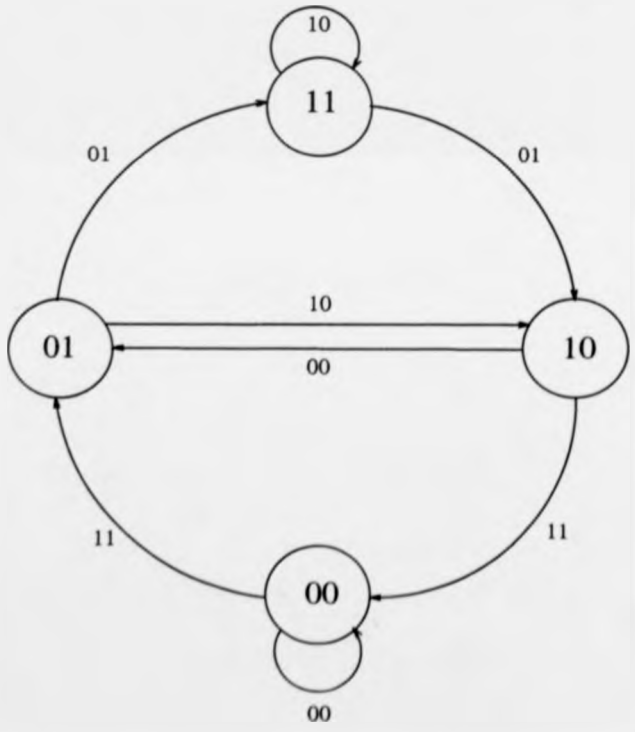


Figure 2.6: State diagram for rate 1/2, L=3 convolutional code

2.4 Decoding of Convolutional Codes

A number of algorithms has been proposed for decoding convolutional codes. The earliest was the sequential decoding algorithm, originally proposed by Wozencraft [Wozencraft, (1957), Wozencraft *et al.* (1961)] and subsequently modified by Fano [Fano, (1963)]. A type of sequential decoding algorithm, called the stack algorithm, has been proposed independently by Jelinek and Zigangirov, [Jelinek(1969), Zigangirov, (1966)]. Another method which has been applied to decoding for a binary symmetric channel (BSC) is feedback decoding [Heller, (1975)]. For some convolutional codes, the feedback decoder simplifies to a form called a majority logic decoder or a threshold decoder [Massey, (1963)]. Finally the optimum decoding algorithm (in the sense of maximum likelihood decoding of the entire sequence) for convolutional codes is the Viterbi algorithm. In the next few sections, the performance and practical implementation of the above techniques are compared.

2.4.1 Viterbi Decoding Algorithm

A practical technique for maximum likelihood decoding is the Viterbi algorithm introduced by Viterbi in 1967 [Viterbi, (1967)]. This decoding algorithm uses the trellis structure of the code and determines the maximum likelihood estimate of the transmitted sequence that has the largest metric [Clark *et al.* (1988)]. The survivor is defined as the most probable path that has the largest accumulated metric. With these definitions, the Viterbi algo-

rithm simply finds the path through the trellis with the largest accumulated metric in such a way that it processes a received sequence in an iterative manner. At each step, it compares the metric of all paths entering each state, stores only the survivor with the largest accumulated metric, and discards the unlikely paths at every state, which reduces the decoding effort. Therefore the Viterbi decoder must produce an estimate of the code sequence based on the received sequence.

The storage requirements of the Viterbi decoder grows exponentially with constraint length L. For a code with rate $1/n$, the decoder retains a set of 2^{L-1} paths after each decoding step. With high probability, these paths will not be mutually disjoint very far back from the present decoding depth [Forney *et al.* (1984)]. All of the 2^{L-1} paths tend to have a common stem which eventually branches to the various states. Thus if the decoder stores enough of the history of the 2^{L-1} paths, the oldest bits on all paths will be the same. A simple decoder implementation, then, contains a fixed amount of path history and outputs the oldest bit on the common or highest metric path each time it steps one level deeper into the trellis.

2.4.2 Sequential Decoding

The sequential decoding algorithm [Wozencraft, (1957), Wozencraft *et al.* (1961)] searches for the most probable path through the tree or trellis by examining one path at a time. The increment added to the metric along each branch

is proportional to the probability of the received signal for that branch, just as in Viterbi decoding, with the exception that an additional negative constant is added to each branch metric. The value of this constant is selected such that the metric for the correct and incorrect path on the average will increase and decrease respectively. By comparing the metric of a candidate path with a moving (increasing) threshold, the algorithm detects and discards incorrect paths. The sequential decoding algorithm requires a buffer memory in the decoder to store incoming demodulated data during periods when the decoder is searching for alternate paths. When a search terminates, the decoder must be capable of processing demodulated bits sufficiently fast to empty the buffer prior to commencing a new search. Occasionally, during extremely long searches, the buffer may overflow [Viterbi *et al.* (1979)]. This causes loss of data, a condition which can be remedied by re-transmissions of the lost information.

The Fano algorithm which is a modified version of the sequential decoding originally proposed by Wozencraft [Wozencraft, (1957), Wozencraft *et al.* (1961)] has been successfully implemented in several communications systems. Its error rate performance is comparable to that of Viterbi decoding. However, in comparison with Viterbi decoding, sequential decoding has a significantly larger decoding delay. On the positive side, sequential decoding requires less storage than Viterbi decoding and, hence, it appears attractive for convolutional codes with a large constraint length.

2.4.3 Stack Algorithm

In contrast to the Viterbi algorithm which keeps track of $2^{(L-1)k}$ paths and corresponding metrics, (L is the constraint length and k is the number of information bits entering the binary convolutional encoder), the stack sequential decoding algorithm deals with fewer paths and their corresponding metrics. In a stack algorithm [Jelinek(1969), Zigangirov, (1966)] the more probable paths are ordered according to their metrics, with the path at the top of the stack having the largest metric. At each step of the algorithm, only the path at the top of the stack is extended by one branch. This yields 2^k successors and their corresponding metrics. These 2^k successors along with the other paths are then re-ordered according to the values of the metrics and all paths with metrics that fall below some pre-selected amount from the metric of the top path are discarded. Then the process of extending the path with the largest metric is repeated. It is apparent that when none of the 2^k extensions of the path with the largest metric remains at the top of the stack, the next step in the search involves the extension of another path which has climbed to the top of the stack. It follows that the algorithm does not necessarily advance by one branch through the trellis in every iteration. Consequently some amount of storage must be provided for received signals in order to allow the algorithm to extend the search along one of the shorter paths, when a path reaches the top of the stack.

In a comparison of the stack algorithm with the Viterbi algorithm, the

stack algorithm requires fewer metric computations, but this computation saving is offset to a large extent by the computations involved in re-ordering the stack after every iteration. In comparison with the Fano algorithm, the stack algorithm is computationally simpler since there is no retracing over the same path as is done in the Fano algorithm. On the other hand, the stack algorithm requires more storage than the Fano algorithm [Viterbi *et al.* (1979)].

2.4.4 Threshold Decoding

In 1963, Massey introduced [Massey, (1963)] a less efficient but simpler to implement decoding method than sequential decoding called majority logic or threshold decoding, which is applicable to convolutional codes. Threshold decoding is conceptually and practically the closest to block decoding. Its conceptual simplicity stems from the fact that threshold decoding of convolutional codes begins with the calculation of a set of syndrome digits, the definition of which is very similar to the definition for block codes. Thus threshold decoding lacks the search aspects of Viterbi and sequential decoding. In fact, threshold decoding is inferior in performance when compared to Viterbi or sequential decoding, but the implementation of the decoder is much simpler.

Threshold decoding can achieve a coding gain of 1 to 3 dB with a relatively simple implementation. Considerable research interest has produced extensive lists of good codes [Robinson *et al.* (1967), Klieber, (1970)].

2.4.5 Feedback Decoding

Another type of sub-optimal decoding is feedback decoding [Heller, (1975)], a special case of which is threshold decoding. Although the performance of Viterbi and sequential decoding is generally superior to that of feedback decoding, the latter has the distinct advantage of being easy to implement. The operation of a feedback decoder is described as follows. The first l branches are examined, and the path yielding the largest metric with respect to the ln (n is the number of channel symbols) code digits of the received sequence is selected. At this point, we decode the first information bit corresponding to the first tree branch on the maximum metric path chosen. This decision uniquely moves us to a specific node at the second step in the tree. Starting there, we consider all paths of length l branches emanating from that node and again make a maximum metric selection now with respect to the sequence of ln received code digits whose first is the $(n+1)^{th}$. We then proceed to decode the second information bit. This process continues in the manner just described. The technique is called feedback decoding because each decoding decision on an information bit is fed back to affect future decisions. Hence, decoding schemes of this type possess the undesirable property of error propagation caused by incorrect decisions. For a large class of convolutional codes, however, this phenomenon does not present a serious problem.

It should be clear that the decoding accuracy of a feedback decoding

scheme improves as the parameter l increases. Unfortunately, however, the complexity of the decision device also grows as l gets large. Threshold decoding provides a partial solution to this paradox, but it is only useful for a specially constructed class of convolutional codes and only for moderate value of L. Feedback decoders are particularly applicable to burst error channels [Viterbi *et al.* (1979)].

2.5 Performance Bounds for Viterbi Decoding

The most useful techniques for estimating the performance of convolutional coding are union bounds [Lin *et al.* (1983)] and computer simulation. The usefulness of computer simulation is limited by the long computation times that are required to get a good statistical sample. Consequently, the greatest usefulness of this technique is in special coding applications where union bounds cannot be used to provide a good performance estimate. The union bound approach for convolutional codes provides performance estimates accurate to within a small fraction of a decibel for all signal to noise ratios large enough to give an error rate of 10^{-3} or less [Lin *et al.* (1983)].

Bounds on the performance afforded by a Viterbi decoder can be obtained if one has knowledge of the distribution of codeword weights for the code employed [Forney, (1973), Viterbi, (1971)]. Since a convolutional encoder is

a sequential circuit, its operation can be described by a state diagram. The state of the encoder is defined as its shift register contents. Suppose that we are given a binary input/output symmetric channel. Assume the all zero path to be the transmitted path. A first error event is made at the j^{th} received branch if the all zero path is eliminated at this point by another path later merging with it. For small constraint lengths the Hamming weight of all such paths can be determined by the following procedure [Viterbi, (1969)].

Consider a state diagram with the all zero state split open. Label the branches with D^t , where t is the Hamming weight of that branch. For example, the $L = 3$, $R_c = 1/2$ convolutional code of Fig 2.3 with sub-generator polynomials $g_1(D) = D^2 + D + 1$ and $g_2(D) = D^2 + 1$ would have a modified state diagram as in Figure 2.7. The overall transfer function of this directed graph can then be obtained.

$$T(D) = \frac{D^5}{1 - 2D} = D^5 + 2D^6 + 4D^7 + \cdots + 2^k D^{k+5} + \cdots \qquad (2.16)$$

Thus there is one incorrect path of weight 5, two of weight 6, etc. Using Eq.2.16, we may then find an upper bound on the probability of error at any step in the Viterbi decoding algorithm. Assume that BPSK modulation is used on an additive white Gaussian noise channel with one sided spectral density N_0. And let the positive and negative pulses both have energy E_b. Then the probability of error in comparing an incorrect path which differs in

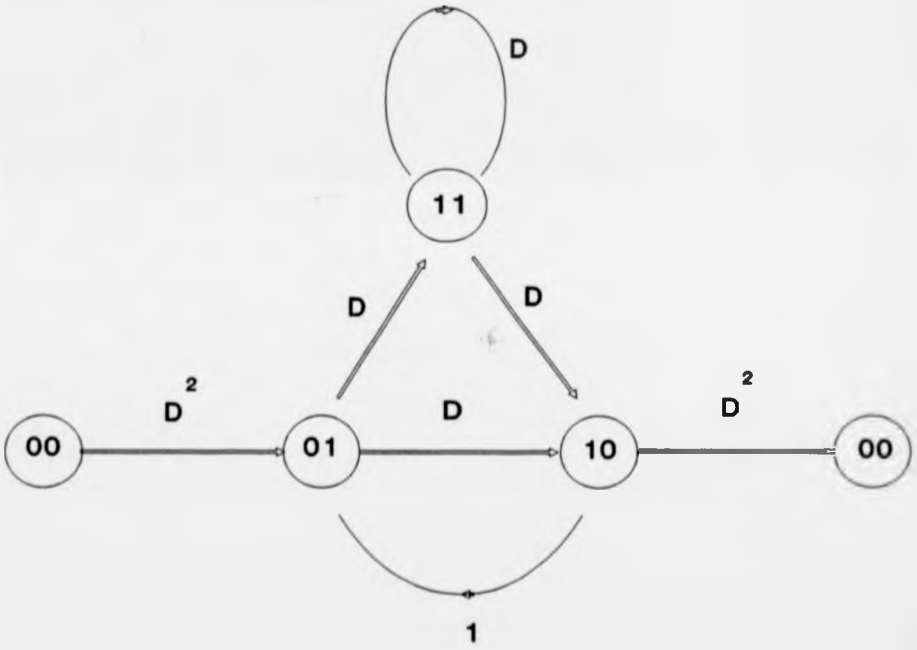


Figure 2.7: Modified state diagram of Fig 2.6

p symbols from the correct path is given by [Wozencraft *et al*, (1965)]:

$$P_e = Q\left(\sqrt{\frac{2pE_b}{vN_0}}\right) \tag{2.17}$$

where

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-\frac{\alpha^2}{2}} d\alpha \tag{2.18}$$

and $v = 1/R_c log_2(M)$ in the general case is the number of channel symbols per information bit for M-ary PSK modulation schemes. If $R_c = k/n$ is the code rate, for (n,k) convolutional code. Then in the case of BPSK, where $v = 1/R_c$, using Eq.2.17 and 2.18, the probability of symbol error (P_s) can be upper bounded with a union bound yielding [Viterbi *et al*, (1979)].

$$\begin{aligned} P_s &\leq Q\left(\sqrt{5\frac{E_b}{N_0}}\right) + 2Q\left(\sqrt{6\frac{E_b}{N_0}}\right) + 4Q\left(\sqrt{7\frac{E_b}{N_0}}\right) + \cdots \\ &\leq Q\left(\sqrt{5\frac{E_b}{N_0}}\right)\left\{1 + 2e^{\frac{E_b}{2N_0}} + e^{-\frac{E_b}{N_0}}\right\} \\ &\leq \frac{Q\left(\sqrt{\frac{E_b}{N_0}}\right)}{1 - e^{-\frac{E_b}{N_0}}} \end{aligned} \tag{2.19}$$

To determine the bit probability of error we need to know the number of bit errors resulting from an incorrect path decision. These may be computed by adding a factor N to each branch of the modified state diagram which corresponds to a "1" input bit. We can also find the lengths of these paths by adding a factor of l to each branch. With these changes, Fig 2.7 becomes the diagram shown in Fig 2.8. Now the transfer function is

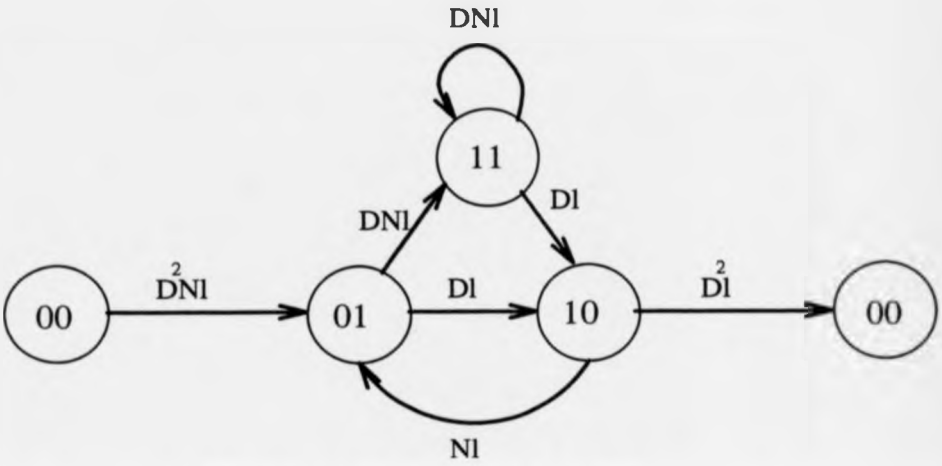


Figure 2.8: Modified diagram of Fig 2.7

$$T(D,N,l) = \frac{D^5Nl^3}{1 - DNl - DNl^2} \tag{2.20}$$
$$= D^5Nl^3 + D^6N^2(l^4 + l^5) + D^7N^3(l^5 + 2l^6 + l^7) + \cdots$$

There is one path of weight 5, and it is three branches long and results in one bit error. There are two paths of weight 6, one of length 4 and one of length 5, and both result in two bit errors, etc. In terms of the definitions above, the upper bound on the bit error rate can be obtained as [Viterbi *et al.* (1979)].

$$P_b \leq \frac{\partial}{\partial N} T(D,N,l)\Big|_{N=l=1} \tag{2.21}$$

Differentiation of Eq.2.21 with respect to N followed by setting N and l equal to one weights the incorrect paths by their number of bits errors. Then using Eqs.2.17, and 2.18 and a union bound yields the following bound on

the bit probability of error [Viterbi *et al*, (1979)].

$$\begin{aligned} P_b &\leq Q\left(\sqrt{5\frac{E_b}{N_0}}\right) + 2.2Q\left(\sqrt{6\frac{E_b}{N_0}}\right) + 3.4Q\left(\sqrt{7\frac{E_b}{N_0}}\right) + \cdots \\ &\leq Q\left(\sqrt{5\frac{E_b}{N_0}}\right)\left\{1 + 2.2e^{\frac{E_b}{2N_0}} + 3.4e^{-\frac{E_b}{2N_0}}\right\} \\ &\leq \frac{Q\left(\sqrt{\frac{5E_b}{N_0}}\right)}{\left(1 - 2e^{-\frac{E_b}{2N_0}}\right)^2} \end{aligned} \tag{2.22}$$

The state diagram approach is also helpful in understanding catastrophic error propagating codes. A code exhibits catastrophic error propagation if and only if there was a zero weight path from some non-zero state back to itself [Massey, (1968)]. In terms of the modified state diagram this zero weight path is just a closed loop with zero weight. Although the state diagram approach is helpful in finding good codes for small constraint length, it is extremely difficult to find the transfer function for codes with large constraint length.

So far the calculation of probability of error has been based on Hamming distance between sequences. Therefore, equation 2.22 represents the probability of error for the rate half code shown in Fig. 2.3, in conjunction with coherent BPSK and hard-decision decoding. In order to calculate P_b, using soft-decision decoding, the Hamming distance in the above calculation should be replaced by Euclidean distance (ED).

2.5.1 Free Distance and Coding Gain

The distance between a selected code sequence and each of the other possible code sequences in Fig. 2.5 can be calculated. The minimum value of these distances is a measure of the code's error-correcting capability [Viterbi *et al*, (1979)]. Since a convolutional code is a group or linear code [Lin *et al*, (1983)], the set of distances is independent of which sequence is selected, so there is no loss in generality in selecting the all-zeros sequence. Assuming the all-zeros sequence is transmitted, an error- event is identified as a departure from the all zeros path followed by a remerging with the all- zeros path. Error events both start and end in the 00 state and do not return to the 00 state anywhere in between. A decoding error occurs whenever the received symbols are closer in distance to some error-event path than to the all-zeros path. Referring to the trellis diagram of Fig. 2.5, one error-event path of length 3 departs from the all-zeros path at time t_1 and remerges with it at time t_4. This path represents the decision that the received sequence is closer to the coded sequence 11 10 11 than to the transmitted sequence 00 00 00. The minimum Euclidean distance in the set of all paths that diverge and remerge, is called the free Euclidean distance d_{free}. Using soft-decision maximum likelihood decoding, and assuming unit average signal power, and Gaussian noise with variance σ^2 per dimension, a lower bound on the error even probability, P_e,

can be expressed in terms of d_{free} as [Forney, (1973)]

$$P_e \geq Q\left(\frac{d_{free}}{2\sigma}\right) \tag{2.23}$$

where $Q(.)$ is the complementary error function. At high signal-to-noise ratios, the bound in the above equation is asymptotically exact. The asymptotic coding gain G, in dB, compared with some reference system of the same average signal power and noise variance, is therefore expressed as

$$G = 20log_{10}\left(\frac{d_{free}}{d_{ref}}\right) \tag{2.24}$$

where d_{ref} is the free Euclidean distance of the reference system. A practical definition of coding gain is the difference between the E_b/N_0 required to achieve a given probability of error, when a code is used and when a code is not used.

2.6 Trellis Coded Modulation

Trellis coded modulation (TCM), which combines channel coding and modulation into one operation, improves the reliability of a digital transmission system without increasing the transmitted power or bandwidth [Biglieri, (1984), Massey(1986), Ungerboeck, (1982)]. It is a "sliding window" method of encoding a sequence of source symbols into a sequence of channel symbols input to a noisy transmission channel [Biglieri *et al.* (1988)][Calderbank *et al.* (1984)].

The receiver ideally performs a maximum likelihood soft decoding. The objective of the code design is to maximise the minimum Euclidean distance between encoded channel symbol sequences [Ungerboeck, (1982)].

It has been shown [Biglieri, (1984), Ungerboeck, (1982)] that TCM can easily achieve significant power gains (from 3 to 6 dB) without bandwidth expansion. At first, it may seem that this statement violates some basic power-bandwidth-error probability trade-off principle. However, there is still a trade-off at work. TCM achieving coding gain at the expense of decoder complexity.

The purpose of trellis coding is to increase the equivalent minimum Euclidean distance of channel symbol sequences without increasing average power or decreasing rate by introducing interdependence between the channel symbols. The channel encoder must have memory in order to create such interdependence. At the receiver side, optimum performance can be achieved by a maximum likelihood sequence estimator, which is usually implemented with the Viterbi algorithm. Thus, receiver complexity is increased.

Consider an uncoded system over an AWGN channel. Suppose we transmit k independent channel symbols, $x_i \in \Omega, \quad i = 1, \cdots, k$ where Ω is the channel input alphabet with size M. There are M^k possible channel symbol sequences and each channel symbol associated with a source symbol is generated independently. The performance of this system depends on the minimum squared Euclidean distance (d^2_{min}) between the closest pair of the signal elements in Ω.

In a coded system, successive symbols in the channel symbol sequence are strongly correlated. One way to introduce this interdependence is to restrict transmitted signal vectors to be a subset of Ω^k. This idea has been used in error correcting codes, where, however, the transmission rate is reduced. To avoid this, we can choose a larger channel input alphabet Ω' with size M', and the same average power. In this approach, we can select M^k signal vectors as a subset of Ω'^k, maintaining the same transmission rate. The minimum Euclidean distance for a given average signal power of the new channel input alphabet is indeed decreased. However, because of the correlation between the channel symbols, this reduced distance no longer determines the system performance. Instead, performance depends on the minimum Euclidean distance d^2_{free} between the two possible sequences, which is greater than the minimum distance d^2_{min} between signals in Ω.

An efficient coding scheme was proposed by Ungerboeck [Ungerboeck, (1982)] and applied to various modulation formats such as M-ary Pulse-Amplitude-Modulation (PAM), Phase-Shift-Keying (PSK), and Quadrature-Amplitude-Modulation (QAM). Fig. 2.9 gives a block diagram of the encoding operation. The entire encoding procedure is better explained by means of an example. We therefore will consider the case of transmission of 2 bits/T by 8-PSK and derive two simple codes that achieve significant coding gains over uncoded transmission by 4-PSK modulation. The information bits are first encoded by a rate 2/3 convolutional encoder of a given constraint length L. Associated with the encoder is a trellis diagram with 2^{L-1} states. The next step

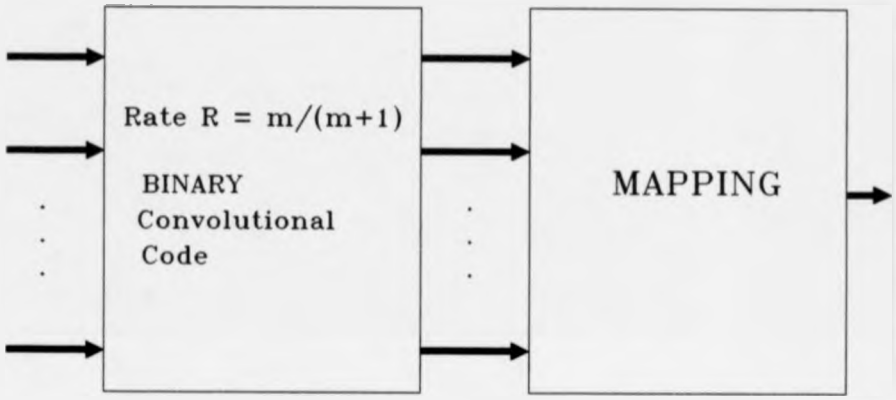


Figure 2.9: Encoder structure

consists in mapping, to the state transitions in the trellis diagram, channel signals from the 8-PSK signal constellation. Since a good code design aims to maximise the minimum Euclidean distance (ED) between all pairs of channel signal sequences $\{x_i\}$ and $\{x'_i\}$ generated by the encoder, mapping rules must be devised so as to maximise the free ED, (d_{free}). The mapping rules proposed in [Ungerboeck, (1982)] and referred to as "mapping by set partitioning" are now explained. The 8-PSK signal constellation is first partitioned into 2 and 4 subsets. Such partitioning, shown in Fig 2.10, has the objective of deriving subsets that have the largest possible minimum ED

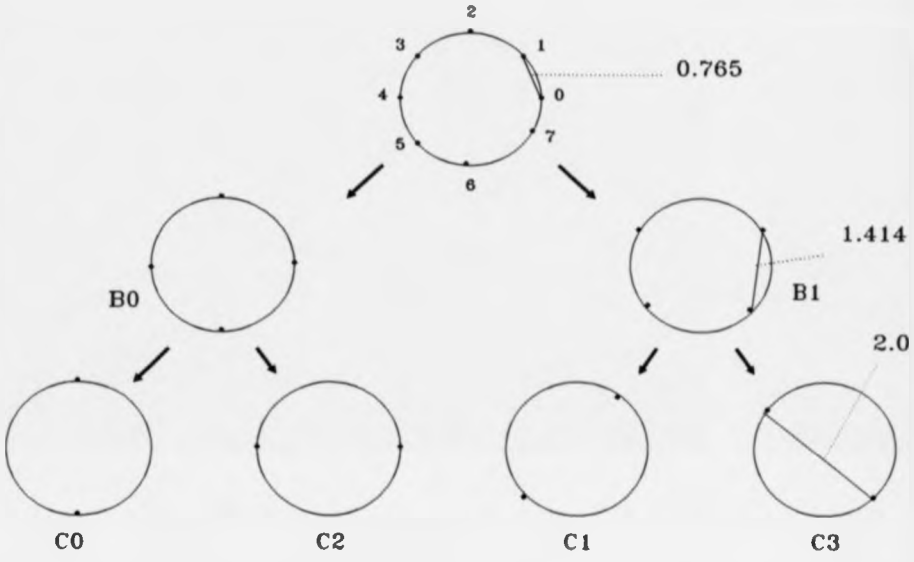


Figure 2.10: Set partitioning of the 8-PSK signal constellation

between any two signals belonging to the same subset. In Fig. 2.10 such minimum ED, denoted by d_0, d_1, and d_2, are given. Here, the average signal energy has been normalised to unity, i.e. $E = 1$. The following set of rules was then used in assigning channel signals to state transitions:

1. transitions originating from the same state receive signals either from subset $B0$ or $B1$.

2. transitions joining in the same state receive signals either from subset $B0$ or $B1$.

3. parallel transitions receive signals either from subset $C0$ or $C1$ or $C2$ or $C3$.

Finally, two simple codes can now be derived. In Fig. 2.11 and Fig. 2.12 the encoders and trellis diagrams for four-state and eight-state encoders are shown, respectively. State transitions are associated with channel signals according to rules 1) – 3). In the first case $d_{free} = d_2 = 2$, namely the minimum ED between the two signals in either subsets $C0$ or $C1$ or $C2$ or $C3$. In the second case, $d_{free} = (d_1^2 + d_0^2 + d_1^2)^{1/2} = 2.14$.

Soft maximum likelihood decoding (Viterbi decoding) of the above codes yields an asymptotic coding gain over uncoded 4-PSK of $d_{free}/\sqrt{2}$. Therefore, the codes of Fig. 2.11 and Fig. 2.12 achieve 3 dB and 3.6 dB coding gains, respectively.

From the trellis diagrams of Figs. 2.11 and 2.12, it is clear that the combination of rules 1)-3) assures that d_{free}^2 is at least $2d_1^2 = d_2^2$, thus yielding a coding gain of at least 3 dB over uncoded 4-PSK.

