Error Control Techniques for the Compound Channel

ERROR CONTROL TECHNIQUES FOR THE COMPOUND CHANNEL

Electrical Theodore J. Dmuchalsky, B.Eng. (McGill) M.Eng.

ABSTRACT

This thesis deals with error control techniques for the compound channel, and particularly with two new burst correcting techniques called compound-concatenated systems and GSA codes. Compound-concatenated systems are constructed by concatenating a random error correcting code with a burst correcting code, thereby giving the burst correcting code a large degree of immunity from noisy guard spaces. GSA codes are modified burst-trapping codes with the unique property that their adaptive guard space requirement is consecutive and immediately adjacent to the burst.

These techniques and other, well-known burst correcting codes, interleaved block codes, diffuse codes, Gallager codes, and burst-trapping codes, are described in some detail and are compared with respect to their performance and their complexity of implementation. Also, insofar as they are important to the understanding of burst correcting methods, two random error correcting codes, linear binary codes (parity-check codes) and majority decodable convolutional codes, are described.

ERROR CONTROL TECHNIQUES FOR THE COMPOUND CHANNEL

by

Theodore J. Dmuchalsky, B.Eng. (McGill)

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

Department of Electrical Engineering,
McGill University,
Montreal, Quebec,
November, 1971.

ADDENDA

Several errors in the text have been brought to the attention of the author. Corrections and clarifications are as follows:

(a) On pages 4, 57, and 65, it is alleged that the diffuse codes of Chapter 7 were first described in 1968 by Kohlenberg and Forney (reference 22). In fact, the first published report on diffuse codes was J.L. Massey, "Advances in Threshold Decoding," in Advances in Communication Systems (Ed. A. Balakrishnan), Vol. III, Academic Press; 1968. Massey's work was available to Kohlenberg and Forney in manuscript form when their paper was written.

(b) On pages 5 and 93, it is inferred that the concept of concatenation, as applied to the compound-concatenated systems of Chapter 10, is due to Forney (reference 28). However, the coding scheme of Chapter 10 is more properly an example of the iterative coding introduced by Elias (reference 9).

(c) On page 15, since modulo-q arithmetic is specified, the number q must itself be a prime and not simply a positive integer power of a prime.

(d) On page 21, the minimum distance d of a code may be as large as n-k+1, not just n-k.

(e) On page 30, a feedback decoder is said to complement each of the J composite parity-checks in order to remove the effect of a detected error. It should be emphasized that this is accomplished by complementing one and only one syndrome bit in each composite parity-check.

(f) On page 73, the Gallager decoder of Figure 8.2.2 would not perform well in practice. This is because a burst may cause several incorrect decoding decisions by the random error corrector among the bits preceding the burst. These incorrect decisions can be cancelled by the burst corrector if several bits of buffering are provided at the storage register outputs. This more practical decoding scheme is described by Kohlenberg and Forney (reference 28).

T.J. Dmuchalsky
April 1972

ABSTRACT

This thesis deals with error control techniques for the compound channel, and particularly with two new burst correcting techniques called compound-concatenated systems and GSA codes. Compound-concatenated systems are constructed by concatenating a random error correcting code with a burst correcting code, thereby giving the burst correcting code a large degree of immunity from noisy guard spaces. GSA codes are modified burst-trapping codes with the unique property that their adaptive guard space requirement is consecutive and immediately adjacent to the burst.

These techniques and other, well-known burst correcting codes, interleaved block codes, diffuse codes, Gallager codes, and burst-trapping codes, are described in some detail and are compared with respect to their performance and their complexity of implementation. Also, insofar as they are important to the understanding of burst correcting methods, two random error correcting codes, linear binary codes (parity-check codes) and majority decodable convolutional codes, are described.

ACKNOWLEDGEMENTS

I should like to acknowledge my gratitude and my indebtedness to those persons who were so helpful to me in the preparation of this thesis. First, to Dr. Michael J. Ferguson, of the Department of Electrical Engineering, McGill University, who was my research director, for his guidance and for his patience. Also, to Brian Tabak and to David Lewis, graduate students in electrical engineering at McGill University, for many rewarding and enlightening discussions.