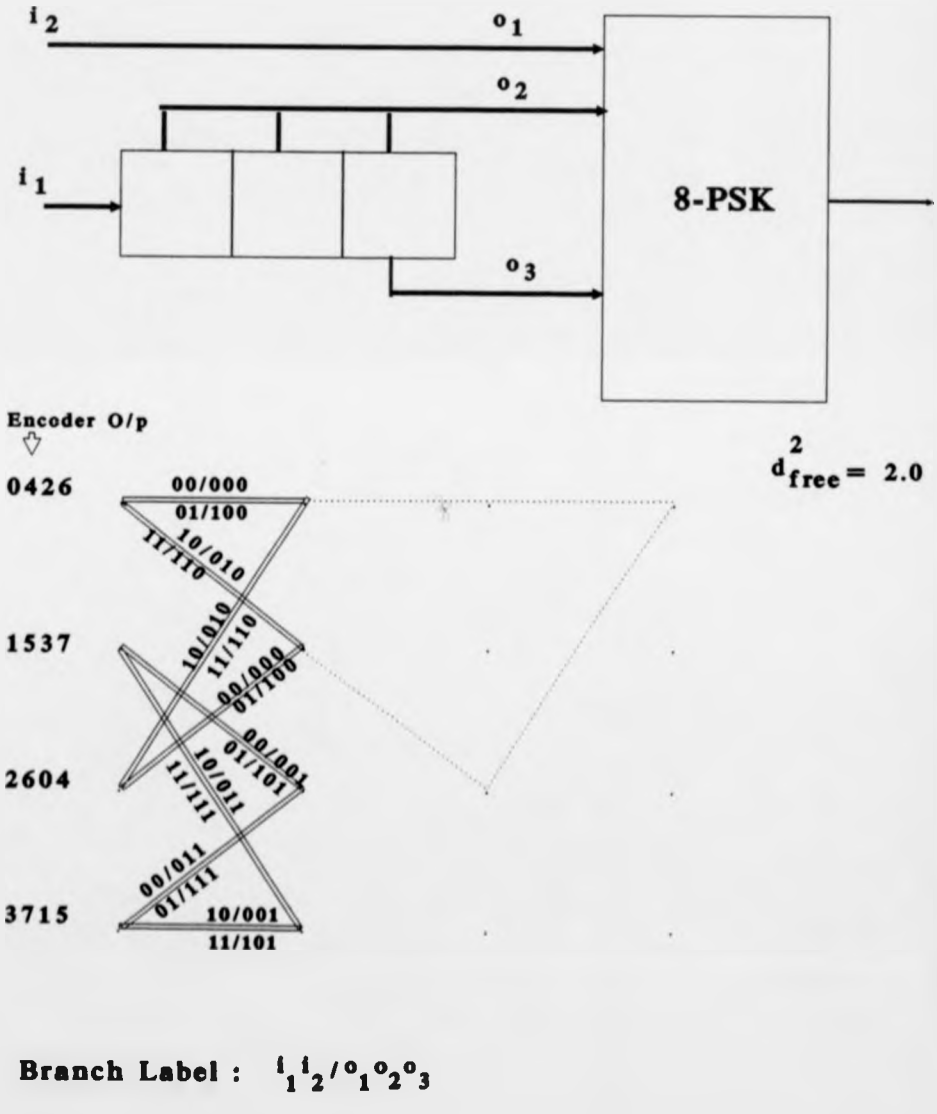


Figure 2.11: Convolutional encoder (constraint length 3), and its corresponding trellis diagram for coded 8-PSK

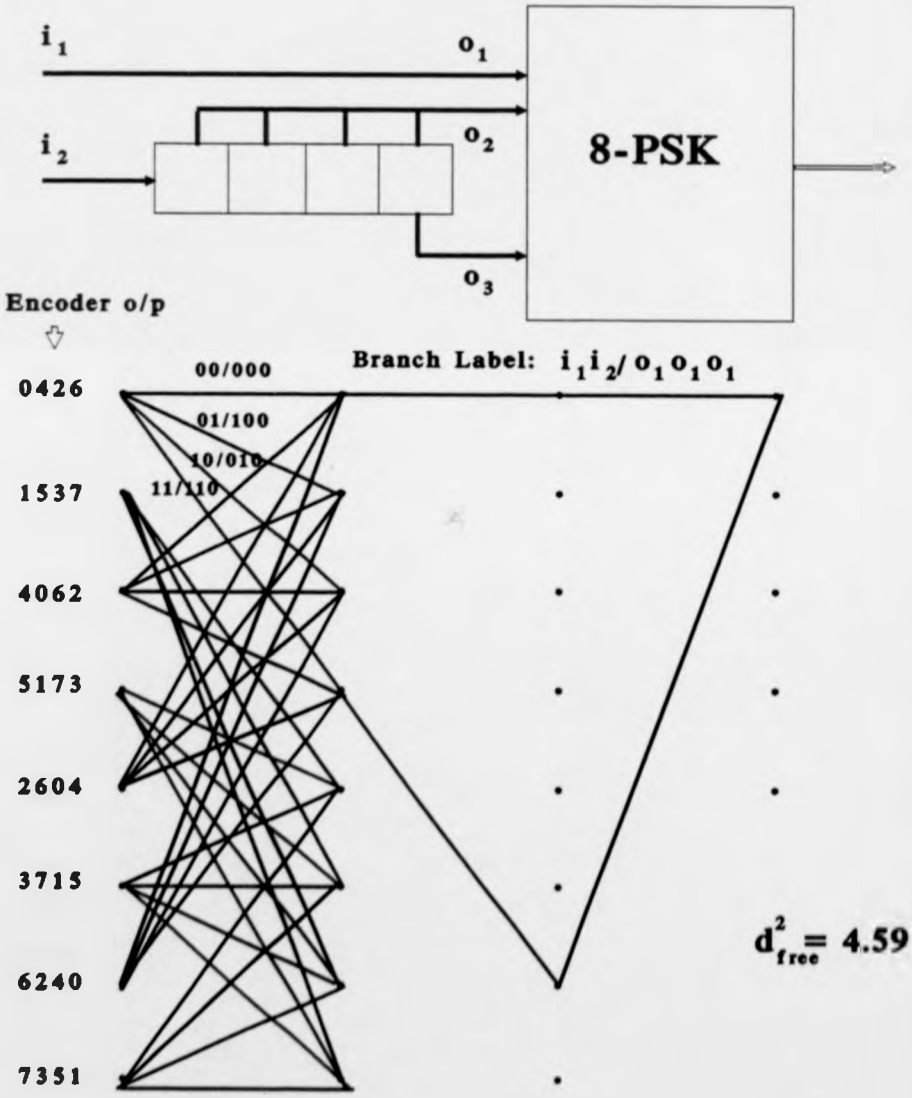


Figure 2.12: Convolutional encoder (constraint length 4), and its corresponding trellis diagram for coded 8-PSK

Chapter 3

Asymmetrical Trellis Coded Modulation

3.1 Introduction

Traditionally, symmetric signal constellations have been used for both uncoded and coded systems. In this chapter, the effect of constellation angle in coded systems using both convolutional and block codes has been investigated and an optimum signal constellation for a given trellis coded system has been found. Therefore, by designing the signal constellations to be asymmetric; i.e., signal points are non-uniformly spaced, one can, in many instances, obtain a performance gain over TCM schemes based on traditional; i.e., symmetric constellations. For example, by designing asymmetric M-PSK signal constellations and combining them with optimised trellis coding, one can further improve the performance of coded systems without increasing average or peak power or changing the bandwidth constraints imposed on the system. In fact, properties of the signal set such as constant envelope and number of dimensions are not changed by the asymmetry. First, asymmetrical signal design using 4-PSK is considered and performance analysis of such a system is investigated. Then trellis decoding of a class of block codes called array codes [Farrell, (1979)] is considered and combined block coding and modulation, better known as block coded modulation (BCM), is described. Finally the effect of asymmetrical signal constellations in BCM is presented.

3.2 Trellis Coding with Asymmetrical Signal Design

The first look at the idea of using other than conventional symmetric signal sets in coded systems appears in the work of Divsalar and Yuen [Divsalar *et al.* (1980), Divsalar *et al.* (1984)].

Typical symmetric and asymmetric signal sets are shown in Fig. 3.1. The asymmetric M signal set is created by adding together the symmetrical $M/2$ point set with a rotated version of itself. The approach of assigning signals to transitions of the trellis code is based on the mapping by set partitioning rule [Ungerboeck, (1982)]. Each subset including the original set is partitioned into two subsets with an equal number of signals and with the largest minimum distance among signals within the subset. Fig. 3.2 demonstrates the set partitioning method as applied to asymmetric 4-PSK. What remains, is to optimise the rotation angle ϕ in asymmetric M-PSK [Arani *et al.* (1990)]. The criterion of optimisation will be to maximise the free Euclidean distance (ED) (or its square) of the TCM scheme for a given channel condition. In the next section the relation of this performance measure, and the average bit error probability of the overall coded system for a half rate convolutional code with asymmetrical 4-PSK is calculated.

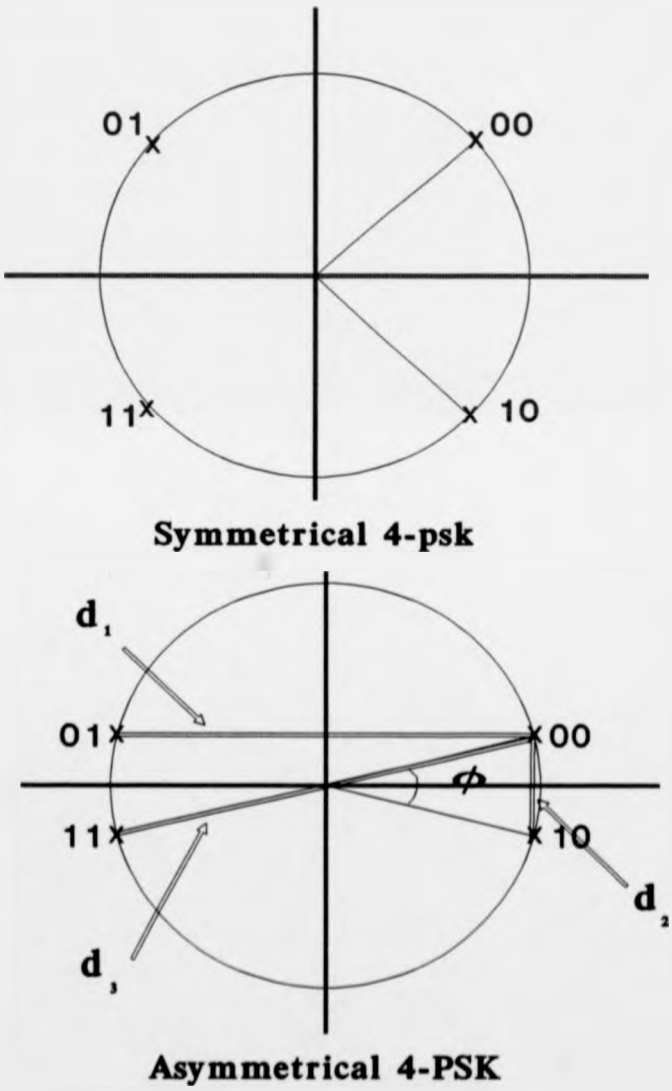


Figure 3.1: Typical symmetric & asymmetric signal sets

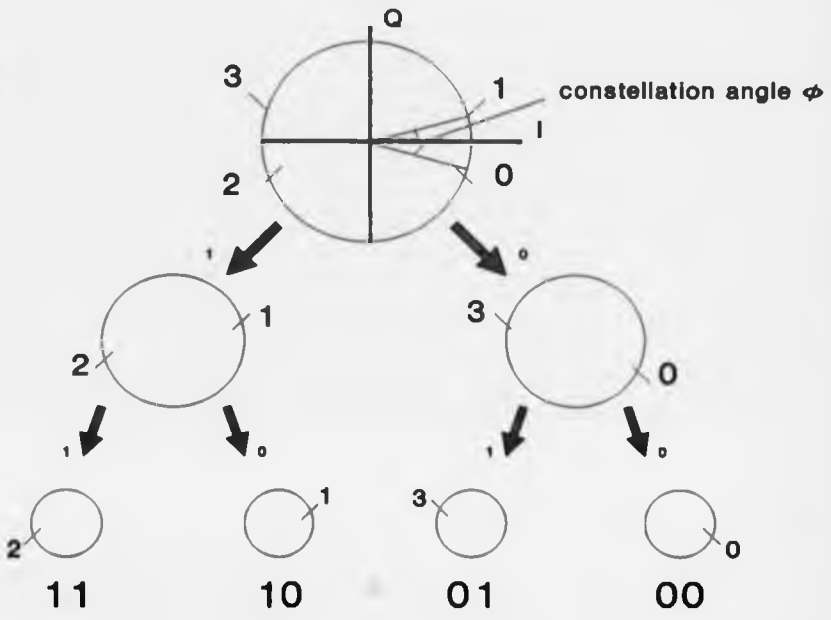


Figure 3.2: Set partitioning of asymmetrical 4-PSK

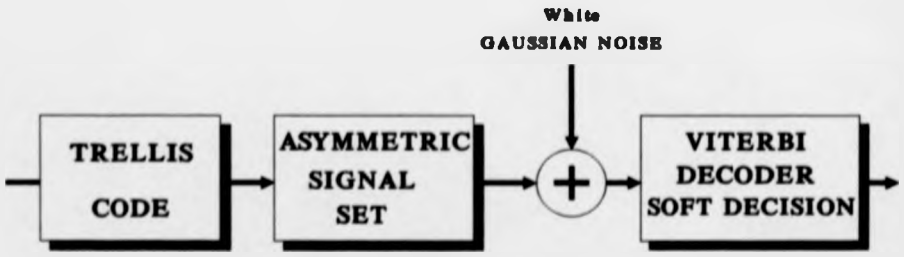


Figure 3.3: System block diagram

3.2.1 Performance Analysis

The block diagram of the system under consideration is shown in Fig. 3.3. We regard an encoder simply as a finite-state machine with a given number of states and specified state transitions. The encoder under consideration is shown in Fig. 3.4. The 4-PSK signal assignment using the trellis diagram is

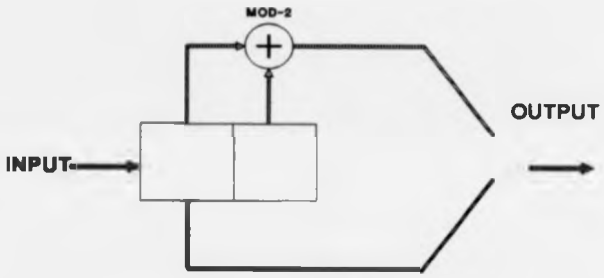


Figure 3.4: Half rate convolutional encoder

shown in Fig. 3.5. If m bits are to be encoded per modulation interval T, there must be 2^m possible transitions from each state to a successor state. We assign channel signals from an extended set of 2^{m+1} signals to the transitions such as to achieve maximum free ED (d_{free}). In order to find d_{free} in the two state trellis diagram of Fig. 3.5, let d_j^2 denote the squared distance from signal point 0 to signal point $j = 0, 1, 2, 3$. Then for the asymmetric constellation of Fig.3.5;

$$d_0^2 = 0;\ d_1^2 = 4\sin^2\frac{\varphi}{2};\ d_2^2 = 4\cos^2\frac{\varphi}{2};\ d_3^2 = 4 \tag{3.1}$$

For the minimum distance path of length 2, we have

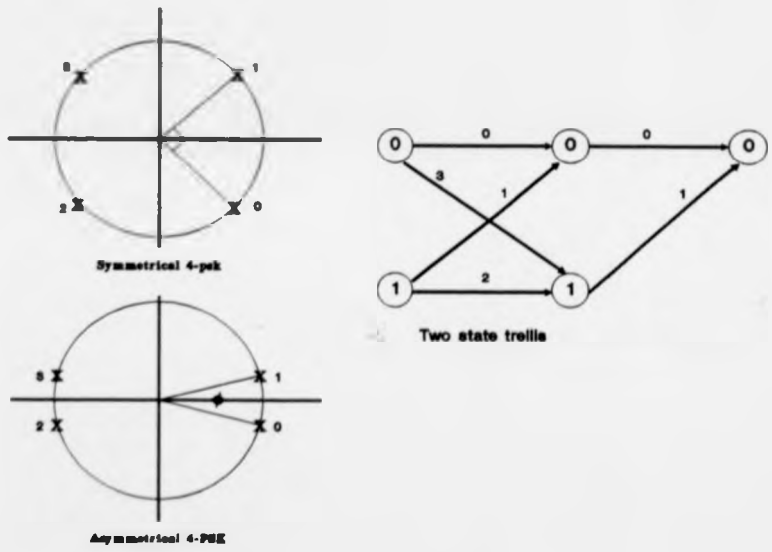


Figure 3.5: 4-PSK signal assignment

$$d^2_{free} = d^2_3 + d^2_1 = 4\left(1 + \sin^2\frac{\phi}{2}\right) \tag{3.2}$$

which for the symmetric signal design ($\phi = \pi/2$) is equal to 6 . The above equation is maximised when $\phi = \pi$; that is, signal points 0 and 2 merge together, and similarly for signal point 1 and 3. In this limiting case, $d^2_{free} = 8$ and the asymptotic energy gain in terms of squared free distance relative to the symmetric constellation is 10 log(8/6) = 1.25 dB. This limiting case results in a catastrophic trellis code and in practice, one would not use this limiting case. Thus we should obtain the practical gain achievable with asymmetry. The most common approach used for the evaluation of practical gain is error probability. It should be mentioned that error probability is a function that decreases when d_{free} increases. This fact shows that the free Euclidean distance is the most significant parameter useful for comparing TCM schemes employed for transmission over the additive white Gaussian noise channel when the signal to noise ratio is large.

3.2.2 Computation of Probability of Error Using State-Transition Diagram

Although the trellis diagram in Fig. 3.5 can be used in computation of probability of error in a straight forward way by labelling the paths with distances from the all-zero distance in the case of all-zero input sequence, a more direct closed-form expression can be obtained by using the state transition diagram

of Fig. 3.6. The state transition diagram is made up of two true correct state

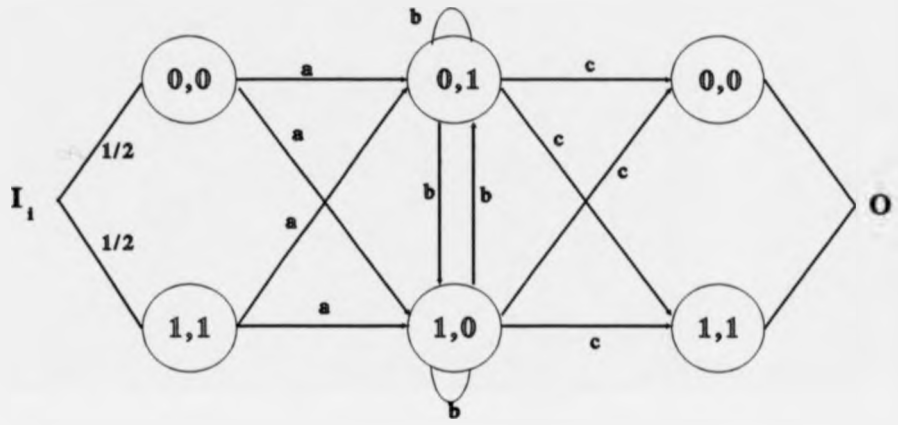


Figure 3.6: State transition diagram

pairs (shown on left and right hand side in Fig.3.6) and a dummy incorrect pair of states (shown in the middle). The two numbers shown in each state, represent the transmitted and the received information. Therefore, the case $0,0$ and $1,1$ represent the correct state and $1,0$ or $0,1$ represent the error event. The state diagram in this form represent all the possible incorrect transitions. By appropriate labelling the branches of this diagram, one can obtain information regarding the squared Euclidean distance between 4-PSK output symbols for the transition between the pair states for all different possible paths between each state in the trellis diagram.

The pair state S (i.e. each circle in the diagram) is defined as [Viterbi *et al.* (1979)]

$$S_k = (s_k, \bar{s}_k) \tag{3.3}$$

We are in a correct pair state when

$$s_k = \bar{s}_k \tag{3.4}$$

In the pair state diagram, the branches are labelled in the form [Divsalar, (1988)]

$$D_b^{d^2} \tag{3.5}$$

Here d^2 is the squared Euclidean distance between MPSK output symbols for the transition between pair states. The parameter D_b is the Bhattacharyya distance [Viterbi *et al.* (1979)] and is related to the pairwise error probability, i.e. $P_E(s_k \mapsto \bar{s}_k)$ when s_k is sent and $\bar{s}_k$ is received. The lower bound of P_E is called the Bhattacharyya bound [Viterbi *et al.* (1979)]. In the state transition diagram of Fig.3.6, I, and O represent the beginning and end (input & output) of transitions between states respectively.

In terms of the above definitions, for the pair state transition diagram in Fig. 3.6 we have :

$$T(D_b, I_i) = \frac{4ac}{(1-2b)} \tag{3.6}$$

where $T(D, I)$ is the transfer function of the state transition diagram of the

TCM scheme shown in Fig. 3.6 and

$$a = \frac{I_i}{2} D_b^4;\ b = \frac{I_i}{2} D_b^{\frac{4}{1+\alpha}};\ c = \frac{1}{2} D_b^{\frac{4\alpha}{1+\alpha}} \tag{3.7}$$

where α is the ratio of powers in each I and Q channel

$$\alpha = \frac{P_Q}{P_I} = tan^2\left(\frac{\phi}{2}\right) \tag{3.8}$$

and D_b can be expressed in terms of noise variance [Viterbi *et al.* (1979)]:

$$D_b = exp\left(-\frac{1}{8\sigma^2}\right) \tag{3.9}$$

The parameter D_b can be related to the system bit energy to noise ratio E_b/N_0 by first recognising that the noise variance is [Viterbi *et al.* (1979)]

$$\sigma^2 = \left(\frac{2E_s}{N_0}\right)^{-1} \tag{3.10}$$

where E_s is the symbol energy. Since we use a half rate encoder, $E_s = 2E_b$, therefore:

$$D_b = exp\left(\frac{-2E_b}{4N_0}\right) \tag{3.11}$$

By substituting eq.3.7 in eq.3.6 we get

$$T(D_b, I_i) = \frac{I D_b^{\frac{4(1+2\alpha)}{(1+\alpha)}}}{1 - I D_b^{\frac{4}{(1+\alpha)}}} \tag{3.12}$$

In terms of the above definitions, the upper bound on the bit error rate is [Omura, (1981)].

$$P_b \leq \frac{1}{n}\frac{d}{dI_i}\, T(D_b, I_i)|_{I_i=1} \tag{3.13}$$

Substituting eq.3.12 in eq.3.13 and performing the differentiation required, yields the desired upper bound on P_b, namely,

$$P_b \leq \frac{D_b^{\frac{4(1+2\alpha)}{(1+\alpha)}}}{\left(1 - D_b^{\frac{4}{(1+\alpha)}}\right)^2} \tag{3.14}$$

Now we choose α to minimize P_b. The optimum α depends on the signal to noise ratio via the parameter D_b. The optimum α is

$$\alpha = -4\left(\frac{ln(D_b)}{ln(3)}\right) - 1 \tag{3.15}$$

Fig. 3.7 shows the optimum ϕ for various values of E_b/N_0. Extensive computer simulation test have been carried out using different angles between constellation points, to determine their relative tolerance to noise. Results of these tests are shown in Figs. 3.8, 3.9, 3.10 and 3.11. It can be seen from the results, that at E_b/N_0 of up to 1.0 dB, the best choice of angle between 30, 90, and 120 is 30 degrees and at a high value of E_b/N_0, i.e. 7 or 8 dB, the obvious choice is 120 deg. Since for uncoded BPSK the square of the minimum distance is four (two signal points diametrically opposed on a circle of diameter 2), the limiting gain of the two-state trellis-coded asymmetric 4-PSK relative to the equivalent bandwidth uncoded system is

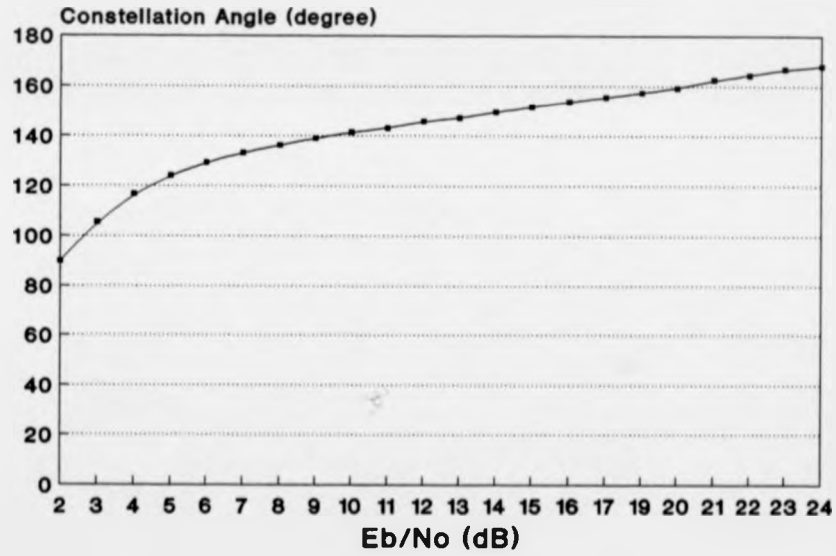


Figure 3.7: Optimum constellation angle vs. E_b/N_0 which minimises BER

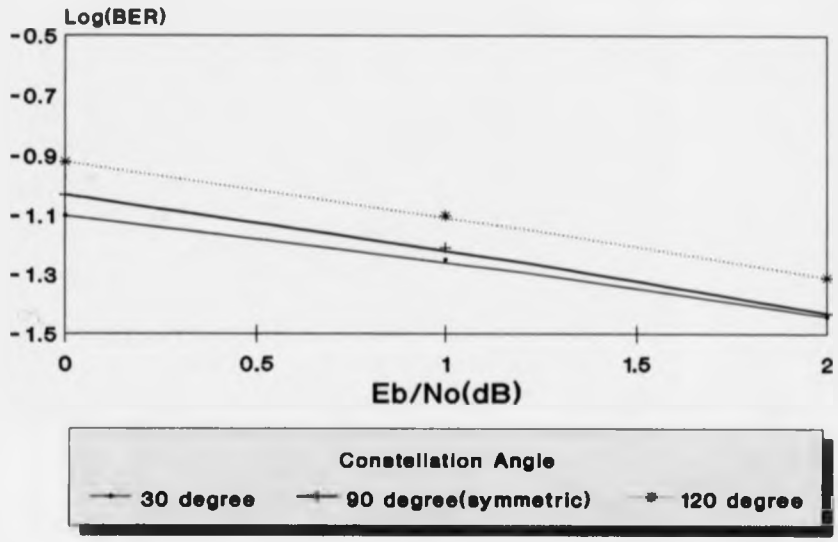


Figure 3.8: Optimum angle which minimises BER for A4PSK (0-2 dB)

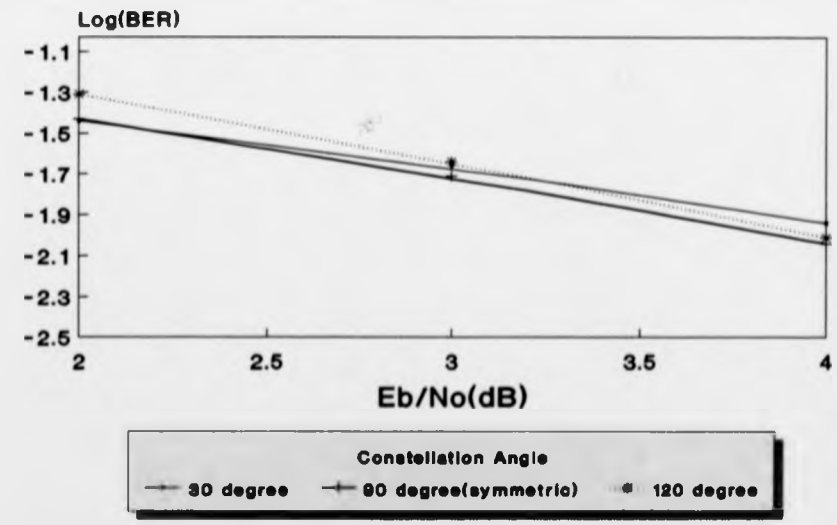


Figure 3.9: Optimum angle which minimises BER for A4PSK (2-4 dB)

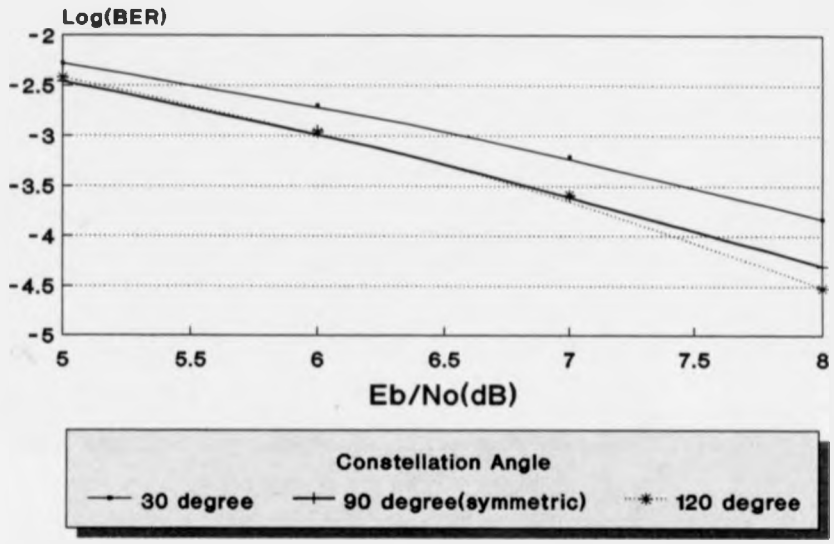


Figure 3.10: Optimum angle which minimises BER for A4PSK (5-8 dB)

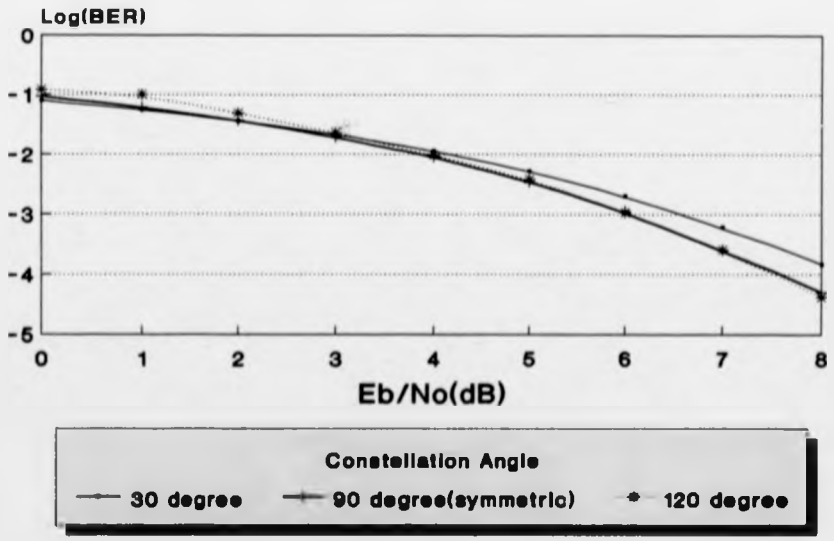


Figure 3.11: Optimum angle which minimises BER for A4PSK (0-8 dB)

$10 \log 10(8/4) = 3.0dB$. This gain corresponds to the case where $\phi = 180$; that is, signal points 1 and 3 merge together, and similarly for signal points 0 and 2. One problem with having signal points too close together in the constellation is that the system becomes less robust in the presence of carrier phase synchronisation errors i.e. the asymmetrical TCM is much less tolerant of carrier phase jitter. There is another technique which makes it possible to achieve the same type of asymptotic performance gains without having to resort to modulation asymmetry. This is by the use of multidimensional constellations which will be described in Chapter 4.