TABLE OF CONTENTS

1. INTRODUCTION

1.1. Purpose

An important problem in communications engineering today is the reliable transmission of digital information over a channel designed for analog signals. A common solution to the transmission aspect of the problem is to modulate the digital message in some suitable fashion and then to demodulate the received signal.

Since every real system is subject to errors at the receiver and demodulator, reliability may not easily be achieved. These errors may be statistically random or they may occur in clusters or bursts. Random errors generally result from the total effects of background thermal noise, non-linear frequency response of the channel, and frequency offset and phase jitter in the receiver. Depending upon the modulation scheme and the transmission rate, some of these error-inducing mechanisms play a much more significant role than others. Bursts of errors are caused by more catastrophic events, including serious environmental disturbances such as lightning and sun spot activity, impulse noise and crosstalk in switched channels, loss of synchronization between receiver and transmitter, or even temporary loss of the channel.

System reliability can often be interpreted as the frequency, or probability, of errors in attempting to recover the original digital message. If a minimum standard is not achieved at the demodulator, safeguards against errors must be incorporated into the system. One alternative is to encode the digital information before modulation in such a way that errors may be corrected after demodulation.

It is the purpose of this thesis to examine encoding and error correction techniques applicable to the compound channel, any channel or system subject to both random errors and bursts of errors. These techniques, called burst correcting codes, are discussed from three points of view: the structure of codes and methods for encoding and decoding; the complexity of implementation of the decoder; and the expected performance in error control in terms of the probability of a decoding error.

No background in coding theory or information theory is required of the reader, though he is expected to have some familiarity with probability theory, linear algebra, and a few electronic circuits such as shift registers. All other essential concepts are developed in the text as they are needed.

1.2. Outline and Historical Background

Chapters 2 to 9 in this thesis may be considered to be introductory or tutorial in nature since they largely cover the work of the authors listed in the bibliography. The original papers, however, were intended for advanced students of coding theory and were written in styles using notational and mathematical conventions most convenient to the individual authors. The introductory chapters of this thesis attempt to unify and simplify the descriptions of codes so that, with few exceptions, the approach in these chapters is quite different from that in the original papers. Except where noted, expressions for the complexity and performance of codes were derived independently in this thesis and many are original.

Chapters 10, 11, and 12 are, so far as is known, completely original in this thesis. The codes described in Chapters 10 and 11 were inspired by other known codes, but the ideas, descriptions, and derivations in these chapters are entirely the independent work of the author.

This thesis deals with coding which, quite simply, is a means of increasing the reliability of a digital communications system. Random error correcting codes are designed for the discrete memoryless channel (DMC), a channel producing only random errors, and burst correcting codes are designed for the compound channel. Chapter 2 develops the foundations for the descriptions of these codes. It presents a simplified model of the communication system [1] wherein only binary data is transmitted, the binary symmetric channel model of the DMC, and Gilbert's model [2] of the compound channel. In addition, it includes the fundamental mathematical properties of codes to be found in any text on coding theory, such as those by Berlekamp [3] and by Peterson [4].

Chapter 3 describes random error correcting block codes, particularly linear binary codes, and a decoding algorithm called minimum distance decoding. Shannon [5] originally showed that block codes could control errors with as much reliability as desired without sacrificing information rate, and a host of others, including Hamming [6], Reed [7], Muller [8], Elias [9], Slepian [10], Hocquenghem [11], Bose and Chaudhuri [12], [13], and Peterson [14], developed and systematized block coding. Berlekamp [3] and Peterson [4] give thorough treatments of block codes, while Gallager [1] and Massey [15] give simple descriptions of linear codes.

Chapter 4 describes a decoding algorithm called majority decoding and an implementation of that algorithm called feedback decoding. Then it describes random error correcting convolutional codes, particularly those which are feedback decodable. It was Elias [16] who first discovered convolutional codes, but they were brought into prominence with the development of sequential decoding by Wozencraft and Reiffen [17] and later of threshold (majority) decoding by Massey [15]. Massey's work forms the basis for this chapter.

Chapter 5 examines important concepts relevant to all burst correcting codes. These include definitions of a burst and its guard space, the definition of burst correcting capability and the development of bounds on this capability, and the differentiation between adaptive and non-adaptive codes. The definitions in this chapter are taken from Gallager [1], while the bounds on burst correcting capability were originally found by Wyner and Ash [18], Gallager [1], and Reiger [19].

Chapter 6 describes burst correcting interleaved block codes. These codes are the first example of modifying known random error correcting codes to make them suitable for the correction of long bursts. They are discussed by Gallager [1] and by Berlekamp [3].

Chapter 7 describes burst correcting convolutional codes known as diffuse codes. It was Hagelbarger [20] who first developed convolutional codes with burst correcting capability, and Massey [21], among others, refined these codes. Diffuse codes in particular were first reported by Kohlenberg and Forney [22] and were more fully developed by Tong [23] and by Ferguson [24].

Chapter 8 describes burst correcting Gallager codes. These codes are adaptive and are obtained by a simple extension of random error correcting convolutional codes. They were discovered by Gallager [1], who called them time-diversity codes, and were first reported by Kohlenberg and Forney [22]. They were later generalized by Sullivan [25].

Chapter 9 describes burst correcting codes known as burst-trapping codes. They are adaptive and are obtained from random error correcting block codes. They were developed by Tong [26] and were later generalized by Burton, Sullivan, and Tong [27].

Chapter 10 introduces a method of extending or generalizing all burst correcting codes such that their required guard spaces need not be error-free. The method is based on concatenated codes, discussed by Forney [28], and the extended codes are called compound-concatenated systems.

Chapter 11 introduces a new burst correcting technique called guard-space-adaptive burst-trapping codes. These codes are adaptive and have an adaptive guard space requirement immediately adjacent to the burst being corrected. Their principle is a modification of Tong's burst-trapping codes [26].

Chapter 12 compares the complexity and performance of the burst correcting codes of Chapters 6 to 11 on specific compound channels. Data for the performance curves was obtained by programming in Fortran the IBM 360/75 computer of the McGill University Computing Centre.

2. SYSTEM MODELS AND MATHEMATICAL BASICS

2.1. Introduction

Before considering specific random error correcting and burst correcting codes, it is necessary to define the environment to which they are applicable and to establish certain notational and mathematical foundations. This chapter is designed to meet this need.

Section 2.2. defines a simple model of a digital communication system. This model treats all transmitted data as a sequence of binary digits, commonly represented "0" and "1." The section also establishes the representation and certain characteristics of binary sequences. Sections 2.3. and 2.4. describe the two binary channel models which shall be used throughout this thesis, the binary symmetric channel [1] and the Gilbert channel [2]. Finally, Section 2.5. provides the basic mathematical background necessary to establish the properties of binary codes applicable to these channel models.

2.2. Model of a Digital Communication System

In a general digital communication system, one of L discrete values at an information source must be reliably transmitted to an information sink over some available channel. A simple system model in which $L = 2^k$, k an integer, and in which only binary data is transmitted, is given in Fig. 2.2.1. For simplicity, we restrict ourselves to this model in the remainder of the thesis.

During some arbitrary time interval from jT seconds to (j + 1)T seconds, a source encoder accepts one of the L source values and uniquely translates it into a message sequence of k binary digits, or bits. This sequence is denoted $\underline{m}^j$,