3.3 Trellis Coding Employing Block Codes

It is known that convolutional codes with Viterbi decoding have powerful error capabilities in the AWGN channel as well as Rayleigh fading channels [Hagenauer, (1980), Hagenauer *et al.* (1988)]. A typical feature of the convolutional codes is that each coded bit at any given time unit depends on the previous input symbol sequence of the code memory length. It should be also noticed that this feature can be a drawback in some channels such as when very slow shadow fading is superimposed on Rayleigh fading [Clarke, (1968)]. For such cases, block codes are preferable because the decoding process of a received code block is independent of any other block. However, the decoding performance of a block codes with conventional hard decision decoding (minimum distance decoding) seems to be insufficient for high quality data

transmission. It is well known that soft decision decoding of block codes offers better performance [Wolf, (1978)]. This is because it uses channel measurement information for estimating the reliability of received bits, while hard decision decoding uses only the algebraic redundancy of the codes. An implementation problem of soft decision decoding is decoder complexity.

Many efficient algorithms have been found for using channel measurement information (i.e., soft decisions) in the decoding of convolutional codes than in the decoding of block codes. Most soft decision algorithms for the block codes are complex. However, Wolf [Wolf, (1978)] has presented a method to construct a trellis for linear block codes such as Bose-Chauduri-Hocquenghen (BCH) codes. This allows the Viterbi algorithm to be applied using the trellis, and the decoding performance is improved over hard decision decoding. Other work by Bahl et al [Bahl *et al*, (1974)], Hartmann et al [Hartmann *et al*, (1976)], Miyakawa and Kaneko [Miyakawa *et al*, (1975)] for soft decision maximum likelihood decoding of block codes appears to require a complex decoder. Others [Chase, (1972), Weldon, (1976), Wainberg *et al*, (1973), Dorsch, (1970)] have given sub-optimum decoding algorithms which are relatively simple, but do not achieve maximum likelihood decoding.

In the next section, a low complexity decoding technique for a class of block codes, called array codes, using channel measurement information, which has recently been proposed [Honary *et al*, (1993)] is described.

3.3.1 Decoding an Array Code Using a Trellis

In an array code [Farrell, (1979)], the information (source) word is composed of k binary digits, and this is formed as a k-dimensional binary vector m to form the code vector c. A parity check vector of $r = n - k$ bits is appended to each message vector m to form the code vector c. The vector c has dimension n. The code rate is defined as $R_c = k/n$. The code vector c is then transmitted over the channel. Such a code is denoted as (n, k, d^H_{min}) code, where d^H_{min} is the minimum Hamming distance of the code. This code can correct $\lfloor (d^H_{min} - 1)/2 \rfloor$ errors, where $\lfloor x \rfloor$ represents the largest integer less than or equal to x. The parameters (n, k, d^H_{min}) of an array code are the products of the corresponding parameters of the single parity check codes $(n_1, n_2, k_1, k_2, 2, 2)$, where n_1 and n_2 are the row and column length of the array respectively, and $k_1 = n_1 - 1$, $k_2 = n_2 - 1$. The parity check equations can be written in vector matrix form: $c = mT$, where $c = [c_1, c_2, \cdots, c_n]$, $m = [i_1, i_2, ..., i_k]$, and $T = [I_k \mid P]$. Here m is the information vector, I_k is a $k \times k$ identity matrix, and P is the $k \times r$ parity check array. The error correction and error detection capabilities can be determined from the minimum Hamming distance d^H_{min}, of the code. The Hamming distance is given by

$$d^H(c_i, c_j) = \sum_{k=1}^{n} (c_i)_k \oplus (c_j)_k \tag{3.16}$$

where $(c_i)_k$ is the kth component of the vector c_i. Then

$$d^*_{min} = \min_{i \neq j} d^H(c_i, c_j) \tag{3.17}$$

Consider the following array code

$$\begin{vmatrix} i_1 & i_2 & p_1 \\ i_3 & i_4 & p_2 \\ p_3 & p_4 & p_5 \end{vmatrix} \tag{3.18}$$

Here $p_1 = i_1 \oplus i_2$, $p_2 = i_3 \oplus i_4$, $p_3 = i_1 \oplus i_3$, $p_4 = i_2 \oplus i_4$ and $p_5 = p_1 \oplus p_2$ and (i_1, i_2, i_3, i_4) are the source or information bits. This array is then transmitted over the channel on a row-by-row basis. The code rate is $\frac{4}{9} = (\frac{2}{3})^2$, that is, the code rate of the $(3, 2)$ parity check code squared. In the conventional method of decoding, using hard decision, the code word is received and the blocks of nine detected bits are entered into the 3-by-3 array above. If row i and column j do not satisfy the parity check equations, then the $(i, j)th$ element is complemented. In this manner the decoded array is derived. Clearly, a single error in any one of nine positions can always be corrected. This is in keeping with the result $t = \lfloor (d^H_{min} - 1)/2 \rfloor$ as $d^H_{min} = 4$. For an array code to be decoded with a soft decision maximum likelihood decoding algorithm, it is desirable that the code have a trellis structure so that the Viterbi algorithm can be applied.

The number of states N_s in the trellis of an array code is given by:

$$N_s = q^{\|n_1-1,n_2-1\|} \tag{3.19}$$

where q is a cardinality of the code symbol set, normally the integers modulo q. In the binary case, $q = 2$; and $\| \ \|$ represents the smallest of the two values. The depth of the trellis is the minimum number of rows or columns, i.e n_1 or n_2The trellis structure for the (c_i, c_j) array code i.e $(n_1, k_1)(n_2, k_2)$ is shown in Fig. 3.12. Each transition between the nodes (states) in the trellis

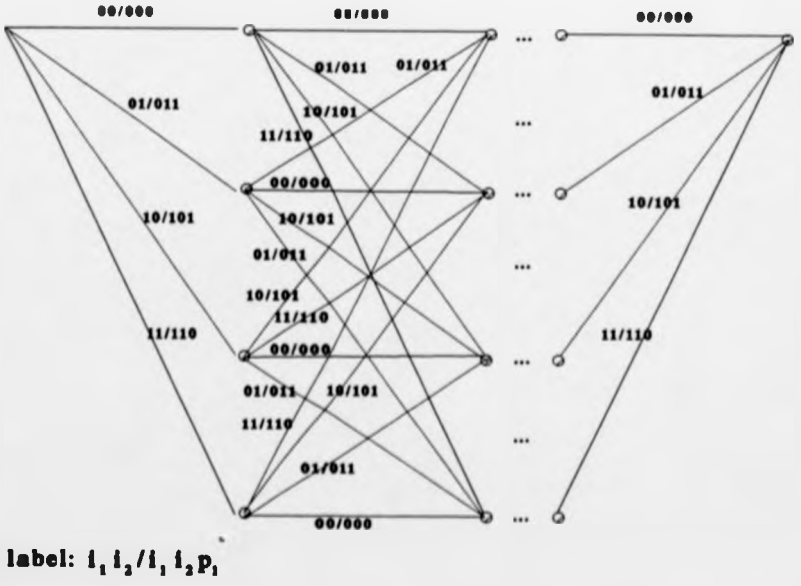


Figure 3.12: Trellis structure for the (c_i, c_j) array code

corresponds to a symbol in a codeword. Each distinct codeword corresponds to a distinct path through the trellis. At depth 0, all codewords start from the same node and at depth n_2 where $n_2 \leq n_1$, all the paths merge together. These paths represent parity bits which do not contain any information.

In order to use the Viterbi algorithm, we need to define a path metric. Let $X(c) = [x(c_1), \cdots, x(c_n)]$ be the sequence of the transmitted codeword and r the received sequence.

The use of soft decision decoding implies that the received sequence r should be the sequence of unquantised demodulator outputs. The likelihood function equals the conditional probability density $p(r \mid c)$. For AWGN channel, maximising $p(r \mid c)$ is equivalent to minimising the squared Euclidean distance $\mid r - x(c) \mid^2$. The path metric $Z(c)$ can be defined as [Clark *et al.* (1988)]:

$$\begin{aligned} Z(c) &= \sum_{i=1}^{n} z(r_i, c_i) \\ &= \sum_{i=1}^{n} \mid r_i - x(c_i) \mid^2 \end{aligned} \tag{3.20}$$

It is now straightforward to use the Viterbi algorithm using $\mid r_i - x(c_i) \mid^2$ as branch metrics. The Viterbi algorithm finds the codeword c that minimises $Z(c)$. It has been shown [Honary *et al.* (1993)] by computer simulation that the soft decision trellis decoding has more than $2dB$ coding gain over the conventional hard decision decoding. This is illustrated in Fig. 3.13 for the (3, 2)(3, 2) array code.

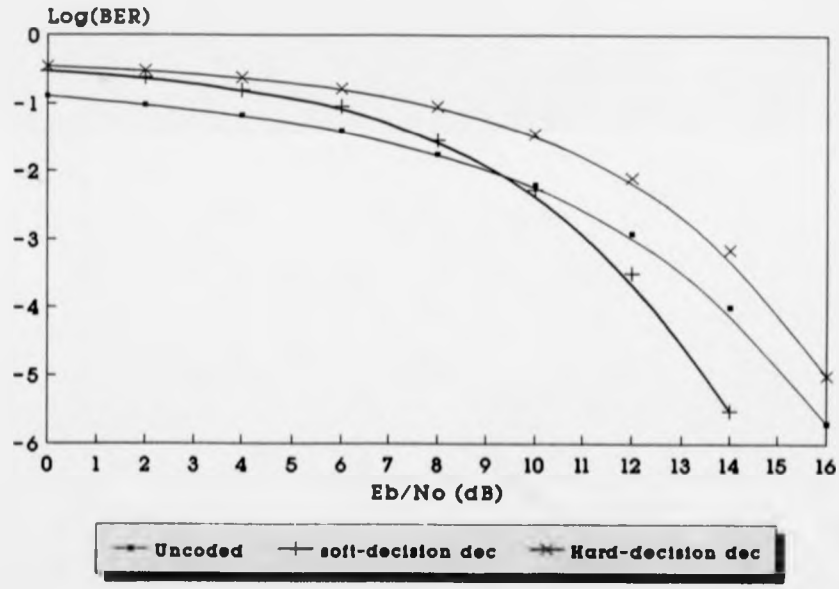


Figure 3.13: Simulation results for a (3,2)(3,2) array code

3.3.2 Block Coded Modulation

Conventional TCM is usually decoded by the Viterbi algorithm. In the previous section, it was shown that the same algorithm can be used for array codes. For the Viterbi algorithm, the branch metric (i.e. squared Euclidean distance) must be computed for each transition branch and all metrics of the paths joining into the same state be compared. An important parameter in TCM is the squared free Euclidean distance (d^2_{free}) and maximisation of d^2_{free} results in coding gain [Clark *et al.* (1988)]. Consider the use of 4-PSK signal constellation in conjunction with the (3,2)(3,2) trellis array code. In order to maximise d^2_{free}, the Ungerboeck set partitioning method [Ungerboeck, (1982)] can be used. This is shown in Fig. 3.14. After partitioning the original two dimensional signal constellation, the minimum squared Euclidean distance between signals within the same subset is greater than that before partition. In Fig. 3.14 $d^2_0 = 2$ and $d^2_1 = 4$ which is twice the original distance. Each branch in the trellis corresponds to one of the subsets B_0 and B_1. For the four point constellation, each of these subsets contain two signal points. In the trellis structure for the $(3,2)(3,2)$ array code (Fig. 3.12), there are four distinct transitions from each state to all other states. There are three paths that diverge from zero state and later remerge at the same state (error paths) namely, $(c_1, c_1), (c_2, c_2)$ and (c_3, c_3) i.e. $(011, 011), (101, 101)$ and $(110, 110)$ assuming the all zero sequence been transmitted. The d^2_{free} of the above paths are 4, 8 and 4 respectively. There-

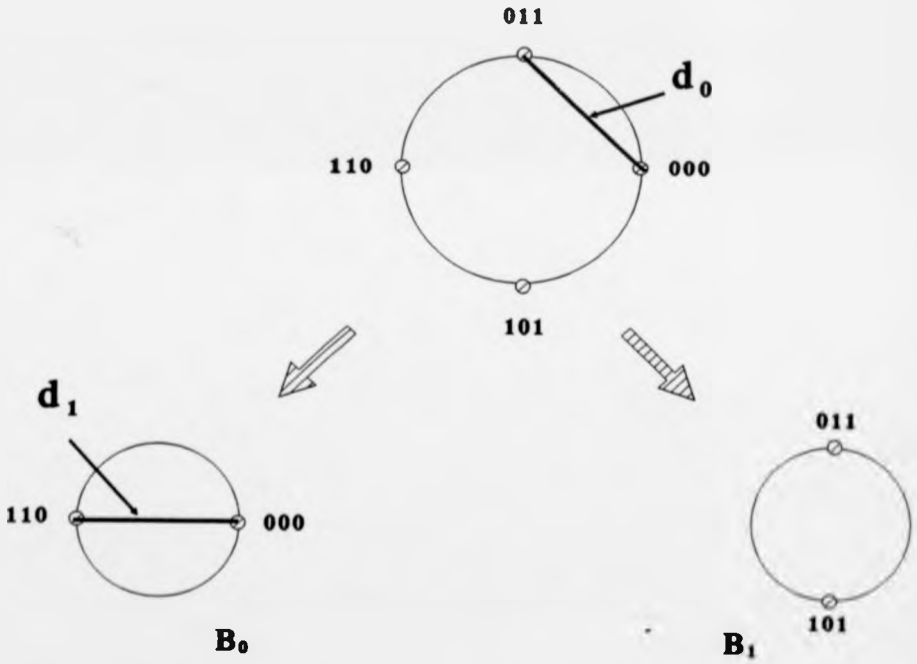


Figure 3.14: Set partitioning for (3,2)(3,2) array code

fore, the squared minimum free Euclidean distance of code is 4. This parameter cannot be further optimised by using set partitioning method in 2-D space, because the principle behind the set partitioning is to maximise ED between the transitions originating from and merging into any state by assigning the appropriate subsets. It can be seen that there are three subsets; i.e. (011),(101) and (110) which all diverge and later remerge to state zero and have an error path of length two. Therefore, the four state trellis diagram illustrated in Fig. 3.12 is optimum in the sense that it provides the largest free Euclidean distance for the given complexity. The coding gain is defined as

$$G = 10\log_{10}\frac{d_{free}^2}{d_0^2} \tag{3.21}$$

where d_0^2 is the minimum Euclidean distance of the MPSK constellation. Therefore the asymptotic coding gain achieved in the above example is

$$\begin{aligned} G &= 10\log_{10}\frac{2d_0^2}{d_0^2} \\ &= 3dB \end{aligned}$$

In BCM employing an array code $(3,2)(3,2)$ as an example, there are four information bits and each block of data, i.e. each row, can be mapped onto the 4-PSK signal constellation. In order to transmit each codeword, three symbols are required. The code rate is $(\frac{2}{3})^2 = \frac{4}{9}$. Hence, we require 3 symbols for transmission of 4 information bits. Therefore, the average transmission

rate is $\frac{4}{3}$ bits per signal interval. An equivalent system in terms of complexity (number of states) employing a convolutional code is a half-rate code (constraint length of 3) in conjunction with 4-PSK modulation, which has a transmission rate in 1 information bits per signal interval. Therefore, in BCM using an array code, the transmission rate may be increased by 33%. The transmission rate of BCM employing an array code increases as the number of information digits in the whole array structure increases. It has been shown that [Honary *et al*, (1993)], the minimum rate in information bits per channel symbol for a given complexity is $\log_2 M$ for an M-ary signalling set.

The performance of $(3,2)(3,2)$ array code combined with 4-PSK modulation scheme is shown in Fig. 3.15.

The performance of BCM employing array code in terms of bit error rate is poorer than that of TCM using convolutional code with the same number of trellis states. The reason behind this is that, we can not match the coded bits to the signal points in the trellis structure for the array code, in order to maximise the squared free Euclidean distance, in the same way as in TCM using a convolutional code, for the reason which was explained earlier.

3.3.3 Asymmetrical Block Coded Modulation

Here, the results in section 3.2 are expanded to asymmetrical block coded modulation. Asymmetrical signal constellations can be applied to BCM in a very similar way to TCM. Here , the effect of asymmetry using array codes

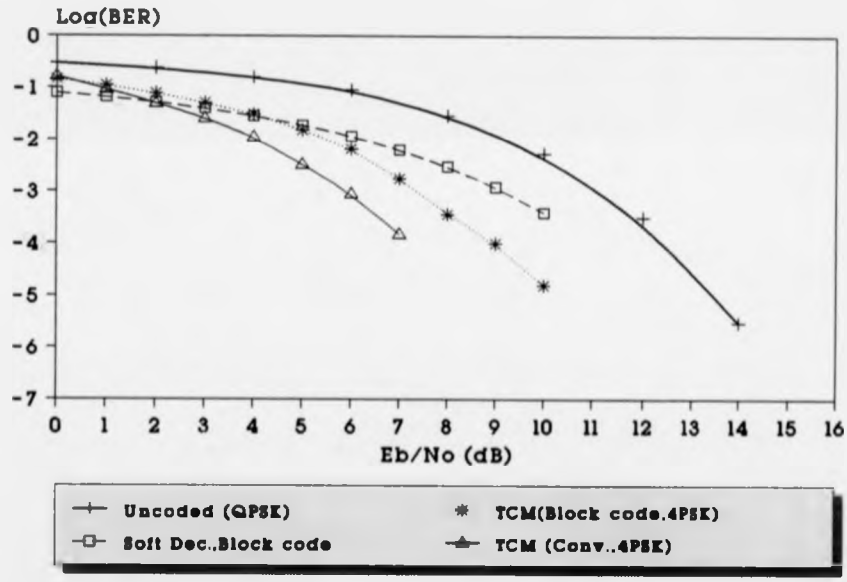


Figure 3.15: Simulation results for BCM employing a (3,2)(3,2) array code

is examined and an optimum constellation for a given array code is found.

Consider the combination of a 4-state trellis code as in Fig. 3.12 and the asymmetric 4-PSK signal set of Figure 3.16, with the (3, 2)(3, 2) array code. By inspection, the following normalised (unit radius circle) squared distances can be obtained:

$$d^2(S_{000}, S_{011}) = 4sin^2\frac{\phi}{2} \tag{3.22}$$

$$d^2(S_{000}, S_{101}) = 4cos^2\frac{\phi}{2} \tag{3.23}$$

$$d^2(S_{000}, S_{110}) = 4 \tag{3.24}$$

From figure 3.12, assuming the all zero path as the correct path, it can be

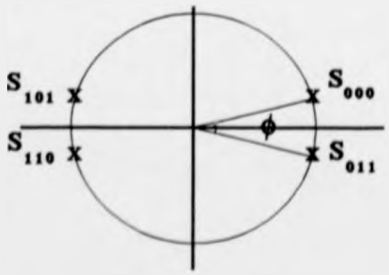


Figure 3.16: Asymmetrical signal set for (3,2)(3,2) array code

seen that there are twelve errors paths that diverge and remerge from the all zero path. Their corresponding squared free distances are shown in Table 3.1.

	path	d^2_{free}
Length 2	C_1C_1	$8\sin^2\frac{\phi}{2}$
	C_2C_2	$8\cos^2\frac{\phi}{2}$
	C_3C_3	8
Length 3	$C_1C_0C_1$	$8\sin^2\frac{\phi}{2}$
	$C_1C_3C_2$	8
	$C_1C_2C_3$	8
	$C_2C_3C_1$	8
	$C_2C_0C_2$	$8\cos^2\frac{\phi}{2}$
	$C_2C_1C_3$	8
	$C_3C_2C_1$	8
	$C_3C_1C_2$	8
	$C_3C_0C_3$	8

Table 3.1: Error paths and their corresponding d^2_{free} distances

To optimise the asymmetric angle, we equate:

$$\sin^2\frac{\phi}{2} = \cos^2\frac{\phi}{2}$$

which results in $\phi = 90$.

Therefore the optimum constellation angle for BCM employing a $(3,2)(3,2)$ array code, 4-state trellis code and 4-PSK signal set is 90 degrees.

Chapter 4

Multi-dimensional Trellis Coded Modulation

4.1 Introduction

The coded modulation format for multi-dimensional TCM which is under study defined over an expanded set of signals varying both in phase and frequency. This scheme is a combination of MFSK and MPSK modulation and the signal points are on a sphere in four dimensional Euclidean space. The performance has been compared with comparable convolutional encoders in the binary field [Arani *et al.* (1991a), Arani *et al.* (1993)]. It is well known that the performance of a TCM scheme, based on a given signal constellation, can be improved by increasing the number of states [Viterbi *et al.* (1979)]. In other words, coding gain is a function of the amount of memory introduced by the encoder, which is the constraint length. However, as the number of states exceeds a certain value, the coding gain increases more slowly [Slepian, (1963)]. Thus, if the signal constellation is not changed, coding gain progressively diminishes. In this chapter, we consider the structure of signal constellations in four dimensional space. Motivation for the use of multidimensional signals for digital transmission dates back to the work of Shannon [Shannon, (1949)]. He recognised that the performance of a signal constellation used to transmit digital information over the additive white Gaussian noise (AWGN) channel can be improved by increasing the dimensionality of the signal set used for transmission. In particular, as the dimension number grows to infinity, the performance tends to an upper limit that defines the capacity of the channel [Slepian, (1963)].

The problem of signal design has been addressed by numerous investigators. It has been shown that a performance improvement can be obtained [Slepian, (1963), Wilson *et al*, (1984), Wilson *et al*, (1983), Gersho *et al*, (1984), Welti *et al*, (1974), Zetterberg *et al*, (1977), Slepian, (1968), Biglieri, (1984), Jacobs, (1967), Calderbank *et al*, (1987)]. Uncoded four dimensional signal sets were considered by Welti and Lee [Wilson *et al*, (1984)], Zetterberg and Brandstorm [Zetterberg *et al*, (1977)], Wilson and Sleeper [Wilson *et al*, (1983)] and Biglieri [Biglieri, (1984)]. They show a 1.2 to 2.4 dB gain in noise margin over conventional two dimensional modulation. Four dimensional signals have a special appeal, because besides being the closest thing to the widely used two dimensional sets, they can be implemented in radio communications without any increase in bandwidth, by utilising two spatially orthogonal electric field polarisations for transmitting on the same carrier frequency.

Multidimensional alphabets in conjunction with TCM can provide us with several positive features. Larger dimensionality provides more room for the signals. As a consequence, larger Euclidean distances can be obtained. Thus an increased noise margin may be derived from the multidimensional constellation itself. An inherent cost with one and two dimensional TCM schemes is that when the size of the constellation is doubled over the uncoded schemes, it may occur that the average energy needed for signalling increases. For example, with two dimensional constellation (QAM) doubling the constellation size makes us lose roughly 3 dB [Biglieri *et al*, (1988)]. Without this increase in energy expenditure, the coding gain of TCM would be greater. By using a

multidimensional constellation, this cost falls from 3 dB for two dimensions to 1.5 dB for four dimensions [Slepian, (1968)], because, for four dimensional constellations the average energy is multiplied by about $\sqrt{2}$ when the number of points in the constellation is doubled.

A simple way of generating a multidimensional constellation is by time division. For example, if N two dimensional signals are sent in a time interval of duration T, and each has duration T/N, we obtain a $2N$ dimensional constellation.

Most research on trellis coded modulation has been focused primarily on binary convolutional codes. The performance improvement by using non-binary codes over fading or partial interference channels is well known [Viterbi *et al.* (1975)], and classes of codes for this application have been proposed by Viterbi [Viterbi, (1972), Viterbi *et al.* (1975), Viterbi, (1967)]. In [Viterbi, (1972)], the so called dual-k codes are introduced and have been analysed by Odenwalder [Oldenwalder, (1976)]. He has described a class of non-binary convolutional codes (dual-k) with a constraint length of 2 in which the information and parity symbols are taken as elements of a ring of integers modulo n, Z_n. Performance results for the dual-5 code with 32-ary signalling and noncoherent detection has been shown to provide 1.5 dB coding gain at post-decoding BER of 10^{-5} over uncoded 32-ary signalling.

Traditionally, a general convolutional encoder is mechanised with a L stage shift register and n modulo-2 adders, where L is the constraint length. Here, we use the same arithmetic system but with a larger finite number of

elements than the binary system. As long as the data consists merely of bit packages, it does not matter what arithmetic system is used by the encoder and decoder. Any definition of addition and multiplication, however artificial, can be invented as long as the theory of convolutional codes is still valid in that number system. The only price we pay is that we can no longer use conventional adders and multipliers to carry out the computations. The important properties of a convolutional code are linearity and time invariance. Linearity means that the mod-q sum of any two codewords is a codeword. Time invariance means that the time shift of any codeword is a codeword.

For each positive integer m, there is a set Z of integers modulo q forming a ring Z_q [Peterson *et al.* (1972)], where $q = 2^m$ and so Z_q has 2^m elements. If q is prime then Z_q is a field. The set of elements of Z_q can be represented as the set of m-bit binary numbers. Therefore Z_4 consists of 4 distinct 2-bit bytes.

In conventional trellis coded modulation techniques a rate $n/(n+1)$ convolutional encoder is combined with an $M = 2^{(n+1)}$ signal point constellation through a suitable mapping to produce a coded modulation scheme, where n is the number of transmitted information digits. Here we demonstrate a trellis coded modulation technique which is capable of achieving an additional gain over conventional TCM. The principle behind this technique is to design a rate $mn/m(n+1)$ encoder and combine it with a $q^{(n+1)}$-point signal constellation in 4-dimensional space. Recently, Ryan & Wilson [Ryan *et al.* (1991)] discussed the design and performance of two classes of non-binary convolu-

tional codes. The first class was called q-ary convolutional codes and the second class was binary-to-q-ary convolutional codes. In this chapter, the q-ary convolutional codes are considered and are combined with suitable mapping function with 2^{n+1} point signal constellation outputting n of these signal points one for each group of $(n+1)$ encoder output symbols in each transmission interval. The binary and q-ary convolutional codes of the same constraint length have been studied. The mapping of signals are considered for 4-D space.