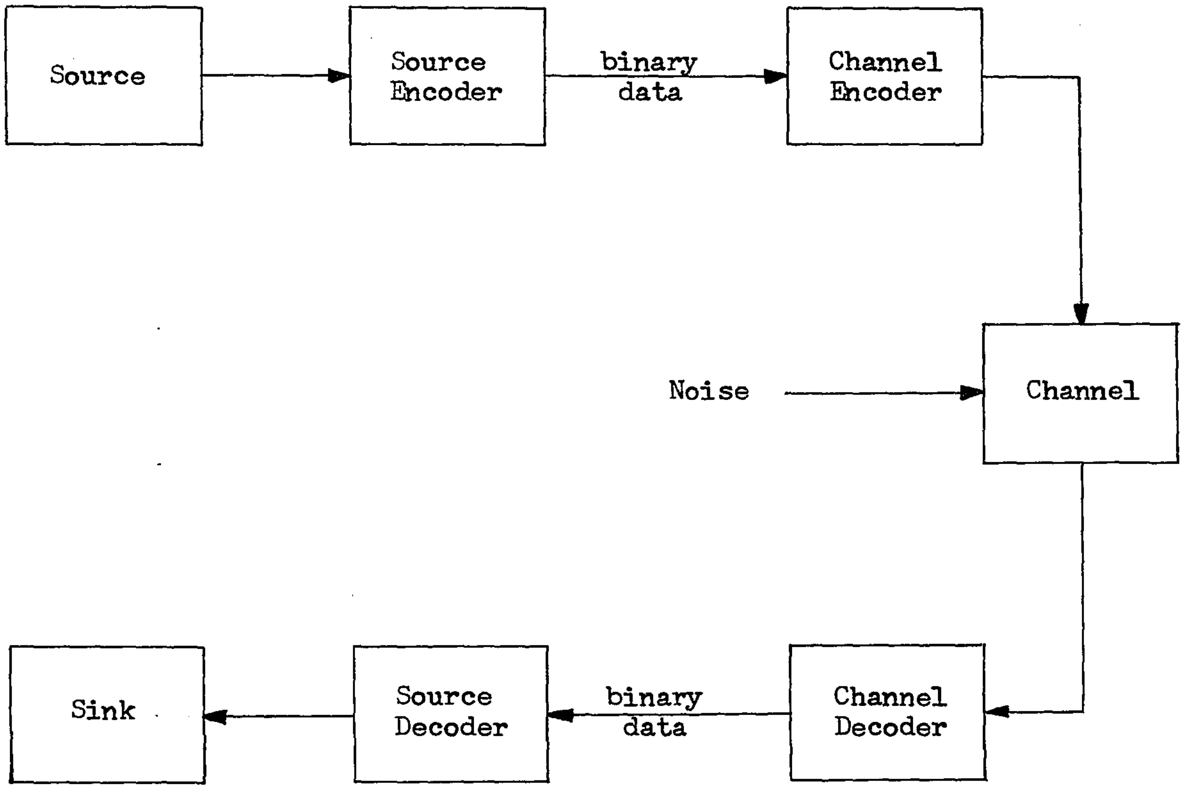


Figure 2.2.1 : Model of a Digital Communication System.

$$\underline{m}^j = (m_1^j, m_2^j, \ldots, m_k^j), \tag{2.2.1}$$

where j is simply an indexing right superscript. During the same interval, the channel encoder accepts the incoming stream of message bits and adds redundancy to them according to some fixed binary code. For each message sequence, the channel encoder transmits a channel sequence, $\underline{t}^j$, of n bits,

$$\underline{t}^j = (t_1^j, t_2^j, \ldots, t_n^j), \; n > k. \tag{2.2.2}$$

The binary code is said to have rate or efficiency R, where

$$R = {}^k/_n. \tag{2.2.3}$$

The transmission channel is considered to have a noise generating mechanism, modelled in Fig. 2.2.2, which corrupts the channel bits, t_m. According to the detailed error statistics of the channel, the noise source produces binary noise digits, e_m, which it adds to the corresponding channel bits. The channel decoder then receives binary digits r_m such that

$$r_m = t_m + e_m. \tag{2.2.4}$$

The addition above is modulo-2, where the mod-2 sum, or product, of any two binary digits is the remainder after division by 2 of the ordinary sum, or product, of the two digits. This remainder may only assume the values 0 and 1 and is thus itself a binary digit. Mod-2 addition gives the results:

$$0 + 0 = 0, \qquad 1 + 1 = 0,$$
$$0 + 1 = 1, \qquad 1 + 0 = 1. \tag{2.2.5}$$

Note that addition and subtraction are equivalent mod-2 operations. Henceforth, all arithmetic operations on binary digits will be understood to be mod-2 unless otherwise specified.