4.2 Mapping Procedure

In this section mappings by set partitioning and trellis coding are applied to signals that vary both in phase and frequency. The orthonormal basis for the four dimensional space is the four signals [Arthurs *et al*, (1962)]:

$$\varphi_1(t) = \sqrt{\frac{2E}{T}}\cos(\omega t + \phi_i(t)) \tag{4.1}$$

$$\varphi_2(t) = \sqrt{\frac{2E}{T}}\cos(\omega t - \phi_i(t)) \tag{4.2}$$

$$\varphi_3(t) = \sqrt{\frac{2E}{T}}\cos(2\omega t + \phi_i(t)) \tag{4.3}$$

$$\varphi_4(t) = \sqrt{\frac{2E}{T}}\cos(2\omega t - \phi_i(t)) \tag{4.4}$$

where ω is an angular frequency of the signal and $\phi_i = 2\pi i/M$ for $i = 1, 2, ..., M$ in M-ary signalling. The new scheme is most easily explained by means of an example. Suppose we want to transmit 1 bit/T, where T is the

modulation interval. A popular scheme is BPSK, in which various combinations of the one bit select one of the two phase shifts to be applied to the carrier. Ungerboeck [Ungerboeck, (1982)] couples a rate $R = 1/2$ convolutional encoder and an expanded signal set, namely an 4-PSK constellation, in which different combinations of the two bits at the output of the encoder select one of the four phase shifts. In our scheme, the 1 symbol/T that we

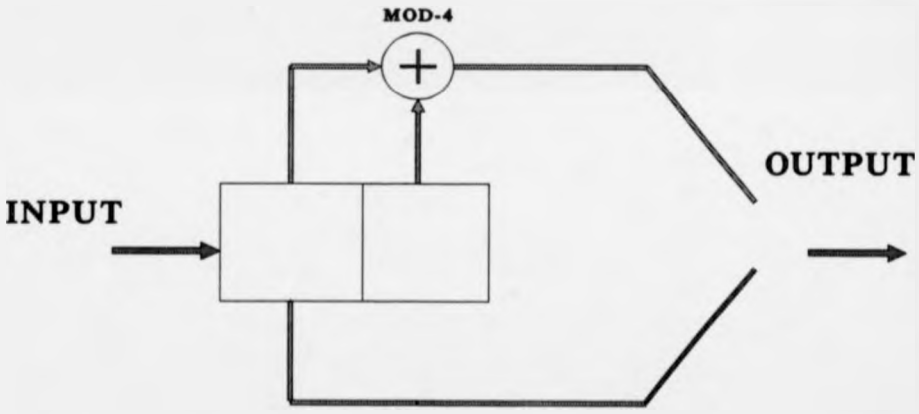


Figure 4.1: Non-binary Z_4 convolutional encoder

want to transmit is fed in to a non-binary Z_4 convolutional encoder as shown in Fig. 4.1 to produce two symbols from the Z_4 set i.e $\{0, 1, 2, 3\}$. Therefore, the output of the encoder can produce one pair of these symbols out of the sixteen possible combinations of them. These combinations are:

$$\{00, 01, 02, 03, 10, 11, 12, 13, 20, 21, 22, 23, 30, 31, 32, 33\} \qquad (4.5)$$

In the q-ary codes, the information symbols are elements of Z_q , and each cell in the shift register stores a q-ary symbol. Clearly, the q-ary encoder has q^{L-1} possible states, where L is the constraint length of the encoder. The trellis diagram for the above encoder is shown in Fig. 4.2. It should be

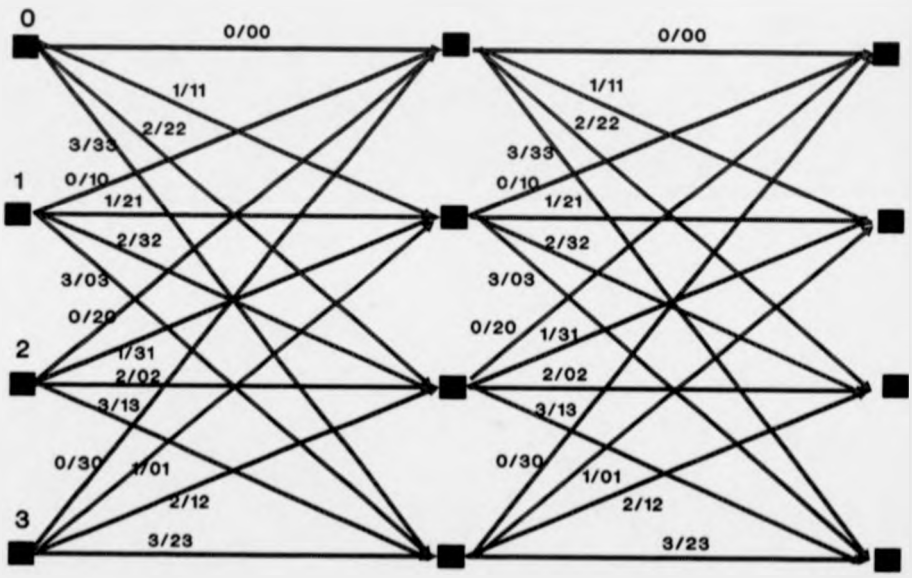


Figure 4.2: Trellis diagram for the Z_4 encoder

noted that the number of branches from each state is four but in the case of the binary convolutional code is two. In the Ungerboeck scheme, in order to transmit n bits, a $2^{(n+1)}$ point signal constellation is used. It is obvious that in Z_q , in order to transmit n symbols, a $q^{(n+1)}$ point signal constellation is required. Therefore, in the case of $q = 4$ and $n = 1$, 16 signal constellation

points are needed. The information source produces every T seconds one of the sixteen equally probable messages $S_m(t)$ $m = 0, ..., 15$ in the form [Arthurs *et al.* (1962)]

$$\begin{aligned} S_0(t) &= \alpha_{01}\varphi_1(t) + \alpha_{02}\varphi_2(t) + \alpha_{03}\varphi_3(t) + \alpha_{04}\varphi_4(t) \\ &\vdots \\ S_n(t) &= \alpha_{n1}\varphi_1(t) + \alpha_{n2}\varphi_2(t) + \alpha_{n3}\varphi_3(t) + \alpha_{n4}\varphi_4(t) \\ &\vdots \\ S_m(t) &= \alpha_{m1}\varphi_1(t) + \alpha_{m2}\varphi_2(t) + \alpha_{m3}\varphi_3(t) + \alpha_{m4}\varphi_4(t) \end{aligned} \tag{4.6}$$

where φ_i for $i = 1, 2, 3, 4$ are as described in Eq. 4.1 to 4.4 and the α_{ij} are real numbers given by the formula

$$\alpha_{ij} = \int_0^T S_i(t)\varphi_j(t)dt \tag{4.7}$$

and $\phi_i \in \{0, \pi/4, \pi/2, 3\pi/4, \pi, 5\pi/4, 3\pi/2, 7\pi/4\}$.

In the case of MPSK,

$$\begin{aligned} \alpha_{i1} &= \sqrt{E}\cos\frac{2\pi i}{M} \\ \alpha_{i2} &= \sqrt{E}\sin\frac{2\pi i}{M} \end{aligned} \tag{4.8}$$

and in the case of MFSK

$$a_{ij} = \begin{cases} \sqrt{E} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

The signal space spanned by such signals is four dimensional, but for the purpose of illustration, we can represent them as points in two different two dimensional planes. The mapping by set partitioning procedure is shown in Fig. 4.3. In this procedure, we begin by partitioning the original signal point constellation into two constellations each with maximum distance among its signal points. Therefore in the case of BFSK-8PSK, assuming $E = 1$ (i.e. unit circle), the following coordinates can be obtained:

$$\begin{aligned}
S_0 &= (1,0,1,0) & S_1 &= (-0.707, 0.707, 0, 1) \\
S_2 &= (-0.707, -0.707, 0, 1) & S_3 &= (0, -1, 1, 0) \\
S_4 &= (0,1,1,0) & S_5 &= (1,0,0,1) \\
S_6 &= (0.707, -0.707, 0, 1) & S_7 &= (-0.707, -0.707, 1, 0) \\
S_8 &= (0.707, 0.707, 1, 0) & S_9 &= (0,1,0,1) \\
S_{10} &= (-1,0,0,1) & S_{11} &= (-0.707, 0.707, 1, 0) \\
S_{12} &= (0.707, -0.707, 1, 0) & S_{13} &= (0.707, 0.707, 0, 1) \\
S_{14} &= (0, -1, 0, 1) & S_{15} &= (-1,0,1,0)
\end{aligned} \tag{4.9}$$

The distance between a pair of points, X, Y, with coordinates $(x_1, x_2, \cdots, x_k)$

and $(y_1, y_2, \cdots, y_k)$, respectively in k dimensional space is given by the formula

$$d(X, Y) = \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2} \tag{4.10}$$

It follows readily that the distance between a signal point S_i with coordinates $(\kappa_{i1}, \kappa_{i2}, \cdots, \kappa_{ik})$ and the origin of the signal space is given by

$$d(S_i, 0) = \sqrt{\sum_{i=1}^{k} \kappa_{ij}^2} \tag{4.11}$$

The mapping of the output symbols in the trellis diagram (Fig. 4.2) to the

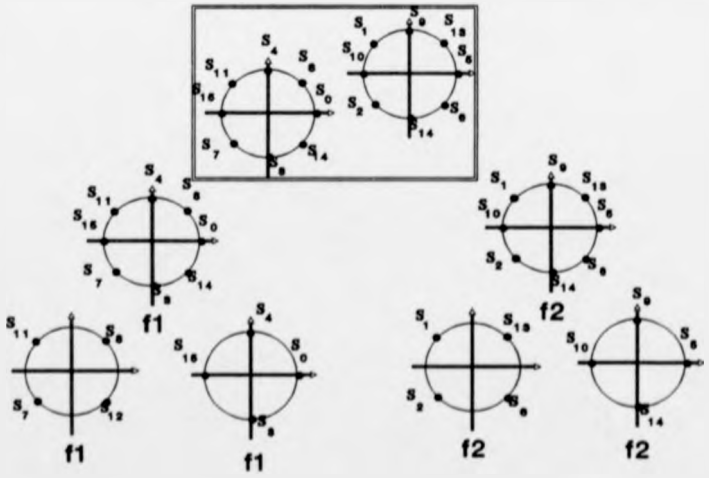


Figure 4.3: Set partitioning of BFSK/8-PSK in 4-D space

set partitioning diagram in Fig. 4.3 is as follow:

$$00 \longmapsto S_0 \quad 11 \longmapsto S_{10} \tag{4.12}$$

$$22 \longmapsto S_5 \quad 33 \longmapsto S_{15}$$

$$10 \longmapsto S_9 \quad 21 \longmapsto S_3$$

$$32 \longmapsto S_4 \quad 03 \longmapsto S_{14}$$

$$20 \longmapsto S_2 \quad 31 \longmapsto S_8$$

$$02 \longmapsto S_7 \quad 13 \longmapsto S_{13} \tag{4.13}$$

$$30 \longmapsto S_1 \quad 01 \longmapsto S_{12}$$

$$12 \longmapsto S_6 \quad 23 \longmapsto S_{11}$$

In Fig. 4.2. assuming the all zero path as the correct path, there are three error paths namely $(11, 10)$, $(22, 20)$ and $(33, 30)$, i.e. (S_{10}, S_9) , (S_5, S_2) and (S_{15}, S_1), which have length of two, leaving state 0 and returning to state 0. Using the above mapping and Eq. 4.10, their distances can be calculated as:

$$d^2_{free}(S_{10}, S_9) = d^2(S_{10}, S_0) + d^2(S_9, S_0) \tag{4.14}$$

$$= 6.0 + 4.0 = 10.0 \tag{4.15}$$

$$d^2_{free}(S_5, S_2) = d^2(S_5, S_0) + d^2(S_2, S_0) \tag{4.16}$$

$$= 2.0 + 5.4 = 7.4 \tag{4.17}$$

$$d^2_{free}(S_{15}, S_1) = d^2(S_{15}, S_0) + d^2(S_1, S_0) \tag{4.18}$$

$$= 4.0 + 5.4 = 9.4 \tag{4.19}$$

Therefore the minimum Euclidean distance in the set of all paths that diverge and reemerge is $d^2_{free}(S_3, S_2) = 7.4$.

4.3 Performance Evaluation

In regard of the complexity of the system , it was mentioned earlier that in Z_q conventional adders and multipliers can no longer be used in order to carry out the computation. Also, it should be noted that, in a Z_q convolutional encoder, each cell in the shift register stores a q-ary symbol. In the performance evaluation, these differences will not be considered in detail and simply the price to pay in terms of hardware complexity by moving from binary to non-binary field is accepted. Our comparison will be based on the rate of the code and the performance of it in terms of free Euclidean distance.

In order to compare a binary convolutional encoder of the same constraint length and rate in which its output can be mapped to the same number of signal points, certain modifications are required. The rate of the Z_4 convolutional encoder is 1/2. At each interval, two binary information bits are encoded into four bits. The binary convolutional encoder of rate 1/2 and its corresponding trellis diagram is shown in Fig. 4.4(a)(b).

The two output bits can be mapped to one of the 4 signals which are available at any instant of time (e.g.4- PSK). But in the case of BFSK-8PSK, we require 16 signals for representation. One way of achieving this, is to consider more consecutive input bits. If we consider 2 input bits, two

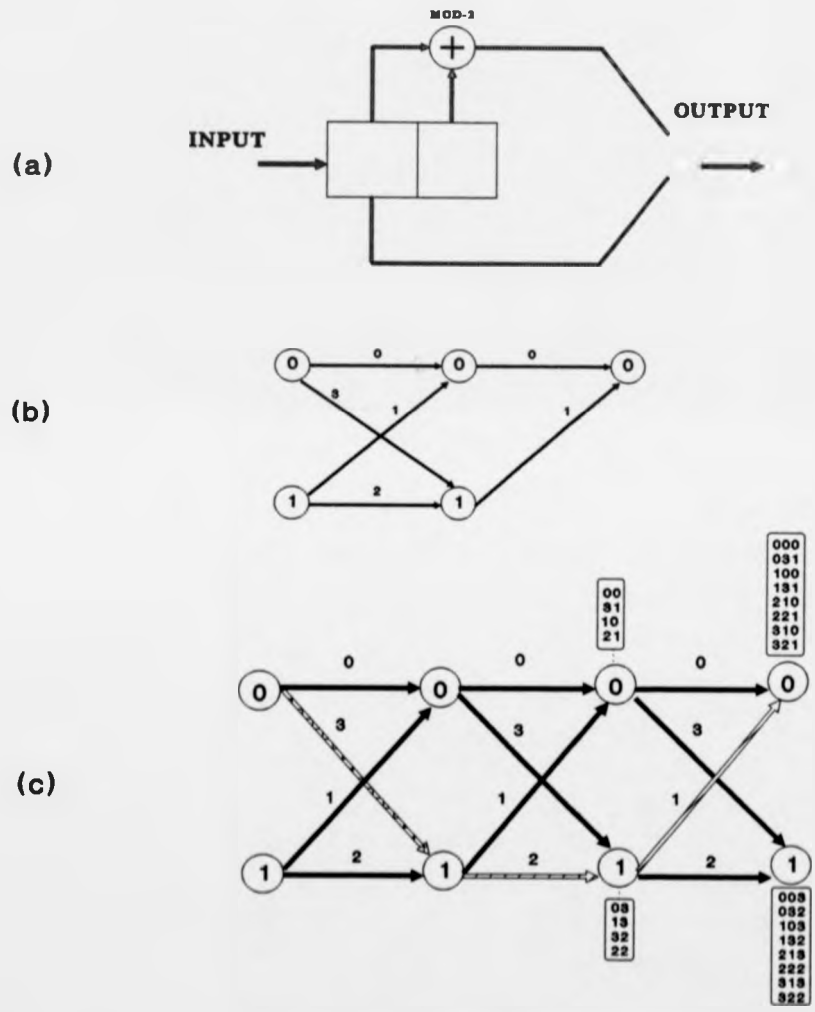


Figure 4.4: Binary convolutional encoder & its corresponding trellis diagram

channel symbols can be produced $\{00, 31, 10, 21, 03, 13, 32, 22\}$[1] , then we only have 8 signal points. Therefore, we consider 3 consecutive input bits which are capable of producing one of the following possible 16 channel symbols (see Fig. 4.4(c)). These are:

$$\{000, 003, 031, 032, 100, 103, 131, 132, 210, 213, 221, 222, 310, 313, 321, 322\}$$

These symbols can be mapped to sixteen symbols in Fig. 4.3.

$$\begin{aligned} 000 &\longmapsto S_0 \quad 210 \longmapsto S_8 \\ 003 &\longmapsto S_1 \quad 213 \longmapsto S_9 \\ 031 &\longmapsto S_2 \quad 221 \longmapsto S_{10} \\ 032 &\longmapsto S_3 \quad 222 \longmapsto S_{11} \\ 100 &\longmapsto S_4 \quad 310 \longmapsto S_{12} \\ 103 &\longmapsto S_5 \quad 313 \longmapsto S_{13} \\ 131 &\longmapsto S_6 \quad 321 \longmapsto S_{14} \\ 132 &\longmapsto S_7 \quad 322 \longmapsto S_{15} \end{aligned} \tag{4.20}$$

Each of these symbols contain 6 bits of which 3 are information. These

[1] Here, 3 and 2 represent 11 and 10 in binary format respectively

16 symbols can be mapped to BFSK-8PSK signal space. This procedure is shown in Fig. 4.5. Assuming the all zero path as the transmitted sequence,

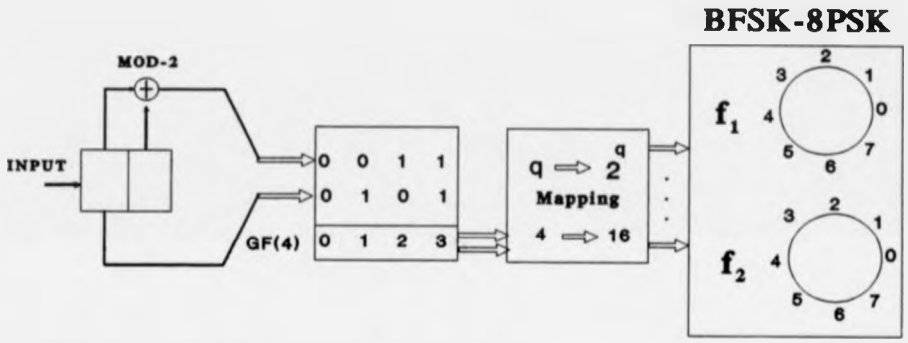


Figure 4.5: Illustration of mapping procedure

then the shortest error path after 3 consecutive input bits in Fig. 4.4(c) is the (3,2,1) path and in the 4-D space[2] its distance is equivalent to the squared distance between symbol 000 and 321 i.e $d^2(S_0, S_{14})$. Using 4.10, this distance can be calculate as:

$$d^2(S_0, S_{14}) = (x_1^0 - y_1^{14})^2 + (x_2^0 - y_2^{14})^2 + (x_3^0 - y_3^{14})^2 + (x_4^0 - y_4^{14})^2 \quad (4.21)$$

where $(x_1^0, \cdots, x_4^0)$ and $(y_1^{14}, \cdots, y_4^{14})$ are co-ordinates of S_0 and S_{14} respec-

[2]Three information bits/3T i.e. 1 bit of transmitted information /T has been considered.

tively. Therefore, the free Euclidean distance is $d^2(S_0, S_{14}) = 4.0$.

The asymptotic coding gain (G), in dB, compared with some suitable reference system is defined as [Viterbi *et al.* (1979)]

$$G = 10\log_{10}\left(\frac{d^2_{free}}{d^2_{ref}}\right) \tag{4.22}$$

where d^2_{ref} and d^2_{free} are the free Euclidean distance of the reference system and the system under consideration. Therefore, the achievable coding gain of TCM by using the convolutional encoder in Z_4 rather than the conventional binary encoder, in 4-D space is

$$G = 10\log(7.4/4.0) = 2.7dB \tag{4.23}$$

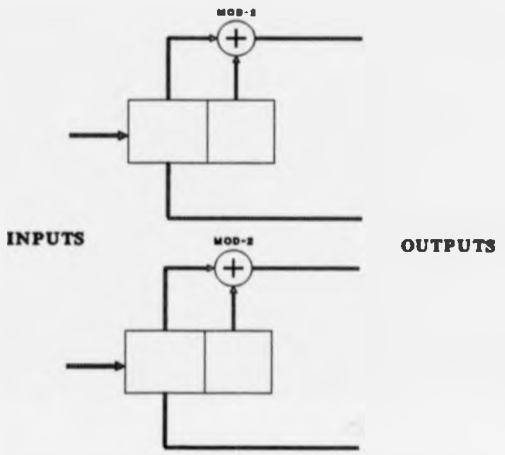
4.3.1 Dual-k Convolutional Code

In the dual-k convolutional encoder shown in Fig. 4.6(a), two encoders identical to the one used previously, are connected in parallel. Two bits are entered into this encoder and 4 encoded bits are produced at the output using two modulo-2 adders. The trellis diagram of this code is shown in Fig. 4.6(b). In this system, there are three error paths that must be considered.

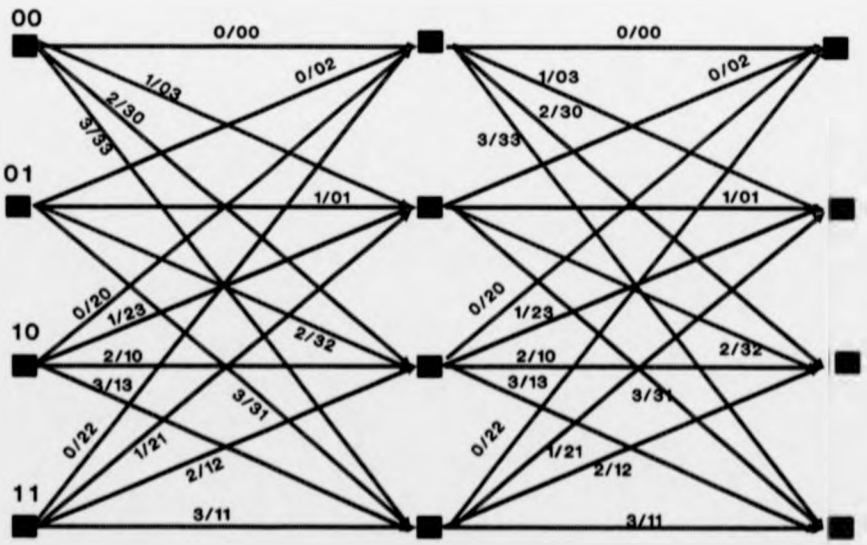
$$d^2_{free}(S_{14}, S_7) = d^2(S_{14}, S_0) + d^2(S_7, S_0) \tag{4.24}$$

$$= 4.0 + 3.4 = 7.4 \tag{4.25}$$

$$d^2_{free}(S_1, S_2) = d^2(S_1, S_0) + d^2(S_2, S_0) \tag{4.26}$$



(a) Binary Convolutional Encoder



(b) Trellis Diagram

Figure 4.6: Trellis diagram for 2/4 rate binary convolutional encoder (dual-k code)

$$= 5.4 + 5.4 = 10.8 \tag{4.27}$$

$$d^2_{free}(S_{15}, S_5) = d^2(S_{15}, S_0) + d^2(S_5, S_0) \tag{4.28}$$

$$= 4.0 + 5.4 = 9.4 \tag{4.29}$$

Therefore, the smallest free Euclidean distance in this case is equal to 7.4. This means that the performance achieved by this dual-2 code is the same as the Z_4 TCM scheme.

4.3.2 Uncoded Transmission

The purpose of coding is to gain noise immunity beyond that provided by standard uncoded transmission at the same data rate. Hence, we compare the performance of TCM over Z_4 with uncoded transmission at the rate of 2 bits/2-dimensional signal shown in Fig. 4.7a.

In this case, all allowable signal sequences are all the elements of Z_4. To achieve uncoded transmission of a 4-dimensional symbol at a rate of 4 bits/symbol, simply two copies of this scheme is taken (Fig. 4.7b). The state transition (trellis) diagram for the uncoded case is shown in Fig.4.7c. This representation is only to illustrate the uncoded scheme from the viewpoint of TCM. Every connected path through the trellis represents an allowed signal sequence. In this case 8 parallel transitions can occur which do not restrict the sequences of BFSK-QPSK signals that can be transmitted, that is, there is no sequence coding. Hence, the optimum decoder can make independent

nearest signal decisions for each noisy BFSK-QPSK signal received. The smallest squared distance (squared free distance) between the BFSK-QPSK signals is 2. Hence, use of 4-dimensional TCM over Z_4 will result in asymptotic coding gain of

$$10\log\frac{(d^2_{free})_{coded}}{(d^2_{free})_{uncoded}} = 10\log\frac{7.4}{2} \tag{4.30}$$
$$= 5.7dB$$

over uncoded BFSK-QPSK signals.

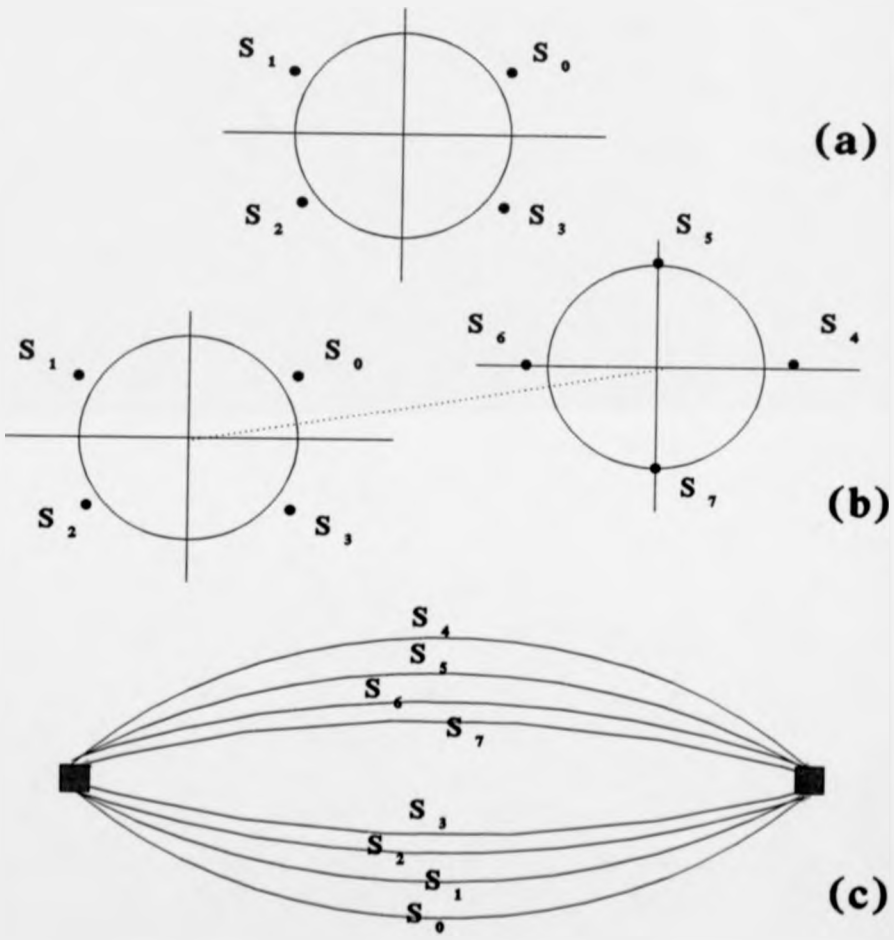


Figure 4.7: Uncoded BFSK-QPSK

Chapter 5

Adaptive Trellis Coded Modulation

5.1 Introduction

Packetised data transmission for mobile satellite or terrestrial radio services in a fading environment achieves reliable transmission if a combination of automatic repeat request (ARQ) and forward error correction (FEC) is used [Lugand *et al*, (1989)].

For data transmission schemes where a return channel is available, ARQ schemes guarantee high reliability for good (low error rate) and moderate channels [Yamamoto *et al*, (1980)]. The throughput performance degrades if the channel is bad or time-varying. A combination with FEC to form a hybrid system combines the advantages of both methods. For this reason, the concept of embedded encoding, based on block codes, was proposed [Darnell *et al*, (1988a), Darnell *et al*, (1988b), Zolghadr, (1989)]. In this encoding scheme, a block of information is divided into streams; each stream is then encoded at a different rate. The resulting encoded block is then transmitted in an ARQ environment [Lin *et al*, (1984)]. The assumption behind the technique is that at least one of the rates will normally be received successfully over a wide range of channel conditions.

In the transmission of a convolutional code, the error probability becomes smaller as the constraint length increases, but the implementation complexity of a Viterbi decoder increases [Viterbi, (1971)]. When a receiver can use a feedback channel to request retransmission, a convolutional code with a shorter constraint length should perform better than a larger one without a

feedback link.

Performance improvements which result from modification of the original embedded code by using convolutional codes and Viterbi decoding have been reported [Zolghadr *et al.* (1988)]. In this scheme, high rate punctured convolutional codes [Cain *et al.* (1979), Yasuda *et al.* (1983)] are used for the purpose of error correction and rate variation. Coding and modulation were treated independently which implies that only two parameters could be adjusted to improve the system performance, namely, transmitted power and channel bandwidth. Specifically, for a fixed error probability, an increased data rate necessitates a wider bandwidth.

With the growth of digital communication systems, bandwidth limitations are progressively becoming more significant. To overcome this problem, it is possible to couple coding with multi-level modulation [Ungerboeck, (1982)]. In this chapter, the novel concept of embedded trellis coded modulation [Honary *et al.* (1990)], which is based on the original embedded encoding [Darnell *et al.* (1988a)] and embedded convolutional code [Zolghadr *et al.* (1988)] is described. The performance of embedded TCM has been analysed using a simulation technique.

5.2 Embedded Trellis Coded Modulation

Trellis coded modulation (TCM), which is a technique wherein a rate $n/(n+1)$ trellis code is combined with a 2^{n+1} point signal constellation to produce

coded modulation has no bandwidth expansion relative to an uncoded 2^n point modulation of the same type, yet gives significant performance improvement [Ungerboeck, (1982)] . Here we demonstrate a TCM technique, combined with the embedded convolutional code [Honary *et al.* (1990)] , in an ARQ environment which employs a real-time channel evaluation (RTCE) [Darnell, (1983)] method for an adaptive system. In embedded trellis-coded modulation an M-ary modulation scheme is used to obtain coding gain on channels with very limited bandwidth. Fig.5.1 illustrates the coded modula-

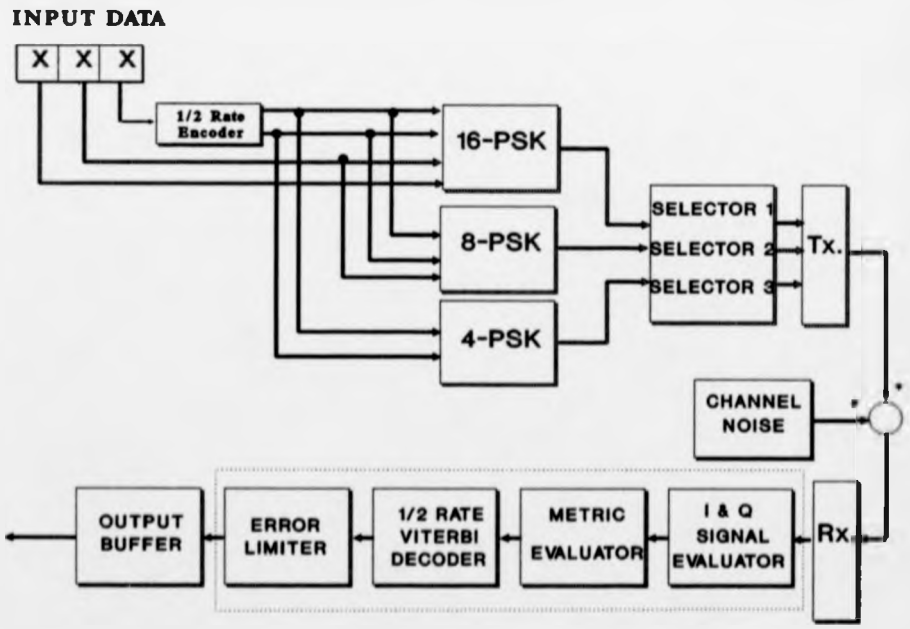


Figure 5.1: Block diagram of embedded TCM system

tion system for 4, 8 and 16 level PSK. The system comprises the following

elements:

- data source, generating binary data;
- 1/2 rate convolutional encoder;
- three different mapping schemes;
- three modulator selectors and corresponding buffers;
- transmitter, receiver and the channel;
- I & Q evaluator;
- metric evaluator;
- 1/2 rate Viterbi decoder;
- error limiting system;
- data sink buffer.

The first bit in the input information stream enters a half rate binary convolutional encoder; the output of the encoder together with two (uncoded) bits from the information source, select one of the sixteen possible phases in the 16-PSK phase plane. The second bit in the input information stream and the encoder output, select one of the possible phases in the 8-PSK phase plane. Finally, the two output bits of the encoder are mapped to the 4-PSK plane.

Now, consider the case where each selector's buffer consists of 60 bits; the number of symbols in the buffers of selectors 1, 2 and 3 are 15, 20 and 30 symbols respectively. The number of information bits in each buffer is 45, 40 and 30. Therefore, the streams are coded with rates 3/4, 2/3 and 1/2 for selectors 1, 2 and 3 respectively. These blocks are then transmitted serially and in a specific format i.e. 16-PSK first, 8-PSK second and finally 4-PSK. Note that unlike the embedded convolutional coding technique [Zolghadr *et al.* (1988)], the resulting streams have the same block length, thus removing the need for code puncturing and subsequently insertion of erasures at the receiver.

At the receiver, the demodulation and decoding processes are combined together. Since one type of encoder is used, the decoding process can be carried out using a single Viterbi decoder. In order to perform the decoding procedure, the inphase and the quadrature (I & Q) components of the received signal are evaluated. This is used to calculate the squared Euclidean distances. The calculated likelihood metrics are then passed on to the maximum likelihood soft decision decoding algorithm.

The real-time channel evaluation is done via further use of the soft decision information within the decoder. ARQ is initiated from this additional RTCE information obtained from the Viterbi decoding procedure. The Viterbi decoder employs the Viterbi algorithm for implementing maximum likelihood decoding (MLD), specifically to perform error correction on the received bit stream by evaluating the accumulated path confidences. The error limiter measures probability of error of the decoder after the initial

correction has been performed by the Viterbi decoder. Due to the nature of the decoding procedure for the convolutional codes, in an ARQ situation the decoder buffer must be "flushed" [Lin *et al.* (1984)]. In order to accommodate this requirement in a repeat request condition, the transmitter will re-transmit the rejected streams. Subsequently, this action overwrites the old contents of the decoder buffer. The decoding then commences as soon as the decoder buffer is full.

The error control technique described above can be viewed as an adaptive hybrid ARQ scheme [Lin *et al.* (1984)], in which the information is encoded/modulated using code and modulation schemes of increasing error correcting capability. Thus, at times when the bit error rate (BER) is relatively high, only the stream encoded with the most powerful (in terms of low BER) modulation scheme may be accepted, at the expense of low overall transmission rate and a subsequent reduction in system throughput efficiency. However, when the channel SNR is comparatively high, the stream using the highest modulation level (less powerful) is accepted, resulting in an increase in the system throughput efficiency.

In an ARQ system, however, the decoder buffer imposes a need for data buffering and more complex protocols. This problem is somewhat amplified when the embedded trellis coded modulation (ETCM) format is employed in the ARQ system. The outputs of the modulators are at three different rates. At the receiver, the MLD uses three buffers to store the most recent L_i, trellises, where i is the stream number and L_i is the decoder search length.

The search lengths, L_i, are arranged to be equal to the block size of the coded/modulated streams. Therefore, decoding can only commence when one complete block has been stored in the MLD buffers. This arrangement simplifies the ARQ protocol since, at the end of decoding any given block, the most recently transmitted block will be in the buffer; at this point, the error limiter can decide whether the coded/modulated stream is acceptable. The error limiter can reach four possible decisions; these are:

(a) accept the whole block (of stream 1,2 and 3);

(b) accept stream 1;

(c) accept stream 2;

(d) accept none of the streams (even if stream 3).

In the first case, where all the streams are accepted, the decoding continues as normal and the transmitter sends the next coded/modulated block in the queue. In the second case, when a segment of the block is accepted, the decoder prepares for its buffers to be flushed. This involves resetting all the decoder variables and pointers. In this situation the first received block is used to overwrite the contents of the decoder buffers and the decoding commences as soon as the second block is received. A similar procedure is applied in the third case. In the case where none of the streams are accepted, the whole block is retransmitted.

In an ARQ situation, the transmitter behaves somewhat differently. If the transmitted block is accepted, the normal mode of operation is contin-

ued. The repeat request from the receiver causes the rejected streams to be modulated with a more powerful modulation scheme. Depending on which of the streams has been accepted, the information digits in the rejected streams are collated with the addition of some new data (if required), to form a new data block. At the receiver, the first data block is used to fill the decoder buffers prior to transmission of the new data block.

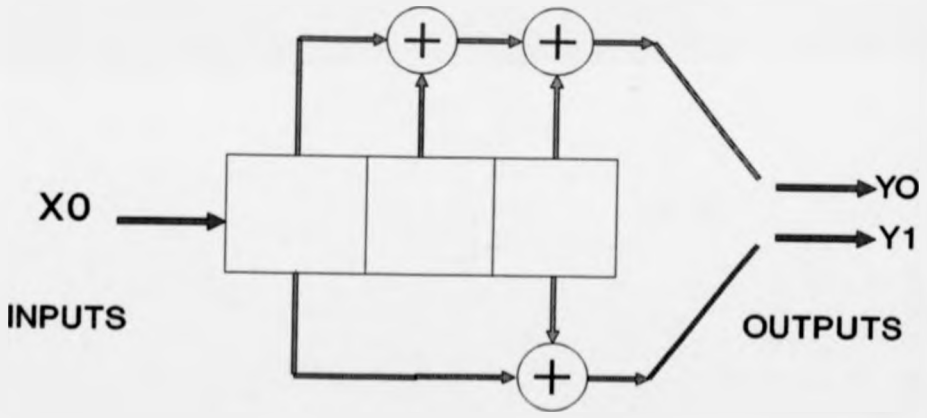


Figure 5.2: Block diagram of half rate convolutional encoder

The convolutional encoder used consists of a 3-bit shift register, two sets of taps and modulo-2 adders. The generator polynomials of the code, G_1 and

G_2, are

$$G_1 = X^2 + X + 1 \tag{5.1}$$

$$G_2 = X^2 + 1 \tag{5.2}$$

The resulting convolutional encoder is shown in Fig.5.2. The trellis representation of encoder/modulator configuration is shown for 4,8 and 16 levels in Fig.5.3. The trellis diagram corresponding to the 1/2 rate code (stream 3) is shown in Fig.5.3(a). This trellis has 4 states, each of which has two inputs and two outputs (i.e branches). The trellis for 8-PSK (rate 2/3, stream 2) (Fig.5.3(b)) has twice as many branches, with essentially the same connectivity, but with all branches paired, since the input bit, which does not enter the rate 1/2 encoder produces two branches with identical connectivity. All pairs of states connected in Fig.5.3(a) by a single branch are now connected by four branches for 16-PSK (rate 3/4, stream 1) trellis diagram, each corresponding to one of the four values of the two input bits, which do not enter the encoder.

The analysis of the channel is based on the following model: the message source generates a stream of equally likely messages which are then encoded and subsequently used to select one of m possible distinct waveforms (each with duration T) for transmission across the channel. The choice of waveform depends on the encoder's tap configuration and on the waveforms transmitted in the preceding time slots. The medium coupling the transmitter to the

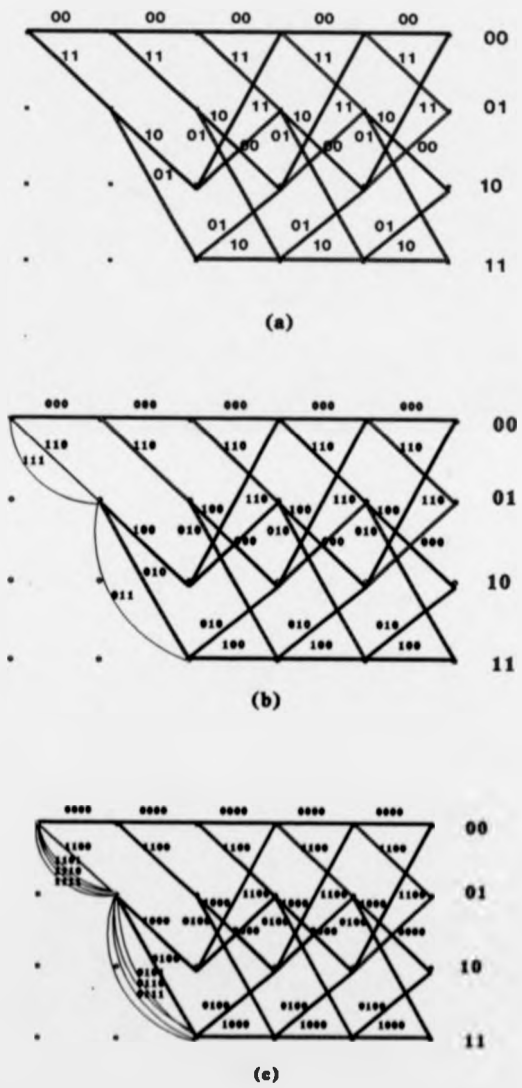


Figure 5.3: Trellis representation of (a) 4-PSK, (b) 8-PSK, (c) 16-PSK

receiver is assumed to add stationary, white, zero-mean, Gaussian noise to the transmitted signal. The received signal $r(t)$ is given by

$$r(t) = s_i(t) + n(t), \qquad 1 \leq t \leq m \tag{5.3}$$

where $n(t)$ is a noise sample function, having a constant power spectral density function equal to $N_o/2$; N_o is noise energy per symbol and s_i is the transmitted signal. By selecting a value for the standard deviation of the noise, the SNR of the system is set.

The metric evaluator is used in conjunction with the soft-decision decoder to perform error correction. The input to this unit is the Euclidean distances which are generated by the I and Q evaluator. These values are sent to the metric evaluator in order to obtain the corresponding metric values for a particular received symbol and modulation level. Depending on the modulation scheme, the metric evaluator determine the Euclidean distance between the received symbol and all possible constellation points available with that particular modulation scheme. Once the Euclidean distance values corresponding to one channel symbol have entered the metric evaluator, the following steps are performed:

(i) decide on the possible transition from the initial node. In the case of 8 and 16- PSK, this includes all the multi-paths;

(ii) allocate metrics for all possible transition branches leading off that node;

(iii) repeat the above steps for all possible nodes in the trellis;

(iv) once completed, output the trellis values to the decoder.

The path metrics are then simply the result of the addition of the metrics of each branch.

As a by-product of the Viterbi algorithm, the accumulated confidence metric of the greatest likelihood path (correct path) is readily available. The RTCE system retrieves this information and causes the decoder to make an additional pass through the trellis in search of the next greatest likelihood path (second path). In order to find the second path, the same Viterbi decoder is used, however, the decoder is not allowed to search all possible branches. It is clear that, for the second path to be the nearest error path to the correct path, it must originate from the same state as the correct path; and should differ from the correct path by at least one branch. This branch must include the branch corresponding to the transition from the initial state to the next state. The decoder is not therefore allowed to pursue paths emerging from the first branch of the correct path. It is evident that when the channel conditions cause an incorrect path to be chosen, the error detection metrics are reduced in magnitude. Error detection metrics are a good indication of the level of noise in the channel.

The probability of error of the Viterbi decoder is dependent on the channel SNR; therefore, once the channel SNR is determined, an estimate of the output probability of error from the decoder for each of the three modulation schemes can be obtained.

5.3 Performance Evaluation

In the embedded trellis coded modulation scheme, the three streams were implemented using a single rate 1/2, constraint length 3 convolutional encoder/decoder with appropriate modifications of the metric input sections. Combination of uncoded and convolutional encoded bits were mapped to three different constellation points in order to generate the multi-rate code. Therefore, the convolutional encoder portion of the trellis is now fixed for all three data rates.

Since in the optimum design of a conventional trellis code, the convolutional encoder portion is not fixed, the above configuration is sub-optimum [Viterbi *et al.* (1979)] . By limiting the code to incorporate a rate 1/2 convolutional encoder, we force the existence of two and four parallel paths per branch for 8-PSK and 16-PSK modulation schemes respectively. Therefore, the free Euclidean distance which is a measure of code performance is now limited by the minimum distance among the parallel paths, which has a smaller value. Table 5.1 presents the performance of trellis-coded M-PSK for 4-, 8- , and 16-PSK modulations with single and parallel branches for 4 and 8

states trellis. It can be seen that, for example in the case of 8-PSK, in order to reduce the number of parallel transitions between states from 2 to 1, the number of states is increased from 4 to 8. In this case, 0.6 dB coding gain can be achieved in expense of increased in complexity of decoding. Therefore, having parallel paths for each branch does not contribute very much towards loss in coding gain in the embedded trellis coding scheme.

Number of Modulation Levels M	Number of Parallel Transitions Between States	Number of States in Trellis	d^2_{min} for uncoded Modulation with M/2 Level	d^2_{free} for *coded* modulation with M level	Gain (dB) of Coded MPSK over uncoded M/2- PSK
4	1	4	4	10	3.98
8	1	8	2	4.586	3.60
8	2	4	2	4	3.01
16	4	4	0.586	1.324	3.54
16	4	8	0.586	1.476	4.014

Table 5.1: Performance of rate $n/(n+1)$ trellis-coded M-PSK vs. uncoded M/2-PSK

Fig.5.4 shows the probability density function (pdf) of the error detection metrics at the output of the 1/2 rate Viterbi decoder for a channel SNR of 5, 9 and 15 dB, with the decoder search length, L, set to 30. It is clear from the diagram that the variance of the pdf is distinct for a given channel SNR value. Therefore, the error detection metrics are a good indication of the level of noise in the channel.

The error limiter therefore averages the amplitude of the error detection metrics and compare them with a set of previously obtained thresholds. The thresholds are arranged such that the overall output probability of error of the system is kept below a specified maximum value.

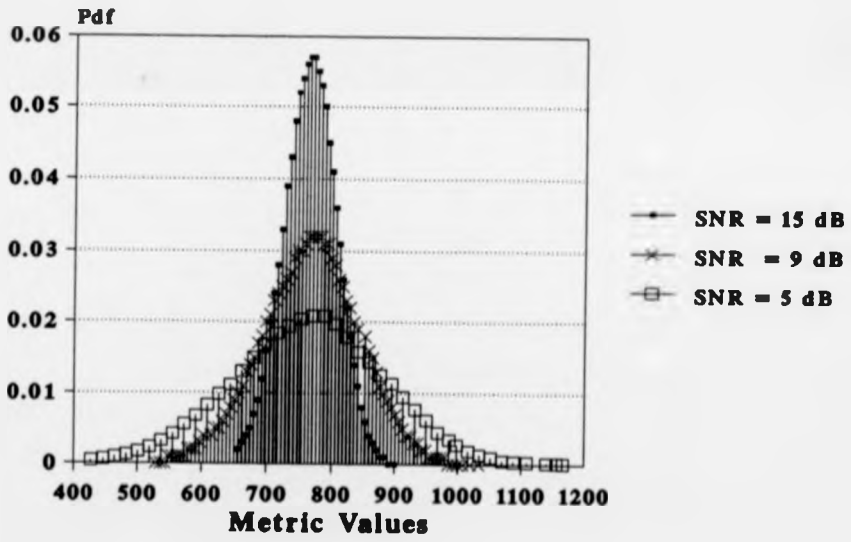


Figure 5.4: Pdf of metrics for channel SNR of 5, 9 and 15 dB

Fig. 5.5 shows the effect of the channel SNR on the probability of error (BER) for the three modulation schemes. The performance of uncoded 4-PSK is shown only for comparison. It can be seen that, by crowding more signal vectors into the signal space (16-PSK), increase in bandwidth utilisation at the expense of error performance can be achieved [Clark *et al.* (1988)].

The embedded trellis coded modulation performs well in environments where the channel SNR varies continuously. Fig.5.5 provides some insight into a basic trade off in ETCM. To obtain an overall system bit error rate of 10^{-5}, when the E_b/N_0 is less than say 4.5 dB, none of the streams are accepted, thus improving the reliability of the system. At higher channel SNR ($4.5dB \leq E_b/N_0 < 6.5dB$) stream 3 is accepted. It is clear that in this range of E_b/N_0, the system's overall probability of error is bounded by the performance of the 1/2 rate code and 4-PSK modulation. Both streams 3 & 2 are accepted at higher values of SNR (i.e. $6.5dB \leq E_b/N_0 < 10dB$). Stream 1 however is only accepted at $E_b/N_0 > 6dB$, where the rate of the overall system is at its highest, i.e. 16-PSK modulation is used. Here, it can be seen that the 16-PSK stream has a higher rate of transmission but the vulnerability of its symbols to errors due to noise in the channel is higher than for 8-PSK, which has a lower transmission rate. Therefore, in contrast to the embedded convolutional code [Zolghadr *et al.* (1988)], as the rate of the code is adjusted, the level of symbol protection alters according to the level of M-ary modulation scheme.

It is clear that in comparison to conventional fixed coded/modulation

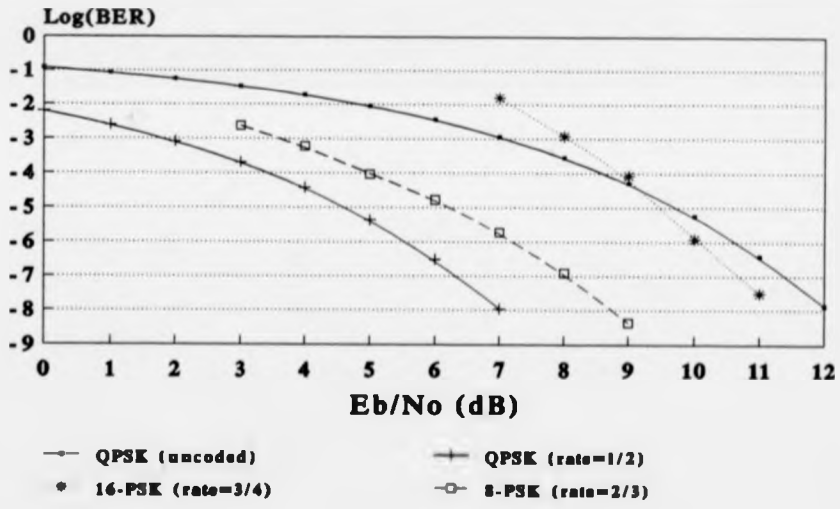


Figure 5.5: Performance comparison of embedded trellis coded modulation

schemes, an improvement in terms of overall system throughput at a fixed BER can be obtained. For example, at BER of 10^{-5}, and $E_b/N0 > 10$ dB, where all streams are accepted, the throughput is higher than the case where only a 1/2 rate convolutional code and 4-PSK modulation scheme is used. Although, for the condition where $E_b/N_0 < 10$ dB the ETCM proves less efficient, this must be viewed in the context of a time variable channel and it is assumed that at least one of the rates will normally be received successfully over a wide range of channel conditions, hence improvement in terms of reliability can be achieved over conventional TCM systems, when used in an ARQ environment.

Chapter 6

Low Complexity Trellis Coded Modulation

6.1 Introduction

The performance of a digital data transmission system can often be improved by using a suitable convolutional coder to form the transmitted symbols, together with an appropriate decoding process at the receiver [Viterbi, (1967), Viterbi, (1971)]. The optimum decoding of the code is achieved by a maximum likelihood process which selects, as the decoded message, the possible sequence of data symbols fed to the coder, that has the maximum likelihood function [Gallager, (1968)]. The decoder holds in store a fixed number of the more likely sequences of data symbols, and their associted metric, and operates on the received signals, after the latter has been sampled once per code symbol. In the presence of stationary additive white Gaussian noise, giving statistically independent Gaussian noise samples at the input of the decoder [Clark, (1977)], and when the different possible received messages are equally likely, as is assumed here, the maximum likelihood sequence is the one which is most likely to be correct. This sequence is the possible sequence of data symbols which is fed to the coder for which there is the minimum unitary [Clark, (1977)] distance between the sequence of the corresponding received samples in the absence of noise and the samples actually received by the decoder [Viterbi, (1971)]. Therefore, under the assumed conditions, maximising the likelihood function is equivalent to minimising the unitary distance.

A practical optimum decoder employs the Viterbi algorithm [Viterbi, (1967),

Viterbi, (1971), Clark, (1977), Omura, (1969), Heller *et al*, (1971), Forney, (1973)]. which utilises the structural and distance properties of convolutional codes systematically, to achieve maximum likelihood (minimum distance) decoding. However, the Viterbi algorithm has generally been considered impractical for codes other than those having relatively short constraint lengths (for example, less than about 10 for a convolutional code operating on a binary signal). This is because of the exponential growth in equipment complexity of the Viterbi decoder with the code constraint length. Theoretically, many convolutional codes with large constraint lengths give a better tolerance to additive white Gaussian noise at very low error rates than those with short constraint lengths, as long as optimum decoding is used. However, optimum decoding is now difficult, if not impossible, to achieve in practice. A number of decoders (detectors) that come close to achieving the maximum likelihood (minimum distance) detection of the received signals without, however, requiring nearly as much as computation and storage as does the Viterbi decoder (detector), have been developed in recent years [Russell *et al*, (1976), Foschini, (1977), Clark *et al*, (1978)]. They are known as near maximum likelihood decoders. These decoders operate with fewer stored metrics than the corresponding Viterbi decoder. The method of selecting the stored vectors differs from one system to another, but it generally uses, as a measure of the goodness of any stored metric value, the unitary distance squared between the sequence of samples received in the presence and in the absence of noise. The greater the unitary distance squared or

metric value, the less likely is the sequence to be correct. The sequences with the highest metric values are simply discarded.

In certain decoding algorithms, the soft decision information is used in a straightforward manner such as for tree codes.

In a Viterbi decoder, each time a new branch is received the branch metric computer determines a new set of metric values for each of the different branches which appear in the code trellis. For rate 1/2 codes there will be four different values or in general,2^n values for codes having n symbols per branch. In the most straight forward case the branch metric is simply the sum of the individual metrics associated with each received symbol. To find the metric values requires calculation of the Euclidean distance between the received symbol and hypothesised symbol. When the branch metric values are used to compute path metrics, it is clear that the number of metric values that is required to represent each path can increase or decrease the complexity of the decoder. Therefore, the complexity of the decoder is a rather strong function of the number of metrics that are required to represent each of the paths. Thus, in many situations it is desirable to keep this number as small as possible.

In our new decoding method [Arani *et al.* (1991)], we have overcome some of the existing limitations in implementation of the Viterbi algorithm. Firstly, the soft decision information is used directly in the decoding procedure and this means that there is no need to calculate the Euclidean distances. Secondly, there is no need to store the metric values at each node, although

storage of certain values of inphase and quadrature components of the received signal are required, which will reduce the overall amount of required storage. Finally, the add-compare-select (ACS) circuitry [Clark *et al*, (1988)] in the decoder is simpler, because the comparison is done by zero threshold level.

In this chapter the new decoding algorithm for convolutional codes is introduced. It has been shown that the decoder complexity is reduced substantially in comparison with a conventional Viterbi decoder. The particular application studied is that of trellis coded 4-PSK modulation.

6.2 Metric Evaluation

In order to evaluate the computational requirement for Viterbi decoding algorithm, consider the trellis structure shown in Fig.6.1 for the binary (n, k, L) convolutional encoder, with k input, n output and memory (constraint length) L bits. In Fig.6.1, the branch leading from S_i to state S_j of the next stage is denoted as b_{ij}. The Viterbi decoding algorithm searches for the shortest path in the trellis diagram by survivor metric evaluation. There are $2^{k(L-1)}$ candidate paths leading to a certain state S_j at time t. Each path consists of the survivor path leading to S_i at the $(t-1)^{th}$ stage from S_i to S_j at the t^{th} stage by the branch b_{ij}.

The computations required for each state are to find the survivor metrics and determine the survivor path among all the candidate paths to S_j. Let

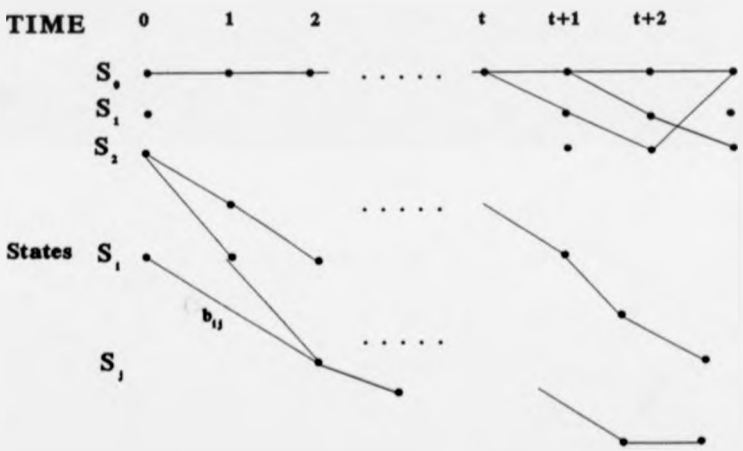


Figure 6.1: Generalised trellis structure

$J(j)$ denotes the set of indexes i of the 2^k states of previous stage S_i leading to S_j, each with a branch b_{ij} on the trellis. In Fig.6.1, for state S_2 at $t+2$, $J(2) = \{1, 2\}$. Path metrics for all the candidate paths of S_j, denoted by $pm(i, j, t)$ for all $j \in J(j)$, will be evaluated via likelihood measurement function f, and the maximum metric among them will be chosen as the survivor metric $sm(j, t)$. Let $r(t)$ be the n bit word received at the t^{th} stage and $w(i, j)$ the valid segment of transition e_{ij}. The $pm(i, j, t)$ and $sm(j, t)$ are related as

$$pm(i, j, t) = f\{r(t), w(i, j), sm(i, t-1)\} \tag{6.1}$$

$$sm(j, t) = max(pm(i, j, t)) \tag{6.2}$$

where max is the maximum comparator and is applied over all S_i with $i \in J(j)$.

Let us consider the trellis diagram (Fig.6.2(a)) for convolutional code (2,1,3). This trellis can be modified by combining p stages with p branches selected as a single branch. In Fig.6.2(b),the first two sub-trellis have been merged. Therefore, each branch is equivalent to exactly a two branch path in the original trellis diagram. The mapping of several sub-trellises into a single trellis can be explained by an example. Assume that $pm(0, 1, t+1)$ in Fig.6.2(b) is chosen as the survivor metric for S_1 at the $(t+1)$ stage and $pm(1, 3, t+2)$ as the survivor metric for S_3 at $(t+2)^{th}$ stage, then the 2-branch path $P_{1,3,t+2}$ composed of $b_{0,1}$ and $b_{1,3}$ is one of the candidate paths of

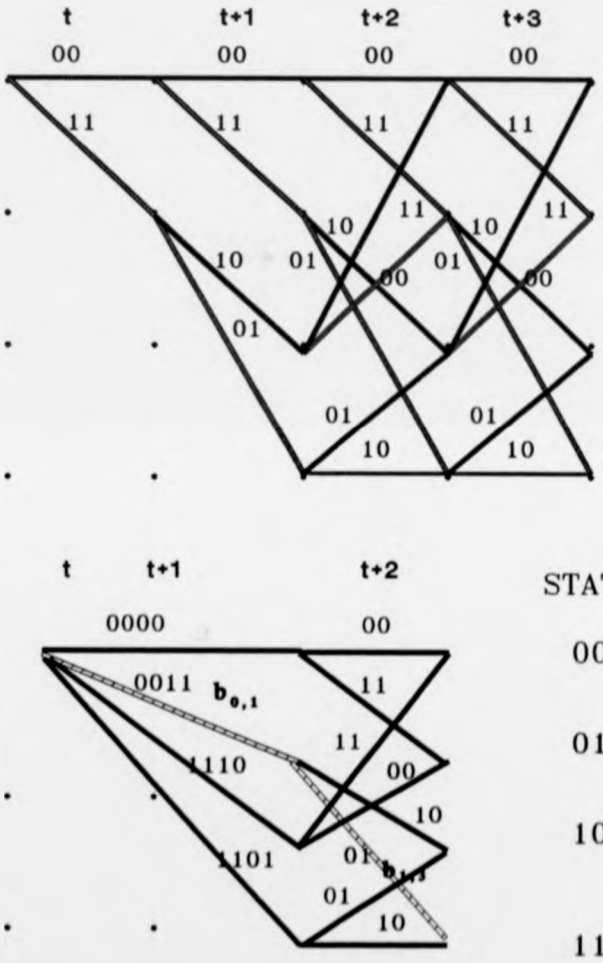


Figure 6.2: Trellis structure for (2,1,3) convolutional code

S_3 at the $(t+2)^{th}$ stage. In the conventional Viterbi decoding, $pm(1,3,t+2)$ is calculated as follows:

$$\begin{aligned} pm(1,3,t+2) &= f(r(t+2), w(1,3), sm(1,t+1)) \\ &= f(r(t+2), w(1,3), max(pm(i,1,t+1))) \\ &= f(r(t+2), w(1,3), pm(0,1,t+1)) \\ &= f(r(t+2), w(1,3), f(r(t+1), w(0,1), sm(0,t))) \\ &= f(r(t+2), w(1,3), f(r(t+1), w(0,1), max(pm(i,0,t))), \cdots) \end{aligned} \tag{6.3}$$

Using this technique requires update availability of survivor metrics for all S_i with $i \in J(j)$. This requires excessive data access operation and storage. In the case where p stages are merged into a single one, m successive branches will be taken into an integrated branch. Therefore, the survivor metric for each state will be updated every p stages and the number of data access operations can be reduced p times. In order for survivor metric evaluation to be compatible with the original Viterbi algorithm, the mapping of the trellis diagram to a combined one must be done in one to one correspondence to a p branch path in the original trellis. The reason for this requirement arises because there is exactly one path between any pair of nodes which are L stages apart. Over p stages where $p < L$, there are either one or no paths between any arbitrary pair of nodes. Hence, each p branch may be uniquely specified by its starting and finishing state.

6.2.1 Transmission Channel and the Receiver Model

The possible transmitted signal, in the case of 4-PSK can be mathematically expressed as

$$\begin{aligned} S_i(t) &= \sqrt{(2E_s/T)}\sin[2\pi ft + \theta_i(t)] \\ &= \phi_1(t)\sqrt{E_s}\cos\theta_i(t) + \phi_2(t)\sqrt{E_s}\sin\theta_i(t) \end{aligned} \tag{6.4}$$

for $0 \leq t \leq T$, $i = 1, 2, 3, 4$ where

$$\begin{aligned} \phi_1(t) &= \sqrt{(\frac{2}{T})}\sin(2\pi ft) \\ \phi_2(t) &= \sqrt{(\frac{2}{T})}\cos(2\pi ft) \end{aligned} \tag{6.5}$$

for $0 \leq t \leq T$,

where

$$T = \textit{symbol duration}$$

$$f = \textit{carrier frequency}$$

$$E_s = \int_0^T S_i^2(t)\,dt = \textit{energy per symbol} \tag{6.6}$$

for $i = 1, 2, 3, 4$

$$\theta_i(t) = \frac{\pi(2i-1)}{4} \qquad 0 \leq t \leq T \tag{6.7}$$

Depending on which symbol is transmitted, because of the AWGN, the received signal $r(t)$ is given by

$$r(t) = S_i(t) + n(t) \qquad 0 \leq t \leq T \tag{6.8}$$

where $n(t)$ is a sample function of the AWGN having a constant power spectral density function equal to $N_0/2$.

If r_1 is the component of $R(t)$ along $\phi_1(t)$ and r_2 is the component of $R(t)$ along $\phi_2(t)$. For symbol S_i;

$$\begin{aligned} r_1 &= \int_0^T [\phi_1(t)\sqrt{E_s}\cos\theta_i(t) + \phi_2(t)\sqrt{E_s}\sin\theta_i(t) + n(t)]\phi_1(t)\,dt \\ &= \sqrt{E_s}\cos(\theta_i) + n_1 \end{aligned} \tag{6.9}$$

where

$$n_1 = \int_0^T n(t)\phi_1(t)\,dt \tag{6.10}$$

and

$$\begin{aligned} r_2 &= \int_0^T [\phi_1(t)\sqrt{E_s}\cos\theta_i(t) + \phi_2(t)\sqrt{E_s}\sin\theta_i(t) + n(t)]\phi_2(t)\,dt \\ &= \sqrt{E_s}\sin(\theta_i) + n_2 \end{aligned} \tag{6.11}$$

where

$$n_2 = \int_0^T n(t)\phi_2(t)\,dt \tag{6.12}$$

n_1 and n_2 are zero mean Gaussian random variables having identical variance σ_n^2. Also n_1 and n_2 are uncorrelated random variables. Because n_1 and n_2 are Gaussian, they are also statistically independent variables.

The square Euclidean distance between a pair of points R_i and S_i with coordinates of (r_1, r_2) and (s_1, s_2) respectively is given by the formula [Lin *et al*, (1983)]:

$$d^2(S_i, R_i) = (r_1 - s_1)^2 + (r_2 - s_2)^2 \tag{6.13}$$

6.2.2 Method of Decision Making

Now, let us examine a digital communication system which uses the convolutional encoder of Fig.6.3(a) and the 4-PSK modulation scheme (Fig.6.3(b). The Viterbi decoder can only remember a part of the convolutional codeword at a time. Although the codeword is effectively infinite in length, all decoding decisions must be made on received word segments of finite length. But, because of the structure of the code, no matter how one chops out a part of the received word for the decoder to work with, there is some interaction with other parts of the received word that the decoder does not see. Consider the trellis showing all the possible transmitted sequences up to time t+3 in Fig.6.2(a). The Viterbi decoder begins by examining the first 3 branches and making $2^K = 8$ pairwise comparisons among those paths with minimum Euclidean distance from the K received symbols.

If we call the Euclidean distances between the received symbol given the

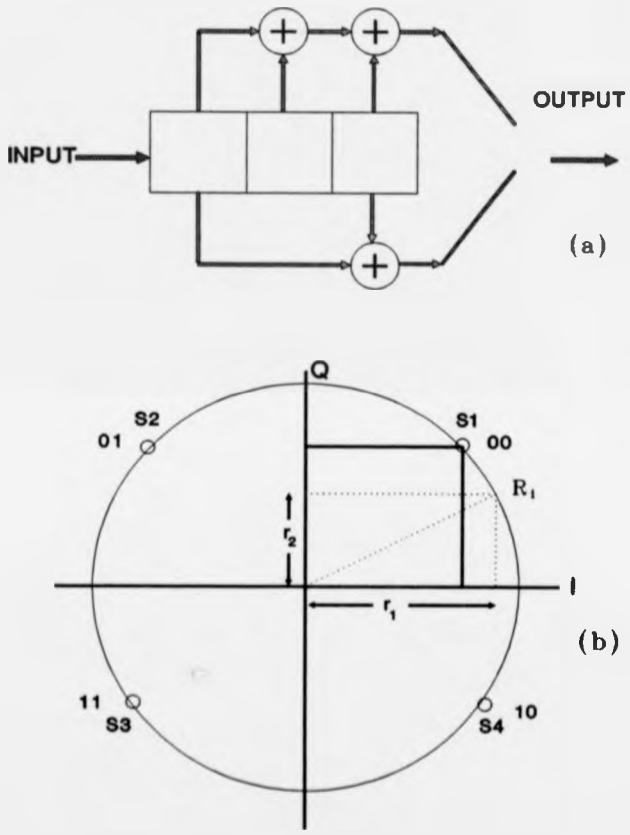


Figure 6.3: (a) Half rate convolutional code, (b) 4-PSK constellation points

symbol 00, 11, 01 and 10 were transmitted d_{00}, d_{11}, d_{01} and d_{10} respectively, then (Fig.6.3 b):

$$d_{00}^2 = (r_1 - \frac{\sqrt{2}}{2})^2 + (r_2 - \frac{\sqrt{2}}{2})^2 \tag{6.14}$$

$$d_{11}^2 = (r_1 + \frac{\sqrt{2}}{2})^2 + (r_2 + \frac{\sqrt{2}}{2})^2 \tag{6.15}$$

$$d_{01}^2 = (r_1 - \frac{\sqrt{2}}{2})^2 + (r_2 + \frac{\sqrt{2}}{2})^2 \tag{6.16}$$

$$d_{10}^2 = (r_1 + \frac{\sqrt{2}}{2})^2 + (r_2 - \frac{\sqrt{2}}{2})^2 \tag{6.17}$$

From the trellis diagram (Fig.6.2(a)), at $t+2$ stage it can be seen that in order to decode the $K-1$ receiving symbols, the following Euclidean distances have to be compared.

$$R_{00}^{1_{t+2}} = d_{00_t}^2 + d_{00_{t+1}}^2 + d_{00_{t+2}}^2 \qquad \textit{at state } 00 \tag{6.18}$$

$$R_{00}^{2_{t+2}} = d_{11_t}^2 + d_{10_{t+1}}^2 + d_{11_{t+2}}^2 \qquad \textit{at state } 00 \tag{6.19}$$

$$R_{01}^{1_{t+2}} = d_{00_t}^2 + d_{00_{t+1}}^2 + d_{11_{t+2}}^2 \qquad \textit{at state } 01 \tag{6.20}$$

$$R_{01}^{2_{t+2}} = d_{11_t}^2 + d_{10_{t+1}}^2 + d_{00_{t+2}}^2 \qquad \textit{at state } 01 \tag{6.21}$$

$$R_{10}^{1_{t+2}} = d_{00_t}^2 + d_{11_{t+1}}^2 + d_{10_{t+2}}^2 \qquad \textit{at state } 10 \tag{6.22}$$

$$R_{10}^{2_{t+2}} = d_{11_t}^2 + d_{01_{t+1}}^2 + d_{01_{t+2}}^2 \qquad \textit{at state } 10 \tag{6.23}$$

$$R_{11}^{1_{t+2}} = d_{00_t}^2 + d_{11_{t+1}}^2 + d_{01_{t+2}}^2 \qquad \textit{at state } 11 \tag{6.24}$$

$$R_{11}^{2_{t+2}} = d_{11_t}^2 + d_{01_{t+1}}^2 + d_{10_{t+2}}^2 \qquad \textit{at state } 11 \tag{6.25}$$

By comparing eqns 6.18 & 6.20 and also 6.19 & 6.21 the following relationship

can be obtained

$$R_{00,01}^{t+2} = d_{11_{t+2}}^2 - d_{00_{t+2}}^2 \tag{6.26}$$

Similarly, comparison between 6.22 & 6.24 and 6.23 & 6.25 will result

$$R_{10,11}^{t+2} = d_{10_{t+2}}^2 - d_{01_{t+2}}^2 \tag{6.27}$$

By observing the trellis diagram, it can be seen that there is a need for another comparison, to satisfy final decision making at any other node in the trellis diagram. This comparison is between 6.26 and 6.27. By doing this comparison all possible transitions in the trellis diagram are considered.

Therefore;

$$R_{00,11}^{t+2} - R_{10,11}^{t+2} = (d_{11_{t+2}}^2 - d_{00_{t+2}}^2) - (d_{10_{t+2}}^2 - d_{01_{t+2}}^2) \tag{6.28}$$

Using eqn.6.28, the trellis diagram and constellation diagram in Fig.6.2, if the $I\&Q$ components of the received symbols at time $t+2$ have the same sign i.e. both positive or both negative, we are at state 00 or 01 and if they have opposite signs, we are at state 10 or 11.

It can be observed that, if we are at state 00 or 01 at $t+2$, in order to decode the received symbols at t and $t+1$, the following comparison has to be made.

$$R_{00,01}^{t} = (d_{11_t}^2 + d_{10_{t+1}}^2) - (d_{00_t}^2 + d_{00_{t+1}}^2) \tag{6.29}$$

Therefore, if the result of 6.29 is positive, it can be concluded that $d_{00} + d_{00}$ is minimum, hence 0000 were the transmitted codeword segments. Similarly, if the result of the above comparison is negative, it can be concluded that symbol 11 and 10 were the transmitted codewords at time t and $t + 1$ respectively. The above relationships can be written in terms of inphase and quadrature components by using equations 6.14-6.17 in 6.29. The result is

$$\mathcal{R}'_{00,01} = r_{1_t} + r_{2_t} + r_{1_{t+1}} \tag{6.30}$$

Similarly, if we are at state 10 or 11 at t+2, the following comparison has to be made.

$$R'_{10,11} = (d^2_{11_t} + d^2_{01_{t+1}}) - (d^2_{00_t} + d^2_{11_{t+1}}) \tag{6.31}$$

This also can be simplified in terms of I & Q components.

$$\mathcal{R}'_{10,11} = r_{1_t} + r_{2_t} - r_{1_{t+1}} \tag{6.32}$$

Therefore, if the above relationship (6.32) is positive, it can be concluded that 1101 were the transmitted codeword segments and if it is negative, 0011 was transmitted.

The summary of the above decoding procedure is shown in Fig.6.4. After decoding the codewords at time t and $t + 1$, the other paths are discarded, therefore, at $t + 2$ we start at one node. After receiving two more consecutive symbols i.e. at $t + 4$, the decoding procedure for symbols at $t + 2$ and $k + 3$

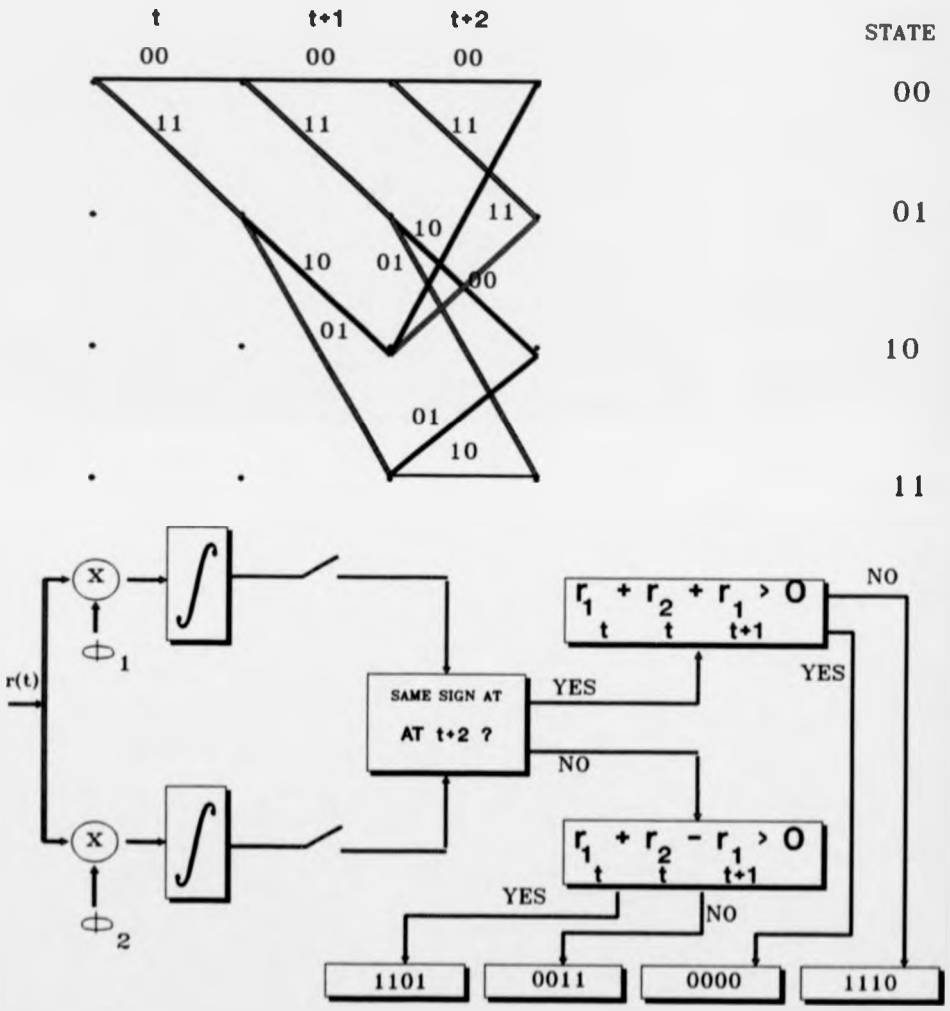


Figure 6.4: Decoding block diagram (t & t+1)

will commence. In order to do this, there are eight comparison which have to be made, depending on the starting state at $t+2$. Similarly, it can be shown that these can be simplified to only two comparisons. The following metric values have to be compared assuming we are at state 00 at $t+2$.

$$R_{00}^{1_{t+4}} = d_{00_{t+2}}^2 + d_{00_{t+2}}^2 + d_{00_{t+4}}^2 \tag{6.33}$$

$$R_{00}^{2_{t+4}} = d_{11_{t+2}}^2 + d_{10_{t+3}}^2 + d_{11_{t+4}}^2 \tag{6.34}$$

$$R_{01}^{1_{t+4}} = d_{00_{t+2}}^2 + d_{00_{t+3}}^2 + d_{11_{t+4}}^2 \tag{6.35}$$

$$R_{01}^{2_{t+4}} = d_{11_{t+2}}^2 + d_{10_{t+3}}^2 + d_{00_{t+4}}^2 \tag{6.36}$$

$$R_{10}^{1_{t+4}} = d_{00_{t+2}}^2 + d_{11_{t+3}}^2 + d_{10_{t+4}}^2 \tag{6.37}$$

$$R_{10}^{2_{t+4}} = d_{11_{t+2}}^2 + d_{01_{t+3}}^2 + d_{01_{t+4}}^2 \tag{6.38}$$

$$R_{11}^{1_{t+4}} = d_{00_{t+2}}^2 + d_{11_{t+3}}^2 + d_{01_{t+4}}^2 \tag{6.39}$$

$$R_{11}^{2_{t+4}} = d_{11_{t+2}}^2 + d_{01_{t+3}}^2 + d_{10_{t+4}}^2 \tag{6.40}$$

Similar to the decoding procedure at $t+2$, if the I & Q components of the received symbols at time $t+4$ have the same sign i.e. both positive or both negative, we are at state 00 or 01 and if they have opposite signs, we are at state 10 or 11. Also, it can be shown that if

$$r_{1_{t+2}} + r_{2_{t+2}} - r_{1_{t+3}} \geq 0 \tag{6.41}$$

then the decoded codeword is

00 00

and if

$$r_{1_{t+2}} + r_{2_{t+2}} + r_{1_{t+3}} \leq 0 \tag{6.42}$$

then the decoded codeword is

11 10

The above decoding procedure for the four possible starting position at $t+2$ in the trellis (i.e. state 00, 01, 10 or 11) are presented in Figs. 6.5, 6.6, 6.7 and 6.8 respectively.

6.2.3 Numerical Example

Let assume the following case where there are two single errors at $t = 1$ and $t = 3$.

time	:	t	$t+1$	$t+2$	$t+3$	$t+4$
Tx	:	00	00	11	10	11
Rx	:	00	01	11	11	11
r_1	:	$\frac{\sqrt{2}}{2}$	$\frac{-\sqrt{2}}{2}$	$\frac{-\sqrt{2}}{2}$	$\frac{-\sqrt{2}}{2}$	$\frac{-\sqrt{2}}{2}$
r_2	:	$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{2}}{2}$	$\frac{-\sqrt{2}}{2}$	$\frac{-\sqrt{2}}{2}$	$\frac{-\sqrt{2}}{2}$

Refer to Fig.6.4.

Do r_1 and r_2 at $t+2$ have the same sign? Answer: YES. Therefore

$$\text{is } r_{1_t} + r_{2_t} + r_{1_{t+1}} \geq 0$$

$$\text{i.e. } \frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2} = \frac{\sqrt{2}}{2}$$

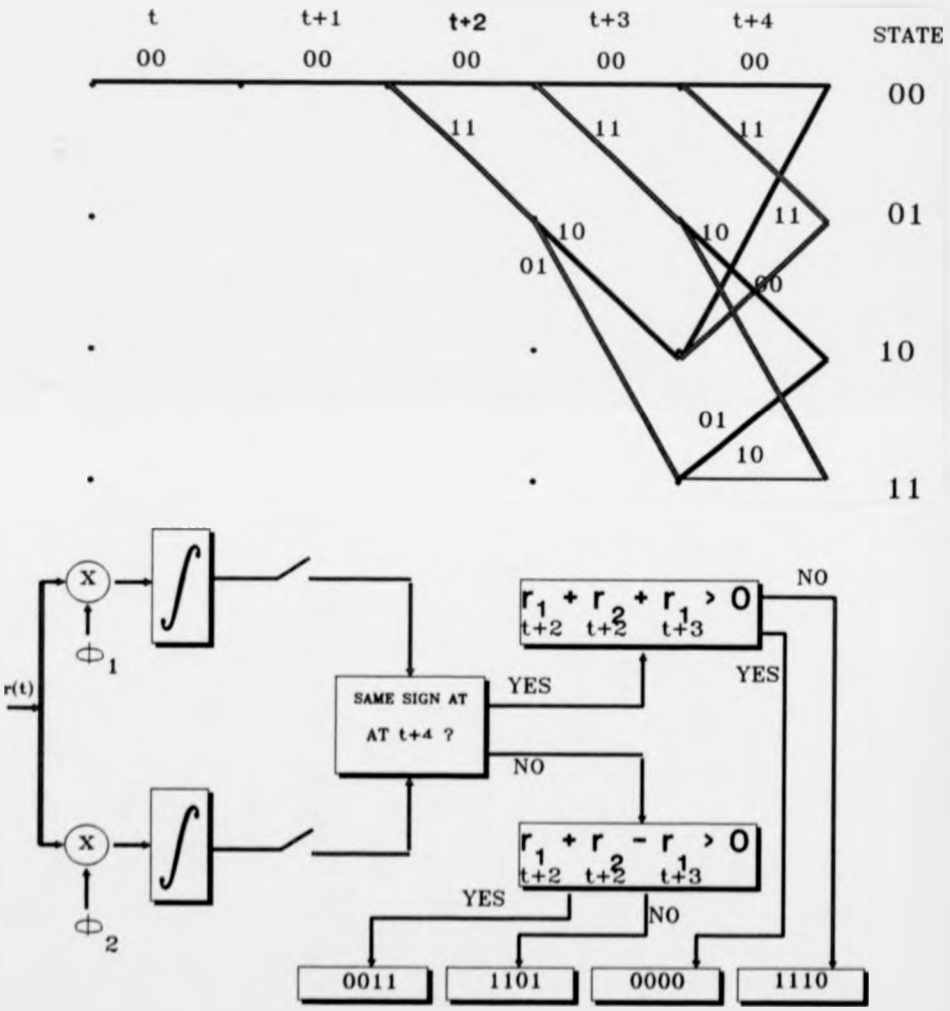


Figure 6.5: Decoding block diagram starting at state 00 (t+1)

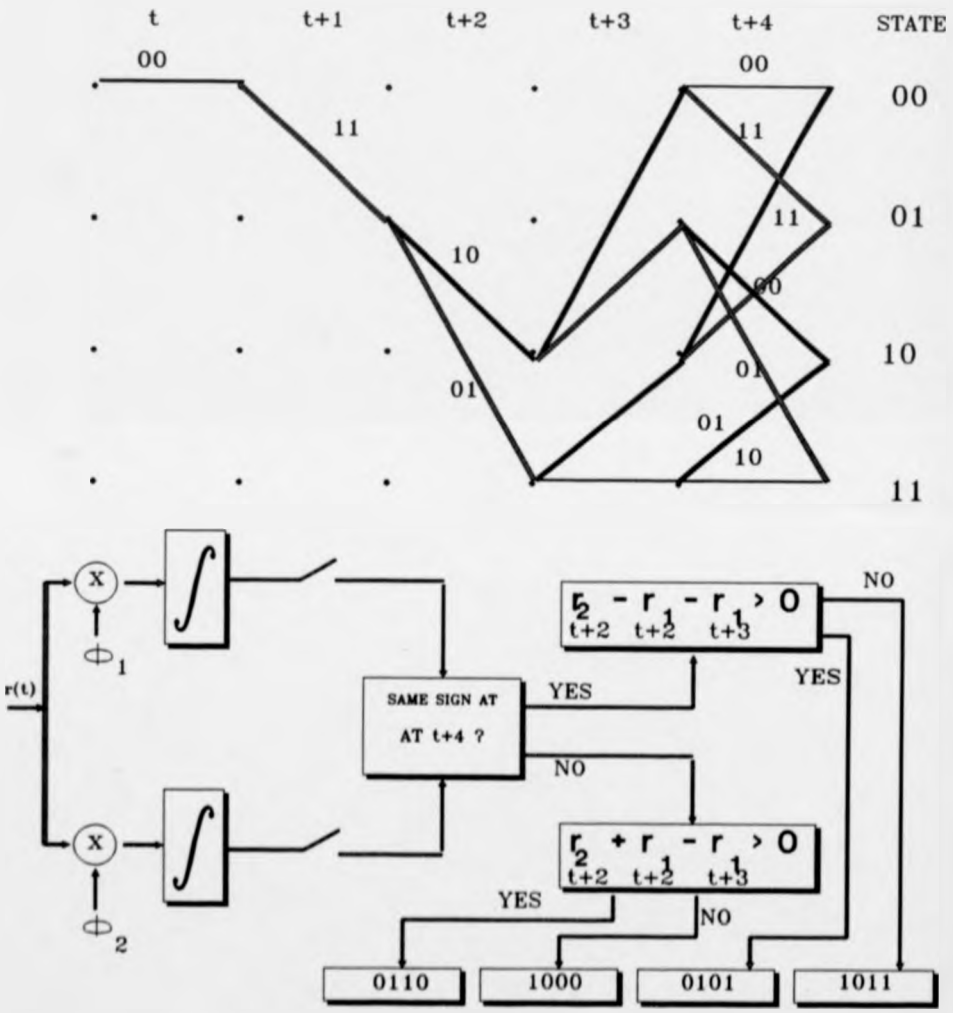


Figure 6.6: Decoding block diagram starting at state 01 (t+1)

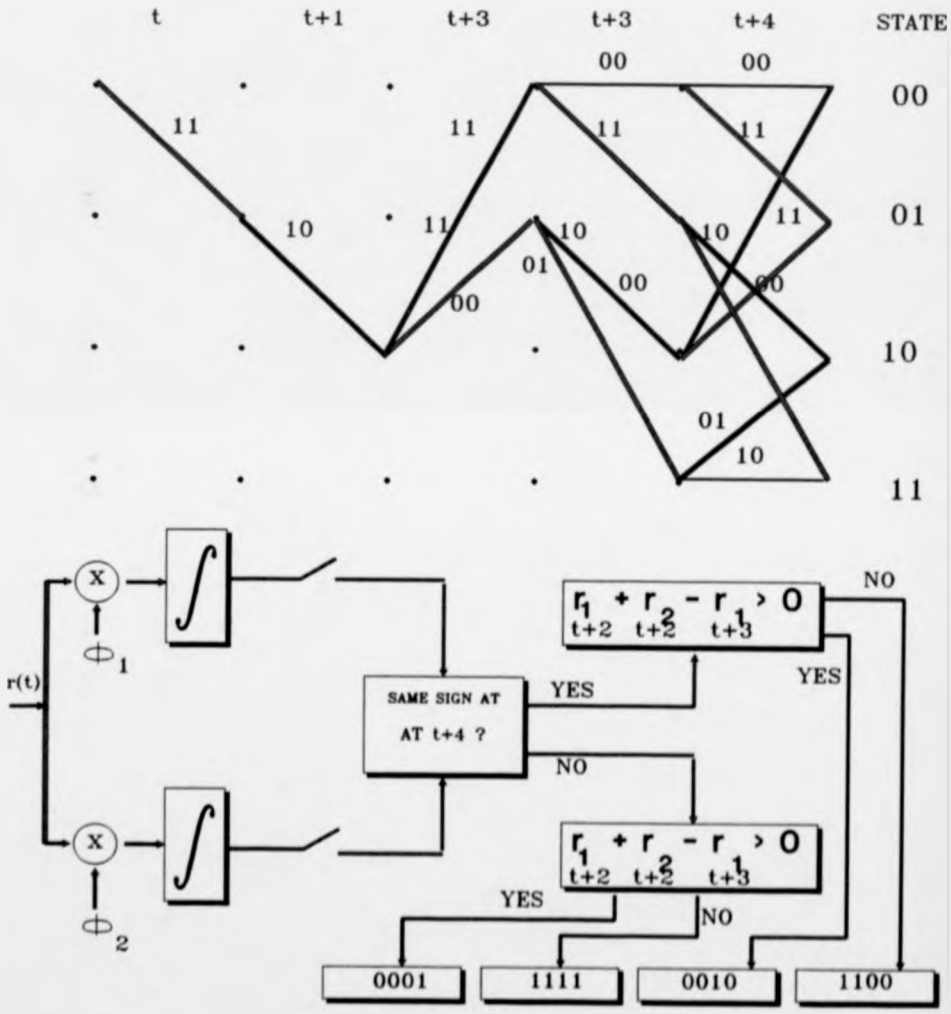


Figure 6.7: Decoding block diagram starting at state 10 (t+1)

htp

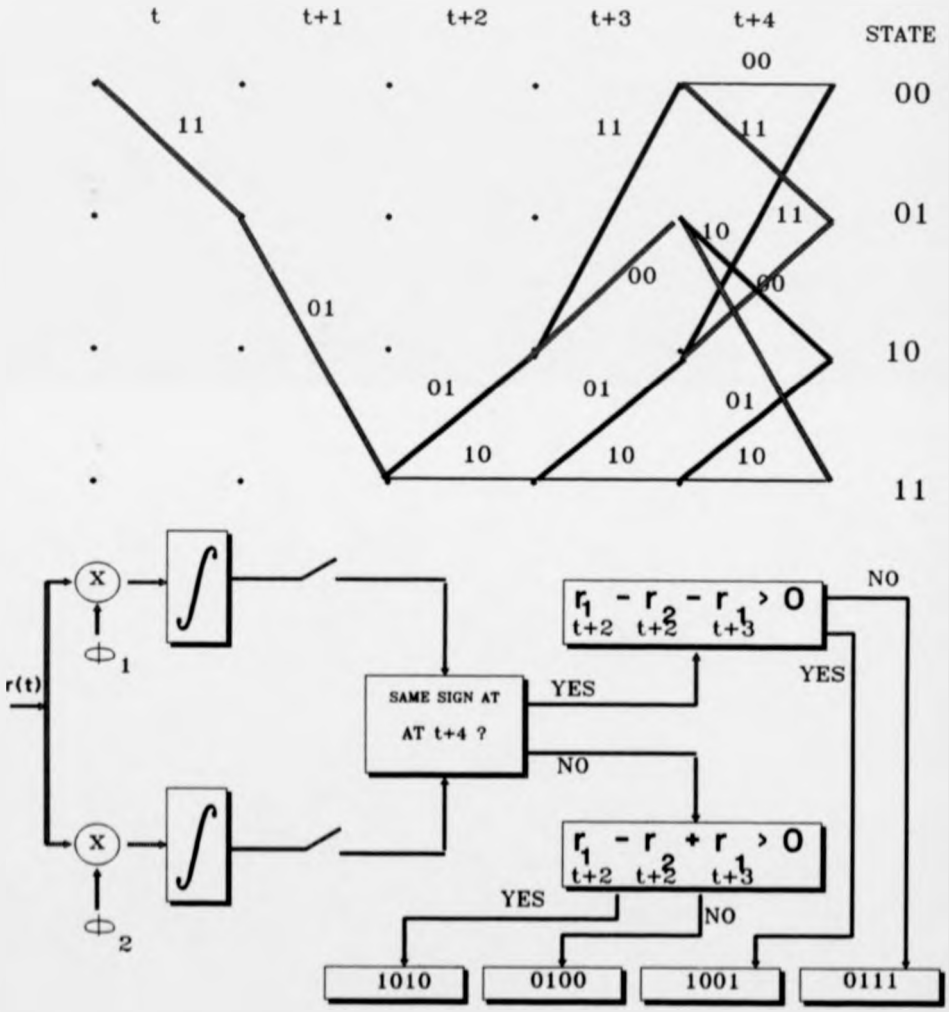


Figure 6.8: Decoding block diagram starting at state 11 (t+1)

Therefore 0000 were the transmitted symbols. We are at state 00 in the trellis. To proceed further we should now refer to the corresponding look-up table for state 00. This is shown in Fig.6.5.

Do r_1 and r_2 at $t+4$ have the same sign? Answer: YES. Therefore

$$\text{is } r_{1_{t+2}} + r_{2_{t+2}} + r_{1_{t+3}} \geq 0$$

$$\text{i.e. } \frac{-\sqrt{2}}{2} - \frac{\sqrt{2}}{2} - \frac{\sqrt{2}}{2} = \frac{-3\sqrt{2}}{2} \text{ which is less than 0.}$$

Therefore 1110 were the transmitted symbols.

By considering the decoded symbol in the first part, the decoded sequence is 00001110.

It can be seen that the two single errors at t=1 and t=3 have been detected and corrected.

A simple soft decision decoding algorithm proposed in this chapter enables us to eliminate the process of storage of Euclidean distances for survivor paths at each node in Viterbi algorithm. This algorithm exploits the trellis structure which at each node has certain permissible and non-permissible codewords. With a simple comparison based on soft decision information, a decision can be made at each node in the trellis diagram for the previous symbol which has merged. The performance of this method is the same as the conventional Viterbi algorithm, which requires the storage of metric values at each node in the trellis, because the mathematical basis of both methods is identical although they differ in the processing operation.

6.3 Computational complexity

Massey [Massey, (1978)] has defined three quantities which are useful in the determination of a trellis decoder complexity. The *number of digit metrics* N_m to be evaluated in order to compute path metrics is equal to the total number of digits on all branches of the trellis. The *number of metric comparisons* N_c is just the number of nodes in the trellis which have two entering branches. Finally, the *number of accumulators* N_a is equal to the maximum number of nodes or states which must be stored at any given node level as the decoder progress through the trellis.

For maximum likelihood decoding of convolutional codes by the Viterbi algorithm, complexity is easily defined [Viterbi *et al.* (1979)], for if the constraint length is L and the code rate is k/n, the number of branch metric computations (N_m) per branch (k bits) is 2^{Lk}, while the number of comparisons N_c, and the number of storage registers (N_a) required for path memories and metrics, is $2^{(L-1)k}$.

We now examine how the computational quantities are affected by the trellis decoding method presented here. Consider the numerical example in section 6.2.3. In order to perform decoding at the $t + 2$ stage, using the conventional technique, we require $N_m = 8$ branch metric computations, four metric comparison (N_c) and four storage registers (N_a). Comparing the number of operations at $t + 2$ in Fig 6.4, it is clear that the number of comparison required is reduced to two and only two storage registers for

storing the inphase and quadrature components of the received symbols at t and $t+1$ are required.

The advantage of the new method will be felt more, as the constraint length of the code gets larger, because the number of decoding calculations and hence the storage is proportional to 2^{L-1}.

Unfortunately, the computational complexity of the Viterbi algorithm increases with the width of the trellis to be searched. In this chapter, a search length of up to $t+4$ was considered, but this technique can be applied to greater search length. The general form of this technique is shown in Appendix A [Arani *et al*, (1989)].

The only requirement of this new method is to know the position of the decoder regarding its state at any particular time which is the same requirement as in the case of usual Viterbi algorithm.

The implementation of this decoder on digital signal processing (DSP) board is expected to be much simpler than the conventional ACS circuitry for the Viterbi algorithm.

Chapter 7

Conclusion and Suggestions for Further Work

7.1 Introduction

The research described in this thesis investigated the use of trellis coded modulation to develop various techniques applicable to digital data transmission systems. Throughout this work, emphasis has been placed on enhancing the performance or complexity of conventional communication systems by simple modifications to the existing structures. The conclusions which have been obtained from the study, including further research topics for the future, are presented in the following sections.

7.2 Asymmetrical TCM

In chapter 3, using an explicit example, it was shown that by properly designing the signal constellation for TCM systems, performance improvement can be obtained over systems with conventional symmetrical signal constellations. Also, the relation between optimum constellation angle and channel signal to noise ratio which minimises bit error rate was found. However, in certain cases of asymmetry, the asymptotic improvement can only be achieved in the limit as points in the signal constellation merge together; i.e. the trellis code becomes catastrophic.

The performance of block coded modulation employing an array code and MPSK modulation was considered. It was shown that an asymptotic coding gain of 3 dB can be achieved over an uncoded scheme. This performance gain was based on the comparison of the minimum free Euclidean distance of

the trellis code relative to the minimum distance of the uncoded modulation. This is an indication of the maximum reduction in required bit energy-to-noise spectral density ratio E_b/N_0 that can be achieved for arbitrarily small system bit error rates. Also simulation results for block coded modulation employing (3,2)(3,2) array code were presented and compared with TCM employing convolutional code with the same number of trellis states. It was found that the performance of block coded modulation was poorer than that of trellis coded modulation using convolutional codes, because, the coded bits can not be matched to the signal points in the trellis structure of array code in order to maximise the squared free Euclidean distances.

The asymmetrical signal constellation was applied to BC and the effect of asymmetry using an array code was examined. It was found that an optimum constellation angle for BC employing a (3,2)(3,2) array code, 4-state trellis and 4-PSK signal is 90 degree.

Two main extensions of asymmetrical constellations can be identified for further work. Firstly, the asymmetrical M signal set used was created by adding together the symmetrical M/2 point set with a rotated version of itself. The effect of totally asymmetrical signal constellation on system performance might be investigated. Secondly, there is potential use of asymmetrical constellation points for unequal error protection of information.

7.3 Multi-dimensional TCM

Chapter 4 presents the concept of multidimensional signal sets in conjunction with TCM. The purpose of this chapter is twofold. First, it has been shown that an increased noise margin may be derived from multidimensional constellation itself. Also, performance improvement can be achieved by using non-binary convolutional codes. It has been shown that by mapping the output of a convolutional code over Z_4 to BFSK-8PSK in 4-D signal space, as much as 2.7 dB gain can be achieved. This comparison has been done by considering a binary convolutional encoder of the same rate and having a similar shift register structure. In comparison with uncoded BFSK-4PSK modulation scheme, a coding gain of 5.7 dB can be achieved. Multidimensional signalling is an attractive proposition for an improvement in performance of TCM techniques.

It should be noted that coding optimisation has not been considered in any of these schemes. It has been shown that a convolutional code over rings can have no catastrophic polynomial encoder [Massey *et al.* (1989)]. Code optimisation can be carried out as a further work in this area and perhaps additional coding gain can be achieved. Also the effect of asymmetrical constellation points in conjunction with convolutional codes over rings might be investigated as a future work.

7.4 Adaptive TCM

In chapter 5, an embedded trellis coded modulation format was presented. In this scheme, a fixed convolutional encoder was used with three different modulation schemes. As a result, the three possible trellis structure arrangements can generate adaptive TCM codes for error correction and detection in conjunction with ARQ. The error detection is performed via an RTCE technique which is a by-product of the Viterbi algorithm.

Unlike the embedded convolutional coding technique [Zolghadr *et al.* (1988)], embedded trellis coded modulation does not require code puncturing and subsequently removes the need for insertion of erasures at the receiver.

For the purpose of practical implementation of this scheme, a reliable synchronisation scheme is required. Code assisted bit synchronisation (CABS) [Honary *et al.* (1989), Honary *et al.* (1992)] is an attractive technique which can be implemented at the receiver for this purpose. This technique derives symbol timing recovery from the received signal, without any additional overheads. It is therefore necessary to investigate the performance of CABS with ETCM schemes.

Two main extensions of the current work can be identified. Firstly, the use of asymmetrical embedded trellis coded modulation schemes should be extensively studied.

Secondly, the use of block coded modulation may provide more immunity against burst errors experienced over HF channels, for example.

7.5 Low Complexity TCM

In chapter 6, a new decoding procedure for the Viterbi algorithm has been presented. This procedure involves the use of inphase and quadrature components of the received signal directly in the decoding process. The main advantage of the new scheme is that, during each decoding process, the decoder now only stores the inphase and quadrature components of the received symbol. Hence, this technique reduces the number of calculations for metric evaluation at each trellis node.

A number of look-up tables are required for decoding procedure. The complexity of these tables depend on the decoding depth. A simple case for a decoding depth four branches long has been presented. However, this scheme can be extended to larger search lengths. The general case of this decoding procedure for a given search length is shown in the appendix.

An obvious extension of this scheme is to generate a general form of decoding procedure for any given search length or constraint length, and to combine it with other suitable complexity reducing techniques.

Bibliography

[Arani *et al.* (1989)] Arani,F., G.H.Khachatrian,(1989),(Armenian Academy of Sciences, CIS), Low Complexity Soft Decision Trellis Decoding, private communication.

[Arani *et al.* (1990)] Arani,F., Honary,B.(1990) Optimisation of Constellation Angles for Trellis Coded 4-PSK Signals, Proc. of Bangor Communications Symposium, Bangor, Gwynedd, 23-24 May.

[Arani *et al.* (1991)] Arani,F., Honary,B.(1991), Low complexity soft decision maximum likelihood decoding algorithm for convolutional codes, IEE Third Int. Conf. on Telecommunications, Edinburgh, U.K. 17-20 March .

[Arani *et al.* (1991a)] Arani,F., Honary,B. Markarian,G. (1991) 3-D Trellis Coded Modulation, Proc. of Bangor Communications Symposium, Bangor, Gwynedd, 23-24 May.

[Arani *et al.* (1993)] Arani,F., Honary,B. (1993), Trellis coded modulation over rings, Proc. of Bangor Communications Symposium, Bangor,

Gwynedd, 2-3 June.

[Arthurs *et al.* (1962)] Arthurs. E. and Dym. H., (1962), On the optimum detection of digital signals in the presence of white Gaussian noise, IRE Trans. on Communications Systems, Dec.1962

[Bahl *et al.* (1974)] Bahl, L.R., Cocke,J., Jelinek,F. and Raviv,J., (1974), Optimum decoding of linear codes for minimizing symbol error rate, IEEE Trans. Inform. Theory, vol. IT-20, pp.284-287, March.

[Benedetto *et al.* (1987)] Benedetto, S., Biglieri,E., and Castellani, V., (1987). Digital Transmission theory,Englewood Cliffs, NJ, Prentice-Hall Publ. Co.

[Biglieri, (1984)] Biglieri, E. (1984) High-level modulation and coding for nonlinear satellite channels, IEEE Trans. Commun., vol COM-32, No.5, pp.616-626.

[Biglieri *et al.* (1988)] Biglieri, E., and Elia, M., Multidimensional modulation and coding for bandlimited digital channels, IEEE Trans. Inf.Theory, vol. IT-34, pp. 803-909, July 1988.

[Biglieri *et al.* (1988)] Biglieri,E. and Liu,Y.J., (1988), On the analytical representation of trellis codes, MILCOM'88, san Diego, CA.

[Cain *et al.* (1979)] Cain,J.B., Clark, G.C.,Jr., and J.M.Geist, (1979), Punctured convolutional codes of rate (n-1)/n and simplified maximum like-

lihood decoding, IEEE Trans. on Info. Theory, vol. IT-25, No.1, pp 97-100, Jan.

[Calderbank *et al.* (1984)] Calderbank,A.R. and Mazo,J.E., (1984), A new description of trellis codes,IEEE Trans. Inform. The, vol. IT-30, pp.784-791

[Calderbank *et al.* (1987)] Calderbank, A.R. and Sloane, N.J.A., (1987), New Trellis Codes Based on Lattices and Cosets, IEEE Trans. on Inform. Theory, vol. IT-33, No. 2, March.

[Chase, (1972)] Chase, D., (1972), A class of algorithms for decoding block codes with channel measurement information, IEEE Trans. Inform. Theory, vol. IT-18, pp.170-182, Jan.

[Clark *et al.* (1978)] Clark, A. P., Harvey, J. D., and Driscoll, J. P., (1978), Near Maximum Likelihood Detection Processes for Distorted Digital Signals, Radio & Electron. Eng., 48, pp. 301-309.

[Clark, (1977)] Clark, A. P., (1977), Advanced Data Transmission Systems, Pentech Press.

[Clark *et al.* (1988)] Clark,G.C. and Cain,J.B., (1988), Error-correction coding for digital communications, Plenum Press, New York.

[Clarke, (1968)] Clarke,R.H., (1968), A statistical theory of mobile-radio reception, Bell Syst. Tech. J.,vol. 47, pp.957-1000.

[Darnell, (1983)] Darnel, M., (1983), Embedded Real-time channel evaluation techniques, AGARD lecture series No. 127, on modern HF communications, May-June.

[Darnell *et al.* (1988a)] Darnell, M., Honary, B., Zolghadr.F., (1988) Embedded coding technique: Principles and theoretical studies, IEE Proc., vol.135, pt.F, No.1. Feb.

[Darnell *et al.* (1988b)] Darnell.M., Honary,B., Zolghadr.F., (1988) Embedded array coding for HF channels, theoretical and practical studies, Proc. of IMA on Cryptography and coding, Oxford Univ. Press.

[Divsalar *et al.* (1980)] Divsalar, D., and Yuen, J.H., (1980), Performance of convolutional coded unbalanced QPSK systems, Proc. National Telecommunications Conference, Houston,Texas, pp. 14.5.1 - 14.5.7, Nov.

[Divsalar *et al.* (1984)] Divsalar, D., and Yuen, J.H., (1984), Asymmetrical MPSK for trellis codes, Proc. Globecom '84, Atlanta, Ga., pp. 20.6.1 - 20.6.8, Nov.

[Divsalar, (1988)] D.Divsalar, (1988), Multiple Trellis Coded Modulation (MTCM), IEEE Transaction on Communications,Vol.36,No.4,April.

[Divsalar *et al.* (1988a)] Divsalar,D., Simon, M.K., (1988), The design of trellis coded MPSK for fading channels: Performance Criteria, IEEE Trans. on Com., Vol.36, No.9, Sept.

[Divsalar *et al*, (1988b)] Divsalar,D., Simon, M.K.,Jedrey, T., (1988), Trellis coding techniques for mobile communications, MILCOM 88, IEEE Military Communications Conference, Vol-2, Oct 23-26, San Diego, USA.

[Dorsch, (1970)] Dorch,B., (1970), Maximum likelihood decoding of binary group codes for the Gaussian channel, presented at IEEE Int. Symp. on Inform. Theory, Noordwijk, The Netherlands.

[Elias, (1955)] Elias.P., (1955), Coding for noisy channels, IRE Convention Record,vol. 3, part 4, pp. 37-46.

[Fano, (1963)] Fano. R.M., (1963), A heuristic discussion of probabilistic decoding, IRE Trans. Inf. Theory, vol. IT9, no. 2, pp. 64-74

[Farrell, (1979)] Farrell, P.G., (1979), Array Codes, Algebraic coding theory and applications, Ed. G.Longo, Springer-Verlog, pp. 231-242.

[Forney, (1966)] Forney Jr. G.D., (1966), Concatenated codes, MIT Press.

[Forney, (1972)] Forney Jr.,G.D., (1972), Maximum likelihood sequence estimation of digital sequences in the presence of intersymbol interference, IEEE Trans. Inform. Theory, vol. 25, IT-18, pp.363-378, May.

[Forney, (1973)] Forney. G.D., (1973), The Viterbi Algorithm, Proc. of the IEEE, Vol.61, No.3, pp.268-278, March.

[Forney *et al.* (1984)] Forney G.D. Jr. *et al.* (1984), Efficient modulation for bandlimited channels, IEEE J. Sel. Areas Commun., vol. SAC2, no. 5, Sept. pp. 632-647.

[Foschini, (1977)] Foschini, G. J., (1977), A Reduced State Variant of Maximum Likelihood Sequence Detection Attaining Performance for High Signal to Noise Ratio, IEEE Trans., IT-23, pp.605-609.

[Franks, (1981)] Franks.L.E., (1981), Signal theory, Revised Edition, Dowden & Culver.

[Gallager, (1968)] Gallager, R. G., (1968), Information Theory and Reliable Communication, J.Wiley.

[Gersho *et al.* (1984)] Gersho, A., and Lawrence, V., (1984), Multidimensional signal design for digital transmission over bandlimited channels, Proceedings of ICC'84, Amsterdam, The Netherlands, pp.377-380, May.

[Hagenauer, (1980)] Hagenauer, J., (1980), Viterbi decoding of convolutional codes for fading and bursty channels, in Conf. Rec. 1980 Zurich Seminar Digital Commun., pp. G2.1-G2.7.

[Hagenauer *et al.* (1988)] Hagenauer, J., Seshadri,N., Sunberg, W., (1988), The performance of rate compatible punctured convolutional codes for future digital mobile radio, in Conf. Rec. IEEE VTC'88, pp.22-29.

[Hartmann *et al.* (1976)] Hartmann, C.R.P. and Rudolph, L.E., (1976), On optimum symbol by symbol decoding rule for linear codes, IEEE Trans. Inform. Theory, vol. IT-22, pp.514-517, Sept.

[Heller *et al.* (1971)] Heller, J. A., and Jacobi, I.M., Viterbi, A., (1971), Decoding for Satellite and Space Communication, ibid., 1971, COM-19, pp. IEEE, 1973, 61, pp.268-278.

[Heller, (1975)] Heller .J.A., (1975), Feedback decoding of convolutional codes, in Advances in communication systems, vol. 4, Academic, New York.

[Honary *et al.* (1989)] Honary,B., Zolghadr,F., Darnell,M., A, (1989), Code-Assisted Bit synchronisation (CABS) scheme, Electronics Letters, vol. 25, No.23, 19th Jan.

[Honary *et al.* (1990)] Honary,B., Arani, F., Darnell, M., (1990), Embedded Modulation and Coding for HF Channels, Proc. of IMA Conf. on Cryptography and Coding, Oxford Univ. Press.

[Honary *et al.* (1992)] Honary,B., Arani,F., Darnell,M., (1992), MFSK and MPSK Modems with Intrinsic Synchronization, IEEE MILCOM' 92, San Diego, CA, USA, 11-14 Oct.

[Honary *et al.* (1993)] Honary,B., Kaya,L., Markarian,G.S. and Darnell,M., (1993), Maximum likelihood decoding of array codes with trellis structure, will appear in IEE Proc. I.

[Jacobs, (1967)] Jacobs, I., (1967) Comparison of M-ary modulation systems, Bell System Technical Journal,vol.46, pp.843-864, May-June.

[Jelinek(1969)] Jelinek. F., (1969), Fast sequential decoding algorithm using a stack, IMB J. Res. Dev., vol.13, Nov, pp. 675-685.

[Kahn *et al.* (1978)] Kahn, R.E., Gronemeyer, S.A., Burchfield,J., and Kunzelman, R.E., (1978), Advances in packet radio technology, Proc. IEEE, vol.66, pp.1468-1487, Nov.

[Klieber, (1970)] Klieber. E.J. (1970), Some difference triangles for constructing self orthogonal codes,IEEE Trans. Inf. Theory, IT-16, March, pp. 237-238.

[Kotelnikov, (1959)] Kotelnikov, V.A., (1959), The theory of optimum noise immunity, McGraw-Hill, New York. (Doctoral dissertation presented in January 1947 before the academic council of the Molotov energy institute in Moscow.)

[Lin *et al.* (1983)] Lin.S. and Costello. D.J. Jr. (1983), Error control coding: Fundamentals and applications, Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

[Lin *et al.* (1984)] Lin, S., Costello, D.J.,Jr and Miller, M.J., (1984), Automatic repeat request error-control schemes, IEEE Communication Magazine, vol.22, No.12, pp. 5-17, Dec.

[Lugand *et al.* (1989)] Lugand, L., Costello, D.J, Jr, Deng, R.H., (1989), Parity retransmission hybrid ARQ using rate 1/2 convolutional codes on a nonstationary channel, IEEE Trans. on Communications, vol.37, No.7, July.

[Massey, (1963)] Massey, J.L. (1963), Threshold decoding, MIT Press, Cambridge, Mass.

[Massey, (1968)] Massey, J.L. (1968),catastrophic error-propagation in convolutional codes, Proceedings, Eleventh Midwest symposium on circuit Theory, University of Notre Dame, Notre Dame, Indiana, May 13-14, Ave Maria Press, Notre Dame, Indiana.

[Massey, (1978)] Massey.J.L. (1978), Fondations and methods of channel coding, Int. Conf. on Information Theory and Systems. NTC-FACHBENCHTE, vol.65.

[Massey(1986)] Massey, J.L. (1986), Coding and modulation in digital communication,International Zurich Seminar in digital communication, Zurich, Switzerland, March.

[Massey *et al.* (1989)] Massey, J.L., Mittelholzer, T., (1989), Convolutional codes over rings, Proc. of 4th Joint Swedish-Soviet Int. workshop on Information Theory, Gotland, Sweden, pp.14-18, Aug. 27- Sept. 1.

[Miyakawa *et al.* (1975)] Miyakawa, H. and Kaneko, T., (1975), Decoding algorithm for error correcting codes by use of analog weights, Electron. Commun, Japan, vol.58-A, pp.18-27.

[Nyquist, (1928)] Nyquist,H., (1928), Certain topics in telegraph transmission theory, Trans. AIEE, vol.47, pp.617-644, April.

[Oldenwalder, (1976)] Odenwalder,J.P., (1976), Dual-k convolutional codes for noncoherently demodulated channels, in Proc. Int. Telemetering Conf., vol 12, pp. 165-174.

[Omura, (1969)] Omura, J.K., (1969), On the Viterbi Decoding algorithm,IEEE Trans., IT-15, pp.177-179.

[Omura, (1981)] Omura, J.K., (1981), Performance bounds for Viterbi algorithms, in Conf. Rec.,Int. Conf, Commun., Denver, CO, pp.2.2.1-2.2.5.

[Peterson *et al.* (1972)] Peterson,W.W. and Weldon.E.J., (1972), Jr. Error Correcting Codes, 2d ed. MIT Press, Cambridge, Mass.

[Robinson *et al.* (1967)] Robinson, J.P. and Bernstein A. J. (1967), A class of binary recurrent codes with limited error propagation, IEEE Trans. Inf. Theory, IT-13, January, pp.106-113.

[Russell *et al.* (1976)] Russell, S. P., and Shohara, A., (1976), Feasibility Study of Reduced State Viterbi Detection, Eason 1976 Record, Washington, USA, Sept., pp. 89A-F.

[Ryan *et al.* (1991)] Ryan,W.E. and Wilson.S.G., (1991), Two classes of convolutional codes over GF(q) for q-ary orthogonal signalling, IEEE Trans. on Comm., vol.39, no.1.

[Shannon, (1948)] Shannon.C.E., (1948), A mathematical theory of communication, Bell System Tech. J., vol.27, (pt.I), pp.379-423, (pt.II), pp.623-656.

[Shannon, (1949)] Shannon, C.E., (1949), Communication in the presence of noise, Proc. IRE, Vol. 37, pp. 10-21, Jan.

[Slepian, (1963)] Slepian, D., Bounds on Communication, Bell System Technical Journal, vol.42, pp.681-707, May 1963.

[Slepian, (1968)] Slepian, D., (1968), Group codes for the Gaussian channel, Bell System Technical Journal, vol.47, pp.575-602, April.

[Ungerboeck, *et al.* (1976)] Ungerboeck.G., Csajka.I.(1976), On improving data-link performance by increasing the channel alphabet and introducing sequence coding,Int. Symp. Inform. Theory, vol. IT-28, pp. 55-67.

[Ungerboeck, (1982)] Ungerboeck.G. (1982), Channel Coding with multilevel/phase signals, IEEE Trans. Information Theory, vol. IT-28, pp. 55-67.

[Viterbi, (1967)] Viterbi. A.J. (1967), Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, IEEE Trans. Inform. Theory, vol. IT-13, pp. 260-269, April.

[Viterbi, (1967)] Viterbi,A.J., (1967), Orthogonal tree codes for communication in the presence of white Gaussian noise, IEEE Trans. Commun. Tech., vol. COM-15, pp. 238-242, Apr.

[Viterbi, (1969)] Viterbi. A.J. (1969), The state diagram approach to optimal decoding and performance analysis for memoryless channels, Jet Propulsion Laboratory, Space programs Summary, 37-58, vol. III, pp. 50-55.

[Viterbi, (1971)] Viterbi. A.J. (1971), Convolutional codes and their performance in communication systems, IEEE Trans. Comm. Tech., COM-19, pp. 751-772.

[Viterbi, (1972)] Viterbi,A.J., (1972), Two constructive classes of convolutional codes for multiple-signal channels, presented at the IEEE Int. Symp. Inform. Theory, Pacific Grove, CA, Jan.

[Viterbi *et al*, (1975)] Viterbi,A.J., and Jacobs,I.M., (1975), Advances in coding and modulation for noncoherent channels affected by fading, partial band and multiple-access interferences, in Advances in Communication Systems, vol.4, New York:Academic.

[Viterbi *et al.* (1979)] Viterbi, A.J. and J.K.Omura, J.K., (1979). Principles of digital Communications and coding, McGraw-Hill, New York.

[Wainberg *et al.* (1973)] Wainberg, S., and Wolf, J.K., (1973), Algebraic decoding of block codes with a q-ary input, Q-ary output channel, Informat. Contr, vol-22, pp.243-247, Apr.

[Weldon, (1976)] Weldon, E.J., (1976), Decoding block codes on Q-ary output channels, IEEE Trans. Inform., Theory, vol.IT-17, pp.713-718, Nov.1976

[Welti *et al.* (1974)] Welti, G.R., and Lee, J.S., (1974), Digital transmission with coherent four dimensional modulation, IEEE Trans. on Inform. Theory, vol.IT-20, pp.497-502, July.

[Wiener, (1949)] Wiener, N., (1949), The extrapolation, interpolation, and smoothing of stationary time series with engineering applications, Wiley, New York. (Original work appeared as an MIT radiation laboratory report in 1942.)

[Wilson *et al.* (1983)] Wilson, S.G., and Sleeper, H.A., (1983), Four dimensional modulation and coding: An alternate to frequency reuse, Rept. No. UVA/528200/EE83/107, Communications systems Laboratory, Dept. of Electrical Engineering, University of Virginia, Charlottesville, VA, September.

[Wilson *et al.* (1984)] Wilson,S.G., Sleeper, H.A., and Srinath,N.K., (1984), Four dimensional modulation and coding : An alternate to frequency reuse, Proceedings of ICC'84, Amsterdam, The Netherlands, pp.919-923, May.

[Wolf, (1978)] Wolf, J.K., (1978), Efficient maximum likelihood decoding of linear block codes using a trellis, IEEE Trans. Inform. Theory, vol. IT-24, pp.76-80.

[Wozencraft, (1957)] Wozencraft,J.M. (1957), Sequential decoding for reliable communication, IRE Natl. Conv. Rec., vol. 5, pt.2, pp. 11-25.

[Wozencraft *et al.* (1961)] Wozencraft, J.M. and Reiffen, B. (1961), Sequential decoding, The MIT Press, Cambridge, Mass.

[Wozencraft *et al.* (1965)] Wozencraft, J.M. and Jacobs, Irwin. M. (1965), Principles of communication engineering, Wiley, New York.

[Yamamoto *et al.* (1980)] Yamamoto, H., Itoh, K. Viterbi,A.J. (1980), decoding algorithm for convolutional codes with repeat request, IEEE Trans., IT-26, No.5 pp. 540-747.

[Yasuda *et al.* (1983)] Yasuda, Y. Kashiki, K. Hirata, Y., (1983), Development of variable rate Viterbi decoder and its performance characteristics., Sixth Int. Conf. on satellite Comm. Phoenix AZ., sept.

[Zetterberg *et al.* (1977)] Zetterberg, L.H., and Brandstrom, H., (1977), Codes for combined phase and amplitude modulated signals in a four

dimensional space, Trans. on Communications,vol. COM-25, pp.943-950, Sept.

[Zigangirov, (1966)] Zigangirov, K.S. (1966), Some sequential decoding procedures, Probl. Peredach. Inform.,vol. 2, pp.13-15.

[Zolghadr, (1989)] Zolghadr,F., (1989), Embedded coding algorithms applicable to time variable channels, University of Warwick, Ph.D. Thesis, Nov.

[Zolghadr *et al.* (1988)] Zolghadr, F., Honary, B., Darnell, M., (1988), Embedded convolutional coding, Proc. of the IERE Fifth Int. Conf. on Digital Processing of Signals in Comm., Sept.

Appendix A

Generalised Low Complexity Convolutional Decoding Algorithm

A.1 State 00

$$R_{00}^{1_{t+4}} = d_{00_t}^2 + d_{00_{t+1}}^2 + d_{00_{t+2}}^2 + d_{00_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.1)}$$

$$R_{01}^{2_{t+4}} = d_{00_t}^2 + d_{00_{t+1}}^2 + d_{00_{t+2}}^2 + d_{00_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.2)}$$

$$R_{10}^{3_{t+4}} = d_{00_t}^2 + d_{00_{t+1}}^2 + d_{00_{t+2}}^2 + d_{11_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.3)}$$

$$R_{11}^{4_{t+4}} = d_{00_t}^2 + d_{00_{t+1}}^2 + d_{00_{t+2}}^2 + d_{11_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.4)}$$

$$R_{01}^{5_{t+4}} = d_{00_t}^2 + d_{00_{t+1}}^2 + d_{11_{t+2}}^2 + d_{10_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.5)}$$

$$R_{00}^{6_{t+4}} = d_{00_t}^2 + d_{00_{t+1}}^2 + d_{11_{t+2}}^2 + d_{10_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.6)}$$

$$R_{11}^{7_{t+4}} = d_{00_t}^2 + d_{00_{t+1}}^2 + d_{11_{t+2}}^2 + d_{01_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.7)}$$

$$R_{10}^{8_{t+4}} = d_{00_t}^2 + d_{00_{t+1}}^2 + d_{11_{t+2}}^2 + d_{01_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.8)}$$

$$R_{10}^{9_{t+4}} = d_{00_t}^2 + d_{11_{t+1}}^2 + d_{10_{t+2}}^2 + d_{00_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.9)}$$

$$R_{11}^{10_{t+4}} = d_{00_t}^2 + d_{11_{t+1}}^2 + d_{10_{t+2}}^2 + d_{00_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.10)}$$

$$R_{00}^{11_{t+4}} = d_{00_t}^2 + d_{11_{t+1}}^2 + d_{10_{t+2}}^2 + d_{11_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.11)}$$

$$R_{01}^{12_{t+4}} = d_{00_t}^2 + d_{11_{t+1}}^2 + d_{10_{t+2}}^2 + d_{11_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.12)}$$

$$R_{11}^{13_{t+4}} = d_{00_t}^2 + d_{11_{t+1}}^2 + d_{01_{t+2}}^2 + d_{10_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.13)}$$

$$R_{11}^{14_{t+4}} = d_{00_t}^2 + d_{11_{t+1}}^2 + d_{01_{t+2}}^2 + d_{10_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.14)}$$

$$R_{01}^{15_{t+4}} = d_{00_t}^2 + d_{11_{t+1}}^2 + d_{01_{t+2}}^2 + d_{01_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.15)}$$

$$R_{00}^{16_{t+4}} = d_{00_t}^2 + d_{11_{t+1}}^2 + d_{01_{t+2}}^2 + d_{01_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.16)}$$

$$R_{01}^{17_{t+4}} = d_{11_t}^2 + d_{10_{t+1}}^2 + d_{00_{t+2}}^2 + d_{10_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.17)}$$

$$R_{00}^{18_{t+4}} = d_{11_t}^2 + d_{10_{t+1}}^2 + d_{00_{t+2}}^2 + d_{10_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.18)}$$

$$R_{11}^{19_{t+4}} = d_{11_t}^2 + d_{10_{t+1}}^2 + d_{00_{t+2}}^2 + d_{01_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.19)}$$

$$R_{10}^{20_{t+4}} = d_{11_t}^2 + d_{10_{t+1}}^2 + d_{00_{t+2}}^2 + d_{01_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.20)}$$

$$R_{00}^{21_{t+4}} = d_{11_t}^2 + d_{10_{t+1}}^2 + d_{11_{t+2}}^2 + d_{00_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.21)}$$

$$R_{01}^{22_{t+4}} = d_{11_t}^2 + d_{10_{t+1}}^2 + d_{11_{t+2}}^2 + d_{00_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.22)}$$

$$R_{10}^{23_{t+4}} = d_{11_t}^2 + d_{10_{t+1}}^2 + d_{11_{t+2}}^2 + d_{11_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.23)}$$

$$R_{11}^{24_{t+4}} = d_{11_t}^2 + d_{10_{t+1}}^2 + d_{11_{t+2}}^2 + d_{11_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.24)}$$

$$R_{11}^{25_{t+4}} = d_{11_t}^2 + d_{01_{t+1}}^2 + d_{10_{t+2}}^2 + d_{10_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.25)}$$

$$R_{10}^{26_{t+4}} = d_{11_t}^2 + d_{01_{t+1}}^2 + d_{10_{t+2}}^2 + d_{10_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.26)}$$

$$R_{01}^{27_{t+4}} = d_{11_t}^2 + d_{01_{t+1}}^2 + d_{10_{t+2}}^2 + d_{01_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.27)}$$

$$R_{00}^{28_{t+4}} = d_{11_t}^2 + d_{01_{t+1}}^2 + d_{10_{t+2}}^2 + d_{01_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.28)}$$

$$R_{10}^{29_{t+4}} = d_{11_t}^2 + d_{01_{t+1}}^2 + d_{01_{t+2}}^2 + d_{00_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.29)}$$

$$R_{11}^{30_{t+4}} = d_{11_t}^2 + d_{01_{t+1}}^2 + d_{01_{t+2}}^2 + d_{00_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.30)}$$

$$R_{00}^{31_{t+4}} = d_{11_t}^2 + d_{01_{t+1}}^2 + d_{01_{t+2}}^2 + d_{11_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.31)}$$

$$R_{01}^{32_{t+4}} = d_{11_t}^2 + d_{01_{t+1}}^2 + d_{01_{t+2}}^2 + d_{11_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.32)}$$

A.2 State 01

$$R_{10}^{1_{t+4}} = d_{10_t}^2 + d_{00_{t+1}}^2 + d_{10_{t+2}}^2 + d_{00_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.33)}$$

$$R_{11}^{2_{t+4}} = d_{10_t}^2 + d_{00_{t+1}}^2 + d_{10_{t+2}}^2 + d_{00_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.34)}$$

$$R_{00}^{3_{t+4}} = d_{10_t}^2 + d_{00_{t+1}}^2 + d_{10_{t+2}}^2 + d_{11_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.35)}$$

$$R_{01}^{4_{t+4}} = d_{10_t}^2 + d_{00_{t+1}}^2 + d_{10_{t+2}}^2 + d_{11_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.36)}$$

$$R_{11}^{5_{t+4}} = d_{10_t}^2 + d_{00_{t+1}}^2 + d_{01_{t+2}}^2 + d_{10_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.37)}$$

$$R_{10}^{6_{t+4}} = d_{10_t}^2 + d_{00_{t+1}}^2 + d_{01_{t+2}}^2 + d_{10_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.38)}$$

$$R_{01}^{7_{t+4}} = d_{10_t}^2 + d_{00_{t+1}}^2 + d_{01_{t+2}}^2 + d_{01_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.39)}$$

$$R_{00}^{8_{t+4}} = d_{10_t}^2 + d_{00_{t+1}}^2 + d_{01_{t+2}}^2 + d_{01_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.40)}$$

$$R_{00}^{9_{t+4}} = d_{10_t}^2 + d_{11_{t+1}}^2 + d_{00_{t+2}}^2 + d_{00_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.41)}$$

$$R_{01}^{10_{t+4}} = d_{10_t}^2 + d_{11_{t+1}}^2 + d_{00_{t+2}}^2 + d_{00_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.42)}$$

$$R_{10}^{11_{t+4}} = d_{10_t}^2 + d_{11_{t+1}}^2 + d_{00_{t+2}}^2 + d_{11_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.43)}$$

$$R_{11}^{12_{t+4}} = d_{10_t}^2 + d_{11_{t+1}}^2 + d_{00_{t+2}}^2 + d_{11_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.44)}$$

$$R_{01}^{13_{t+4}} = d_{10_t}^2 + d_{11_{t+1}}^2 + d_{11_{t+2}}^2 + d_{10_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.45)}$$

$$R_{00}^{14_{t+4}} = d_{10_t}^2 + d_{11_{t+1}}^2 + d_{11_{t+2}}^2 + d_{10_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.46)}$$

$$R_{11}^{15_{t+4}} = d_{10_t}^2 + d_{11_{t+1}}^2 + d_{11_{t+2}}^2 + d_{01_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.47)}$$

$$R_{10}^{16_{t+4}} = d_{10_t}^2 + d_{11_{t+1}}^2 + d_{11_{t+2}}^2 + d_{01_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.48)}$$

$$R_{11}^{17_{t+4}} = d_{01_t}^2 + d_{10_{t+1}}^2 + d_{10_{t+2}}^2 + d_{10_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.49)}$$

$$R_{10}^{18_{t+4}} = d_{01_t}^2 + d_{10_{t+1}}^2 + d_{10_{t+2}}^2 + d_{10_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.50)}$$

$$R_{01}^{19_{t+4}} = d_{01_t}^2 + d_{10_{t+1}}^2 + d_{10_{t+2}}^2 + d_{01_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.51)}$$

$$R_{00}^{20_{t+4}} = d_{01_t}^2 + d_{10_{t+1}}^2 + d_{10_{t+2}}^2 + d_{01_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.52)}$$

$$R_{10}^{21_{t+4}} = d_{01_t}^2 + d_{10_{t+1}}^2 + d_{01_{t+2}}^2 + d_{00_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.53)}$$

$$R_{11}^{22_{t+4}} = d_{01_t}^2 + d_{10_{t+1}}^2 + d_{01_{t+2}}^2 + d_{00_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.54)}$$

$$R_{00}^{23_{t+4}} = d_{01_t}^2 + d_{10_{t+1}}^2 + d_{01_{t+2}}^2 + d_{11_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.55)}$$

$$R_{01}^{24_{t+4}} = d_{01_t}^2 + d_{10_{t+1}}^2 + d_{01_{t+2}}^2 + d_{11_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.56)}$$

$$R_{01}^{25_{t+4}} = d_{01_t}^2 + d_{01_{t+1}}^2 + d_{00_{t+2}}^2 + d_{10_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.57)}$$

$$R_{00}^{26_{t+4}} = d_{01_t}^2 + d_{01_{t+1}}^2 + d_{00_{t+2}}^2 + d_{10_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.58)}$$

$$R_{11}^{27_{t+4}} = d_{01_t}^2 + d_{01_{t+1}}^2 + d_{00_{t+2}}^2 + d_{01_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.59)}$$

$$R_{10}^{28_{t+4}} = d_{01_t}^2 + d_{01_{t+1}}^2 + d_{00_{t+2}}^2 + d_{01_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.60)}$$

$$R_{00}^{29_{t+4}} = d_{01_t}^2 + d_{01_{t+1}}^2 + d_{11_{t+2}}^2 + d_{00_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.61)}$$

$$R_{01}^{30_{t+4}} = d_{01_t}^2 + d_{01_{t+1}}^2 + d_{11_{t+2}}^2 + d_{00_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.62)}$$

$$R_{10}^{31_{t+4}} = d_{01_t}^2 + d_{01_{t+1}}^2 + d_{11_{t+2}}^2 + d_{11_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.63)}$$

$$R_{11}^{32_{t+4}} = d_{01_t}^2 + d_{01_{t+1}}^2 + d_{11_{t+2}}^2 + d_{11_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.64)}$$

A.3 State 10

$$R_{01}^{1_{t+4}} = d_{00_t}^2 + d_{10_{t+1}}^2 + d_{00_{t+2}}^2 + d_{10_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.65)}$$

$$R_{00}^{2_{t+4}} = d_{00_t}^2 + d_{10_{t+1}}^2 + d_{00_{t+2}}^2 + d_{10_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.66)}$$

$$R_{11}^{3_{t+4}} = d_{00_t}^2 + d_{10_{t+1}}^2 + d_{00_{t+2}}^2 + d_{01_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.67)}$$

$$R_{10}^{4_{t+4}} = d_{00_t}^2 + d_{10_{t+1}}^2 + d_{00_{t+2}}^2 + d_{01_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.68)}$$

$$R_{00}^{5_{t+4}} = d_{00_t}^2 + d_{10_{t+1}}^2 + d_{11_{t+2}}^2 + d_{00_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.69)}$$

$$R_{01}^{6_{t+4}} = d_{00_t}^2 + d_{10_{t+1}}^2 + d_{11_{t+2}}^2 + d_{00_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.70)}$$

$$R_{10}^{7_{t+4}} = d_{00_t}^2 + d_{10_{t+1}}^2 + d_{11_{t+2}}^2 + d_{11_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.71)}$$

$$R_{11}^{8_{t+4}} = d_{00_t}^2 + d_{10_{t+1}}^2 + d_{11_{t+2}}^2 + d_{11_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.72)}$$

$$R_{11}^{9_{t+4}} = d_{00_t}^2 + d_{01_{t+1}}^2 + d_{10_{t+2}}^2 + d_{10_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.73)}$$

$$R_{10}^{10_{t+4}} = d_{00_t}^2 + d_{01_{t+1}}^2 + d_{10_{t+2}}^2 + d_{10_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.74)}$$

$$R_{01}^{11_{t+4}} = d_{00_t}^2 + d_{01_{t+1}}^2 + d_{10_{t+2}}^2 + d_{01_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.75)}$$

$$R_{00}^{12_{t+4}} = d_{00_t}^2 + d_{01_{t+1}}^2 + d_{10_{t+2}}^2 + d_{01_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.76)}$$

$$R_{10}^{13_{t+4}} = d_{00_t}^2 + d_{01_{t+1}}^2 + d_{01_{t+2}}^2 + d_{00_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.77)}$$

$$R_{11}^{14_{t+4}} = d_{00_t}^2 + d_{01_{t+1}}^2 + d_{01_{t+2}}^2 + d_{00_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.78)}$$

$$R_{00}^{15_{t+4}} = d_{00_t}^2 + d_{01_{t+1}}^2 + d_{01_{t+2}}^2 + d_{11_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.79)}$$

$$R_{01}^{16_{t+4}} = d_{00_t}^2 + d_{01_{t+1}}^2 + d_{01_{t+2}}^2 + d_{11_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.80)}$$

$$R_{00}^{17_{t+4}} = d_{11_t}^2 + d_{00_{t+1}}^2 + d_{00_{t+2}}^2 + d_{00_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.81)}$$

$$R_{01}^{18_{t+4}} = d_{11_t}^2 + d_{00_{t+1}}^2 + d_{00_{t+2}}^2 + d_{00_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.82)}$$

$$R_{10}^{19_{t+4}} = d_{11_t}^2 + d_{00_{t+1}}^2 + d_{00_{t+2}}^2 + d_{11_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.83)}$$

$$R_{11}^{20_{t+4}} = d_{11_t}^2 + d_{00_{t+1}}^2 + d_{00_{t+2}}^2 + d_{11_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.84)}$$

$$R_{01}^{21_{t+4}} = d_{11_t}^2 + d_{00_{t+1}}^2 + d_{11_{t+2}}^2 + d_{10_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.85)}$$

$$R_{00}^{22_{t+4}} = d_{11_t}^2 + d_{00_{t+1}}^2 + d_{11_{t+2}}^2 + d_{10_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.86)}$$

$$R_{11}^{23_{t+4}} = d_{11_t}^2 + d_{00_{t+1}}^2 + d_{11_{t+2}}^2 + d_{01_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.87)}$$

$$R_{10}^{24_{t+4}} = d_{11_t}^2 + d_{00_{t+1}}^2 + d_{11_{t+2}}^2 + d_{01_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.88)}$$

$$R_{10}^{25_{t+4}} = d_{11_t}^2 + d_{11_{t+1}}^2 + d_{10_{t+2}}^2 + d_{00_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.89)}$$

$$R_{11}^{26_{t+4}} = d_{11_t}^2 + d_{11_{t+1}}^2 + d_{10_{t+2}}^2 + d_{00_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.90)}$$

$$R_{00}^{27_{t+4}} = d_{11_t}^2 + d_{11_{t+1}}^2 + d_{10_{t+2}}^2 + d_{11_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.91)}$$

$$R_{01}^{28_{t+4}} = d_{11_t}^2 + d_{11_{t+1}}^2 + d_{10_{t+2}}^2 + d_{11_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.92)}$$

$$R_{11}^{29_{t+4}} = d_{11_t}^2 + d_{11_{t+1}}^2 + d_{01_{t+2}}^2 + d_{10_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.93)}$$

$$R_{10}^{30_{t+4}} = d_{11_t}^2 + d_{11_{t+1}}^2 + d_{01_{t+2}}^2 + d_{10_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.94)}$$

$$R_{01}^{31_{t+4}} = d_{11_t}^2 + d_{11_{t+1}}^2 + d_{01_{t+2}}^2 + d_{01_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.95)}$$

$$R_{00}^{32_{t+4}} = d_{11_t}^2 + d_{11_{t+1}}^2 + d_{01_{t+2}}^2 + d_{01_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \quad \text{(A.96)}$$

A.4 State 11

$$R_{11}^{1_{t+4}} = d_{10_t}^2 + d_{10_{t+1}}^2 + d_{10_{t+2}}^2 + d_{10_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \quad \text{(A.97)}$$

$$R_{10}^{2_{t+4}} = d_{10_t}^2 + d_{10_{t+1}}^2 + d_{10_{t+2}}^2 + d_{10_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \quad \text{(A.98)}$$

$$R_{01}^{3_{t+4}} = d_{10_t}^2 + d_{10_{t+1}}^2 + d_{10_{t+2}}^2 + d_{01_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \quad \text{(A.99)}$$

$$R_{00}^{4_{t+4}} = d_{10_t}^2 + d_{10_{t+1}}^2 + d_{10_{t+2}}^2 + d_{01_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \text{ (A.100)}$$

$$R_{10}^{5_{t+4}} = d_{10_t}^2 + d_{10_{t+1}}^2 + d_{01_{t+2}}^2 + d_{00_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \text{ (A.101)}$$

$$R_{11}^{6_{t+4}} = d_{10_t}^2 + d_{10_{t+1}}^2 + d_{01_{t+2}}^2 + d_{00_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \text{ (A.102)}$$

$$R_{00}^{7_{t+4}} = d_{10_t}^2 + d_{10_{t+1}}^2 + d_{01_{t+2}}^2 + d_{11_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \text{ (A.103)}$$

$$R_{01}^{8_{t+4}} = d_{10_t}^2 + d_{10_{t+1}}^2 + d_{01_{t+2}}^2 + d_{11_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \text{ (A.104)}$$

$$R_{01}^{9_{t+4}} = d_{10_t}^2 + d_{01_{t+1}}^2 + d_{00_{t+2}}^2 + d_{10_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \text{ (A.105)}$$

$$R_{00}^{10_{t+4}} = d_{10_t}^2 + d_{01_{t+1}}^2 + d_{00_{t+2}}^2 + d_{10_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \text{ (A.106)}$$

$$R_{11}^{11_{t+4}} = d_{10_t}^2 + d_{01_{t+1}}^2 + d_{00_{t+2}}^2 + d_{01_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \text{ (A.107)}$$

$$R_{10}^{12_{t+4}} = d_{10_t}^2 + d_{01_{t+1}}^2 + d_{00_{t+2}}^2 + d_{01_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \text{ (A.108)}$$

$$R_{00}^{13_{t+4}} = d_{10_t}^2 + d_{01_{t+1}}^2 + d_{11_{t+2}}^2 + d_{00_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \text{ (A.109)}$$

$$R_{01}^{14_{t+4}} = d_{10_t}^2 + d_{01_{t+1}}^2 + d_{11_{t+2}}^2 + d_{00_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \text{ (A.110)}$$

$$R_{10}^{15_{t+4}} = d_{10_t}^2 + d_{01_{t+1}}^2 + d_{11_{t+2}}^2 + d_{11_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \text{ (A.111)}$$

$$R_{11}^{16_{t+4}} = d_{10_t}^2 + d_{01_{t+1}}^2 + d_{11_{t+2}}^2 + d_{11_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \text{ (A.112)}$$

$$R_{10}^{17_{t+4}} = d_{01_t}^2 + d_{00_{t+1}}^2 + d_{10_{t+2}}^2 + d_{00_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \text{ (A.113)}$$

$$R_{11}^{18_{t+4}} = d_{01_t}^2 + d_{00_{t+1}}^2 + d_{10_{t+2}}^2 + d_{00_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \text{ (A.114)}$$

$$R_{00}^{19_{t+4}} = d_{01_t}^2 + d_{00_{t+1}}^2 + d_{10_{t+2}}^2 + d_{11_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \text{ (A.115)}$$

$$R_{01}^{20_{t+4}} = d_{01_t}^2 + d_{00_{t+1}}^2 + d_{10_{t+2}}^2 + d_{11_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \text{ (A.116)}$$

$$R_{11}^{21_{t+4}} = d_{01_t}^2 + d_{00_{t+1}}^2 + d_{01_{t+2}}^2 + d_{10_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \text{ (A.117)}$$

$$R_{10}^{22_{t+4}} = d_{01_t}^2 + d_{00_{t+1}}^2 + d_{01_{t+2}}^2 + d_{10_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \text{ (A.118)}$$

$$R_{01}^{23_{t+4}} = d_{01_t}^2 + d_{00_{t+1}}^2 + d_{01_{t+2}}^2 + d_{01_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \text{ (A.119)}$$

$$R_{00}^{24_{t+4}} = d_{01_t}^2 + d_{00_{t+1}}^2 + d_{01_{t+2}}^2 + d_{01_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \text{ (A.120)}$$

$$R_{00}^{25_{t+4}} = d_{01_t}^2 + d_{11_{t+1}}^2 + d_{00_{t+2}}^2 + d_{00_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 00 \text{ (A.121)}$$

$$R_{01}^{26_{t+4}} = d_{01_t}^2 + d_{11_{t+1}}^2 + d_{00_{t+2}}^2 + d_{00_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 01 \text{ (A.122)}$$

$$R_{10}^{27_{t+4}} = d_{01_t}^2 + d_{11_{t+1}}^2 + d_{00_{t+2}}^2 + d_{11_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 10 \text{ (A.123)}$$

$$R_{11}^{28_{t+4}} = d_{01_t}^2 + d_{11_{t+1}}^2 + d_{00_{t+2}}^2 + d_{11_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 11 \text{ (A.124)}$$

$$R_{01}^{29_{t+4}} = d_{01_t}^2 + d_{11_{t+1}}^2 + d_{11_{t+2}}^2 + d_{10_{t+3}}^2 + d_{00_{t+4}}^2 \quad \textit{at state } 01 \text{ (A.125)}$$

$$R_{00}^{30_{t+4}} = d_{01_t}^2 + d_{11_{t+1}}^2 + d_{11_{t+2}}^2 + d_{10_{t+3}}^2 + d_{11_{t+4}}^2 \quad \textit{at state } 00 \text{ (A.126)}$$

$$R_{11}^{31_{t+4}} = d_{01_t}^2 + d_{11_{t+1}}^2 + d_{11_{t+2}}^2 + d_{01_{t+3}}^2 + d_{10_{t+4}}^2 \quad \textit{at state } 11 \text{ (A.127)}$$

$$R_{10}^{32_{t+4}} = d_{01_t}^2 + d_{11_{t+1}}^2 + d_{11_{t+2}}^2 + d_{01_{t+3}}^2 + d_{01_{t+4}}^2 \quad \textit{at state } 10 \text{ (A.128)}$$

A.5 Starting state 00

$$\begin{aligned} Eq.A.17 - Eq.6.18 &= Eq.A.23 - Eq.A.7 \qquad \text{(A.129)} \\ &= Eq.A.23 - Eq.A.7 \\ &= Eq.A.24 - Eq.A.8 \\ &= R_1 + r_1^{(1)} + r_1^{(3)} \end{aligned}$$

$$Eq.A.19 - Eq.A.3 = Eq.A.20 - Eq.A.4 \qquad \text{(A.130)}$$

$$= Eq.A.21 - Eq.A.5$$

$$= Eq.A.22 - Eq.A.6$$

$$= R_1 + r_1^{(1)} - r_1^{(3)}$$

$$Eq.A.25 - Eq.A.9 = Eq.A.26 - Eq.A.10 \tag{A.131}$$

$$= Eq.A.31 - A.15$$

$$= Eq.A.32 - Eq.A.16$$

$$= R_1 - r_1^{(1)} + r_1^{(3)}$$

$$Eq.A.27 - Eq.A.11 = Eq.A.28 - Eq.A.12 \tag{A.132}$$

$$= Eq.A.29 - Eq.A.13$$

$$= Eq.A.30 - Eq.A.14$$

$$= R_1 - r_1^{(1)} - r_1^{(3)}$$

Let:

$$R_1 = r_1^{(0)} + r_2^{(0)} \tag{A.133}$$

$$a_1 = \begin{cases} 0 & \text{if } R_1 + r_1^{(1)} + r_2^{(0)} \geq 0 \\ 1 & \text{otherwise} \end{cases}$$

$$a_2 = \begin{cases} 0 & \text{if } R_1 + r_1^{(1)} - r_2^{(0)} \geq 0 \\ 1 & \text{otherwise} \end{cases}$$

$$a_3 = \begin{cases} 0 & \text{if } R_1 - r_1^{(1)} + r_2^{(0)} \geq 0 \\ 1 & \text{otherwise} \end{cases}$$

$$a_4 = \begin{cases} 0 & \text{if } R_1 - r_1^{(1)} - r_2^{(0)} \geq 0 \\ 1 & \text{otherwise} \end{cases}$$

Let:

$a = (a_1, a2, a_3, a4)$

If $wt(a) = 4$ then the decoded symbol is 11.

If $wt(a) = 0$ then the decoded symbol is 00.

If $wt(a) = 1$ then we consider the following:

Let:

$$\beta_1 = \begin{cases} 0 & \text{if} R_1(-1)^{\alpha_3 \oplus \alpha_4} r_1^{(1)} + r_1^{(2)} + (-1)^{\alpha_3 \oplus \alpha_4} r_2^{(2)} \geq 0 \\ 1 & \text{otherwise} \end{cases}$$

$$\beta_2 = \begin{cases} 0 & \text{if } R_1(-1)^{\alpha_3 \oplus \alpha_4} r_1^{(1)} - r_1^{(2)} + (-1)^{\alpha_3 \oplus \alpha_4} r_2^{(2)} \geq 0 \\ 1 & \text{otherwise} \end{cases}$$

Let:

$\beta = (\beta_1, \beta_2)$

If $wt(\beta) = 0$ then the decoded symbol is 00

If $wt(\beta) = 1$ then:

Let:

$$\gamma = \begin{cases} 0 & \text{if } R_1(-1)^{\alpha_1 \oplus \alpha_2} r_2^{(1)} + (-1)^{\alpha_1 \beta_2 \oplus \beta_1 (\alpha_1 \oplus \alpha_2 \oplus \alpha_3)} r_1^{(2)} (-1)^{\beta_2 (\alpha_1 \oplus \alpha_2 + \alpha_3) \oplus \alpha_4 \beta_3} r_2^{(3)} \geq 0 \\ 1 & \text{otherwise} \end{cases}$$

If $\gamma = 0$ then the decoded symbol is 11

The above procedure can similarly be done [Arani*et al.* (1989)] for starting states 01, 10, and 11.