The noise bit e_m is called an error if $e_m = 1$, since in that case the channel bit t_m and the received bit r_m must differ. Since t_m is an element of some channel sequence $\underline{t}^j$, e_m may be considered to be an element of a corres-

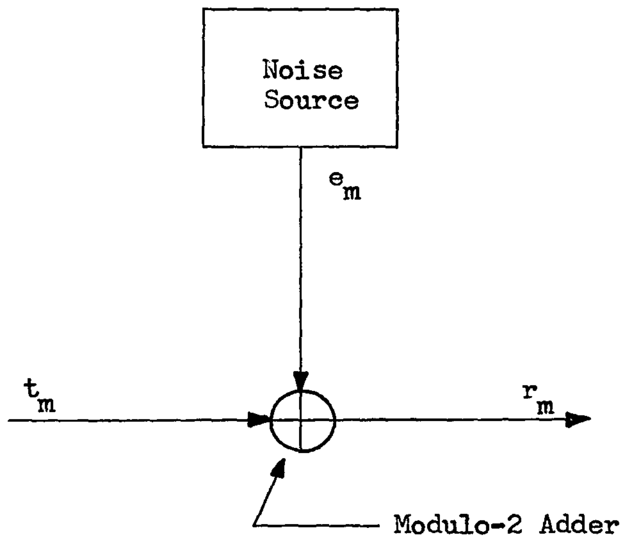


Figure 2.2.2 : Noise Generating Mechanism of the Channel.

ponding noise sequence, or error pattern, $\underline{e}^j$,

$$\underline{e}^j = (e_1^j, e_2^j, \ldots, e_n^j). \tag{2.2.6}$$

Similarly, r_m is an element of a received sequence $\underline{r}^j$,

$$\underline{r}^j = (r_1^j, r_2^j, \ldots, r_n^j), \tag{2.2.7}$$

and
$$\underline{r}^j = \underline{t}^j + \underline{e}^j = (t_1^j + e_1^j, \ldots, t_n^j + e_n^j). \tag{2.2.8}$$

Note from (2.2.8) that the mod-2 addition of equal length binary sequences is the mod-2 addition of the corresponding elements. Binary sequences have two other simple properties defined below.

Definition 2.2.1. A binary sequence of length n which contains w 1's and (n - w) 0's is said to have weight w.

Definition 2.2.2. If two binary sequences of length n differ in d of their corresponding elements, then the (Hamming) distance between them is d.

From (2.2.5), the mod-2 sum of like digits is 0 and of unlike digits is 1. Thus, in (2.2.8), if the distance between $\underline{t}^j$ and $\underline{e}^j$ is d, then $\underline{r}^j$ must have weight d.

The channel decoder in Fig. 2.2.1 accepts the incoming stream of received bits. Using the same code as the channel encoder, it attempts to recover the original message sequences $\underline{m}^j$, possibly successfully. The source decoder then assigns one of the L source values to each of the decoded messages and feeds this information to the sink.

2.3. The Binary Symmetric Channel

A digital transmission channel which produces only random errors is known as a discrete memoryless channel, or DMC. Gallager [1] describes many different DMC models, but the simplest and most widely applied model is the binary symmetric channel, or BSC, shown in Fig. 2.3.1.

Each digit in any DMC noise sequence is statistically independent of all other noise digits, and errors occur according to some fixed probability distribution. On the BSC in particular, noise digits are binary and errors occur with probability p_o. If $p_o < 0.5$, then lower weight noise sequences are more probable.

Henceforth, we shall always use the BSC model of a DMC.

2.4. The Gilbert Channel

The compound channel, or discrete channel with memory, or burst-noise channel, produces noise sequences in which errors may be either random and independent or clustered into bursts and generally not independent. Gallager [1] describes several models of the discrete channel with memory, including the very simple Gilbert channel [2].

The Gilbert channel model assumes that a noise sequence has the properties of a Markov chain [29]. The channel is assigned two states or modes of behaviour. In the "good" state, or random mode, errors occur with some low probability p_o, say 10^{-6}, and are independent. In the "bad" state, or burst mode, errors occur with much higher probability q_o, say 10^{-1}, and are also modelled as independent. The transition probabilities between states should be chosen such that the frequency and lengths of bursts are similar to experimentally obtained values on the real channel to be modelled.

Later, when considering coding techniques for the compound channel, we shall make use only of the property of the Gilbert channel model that errors in either channel mode are statistically independent.

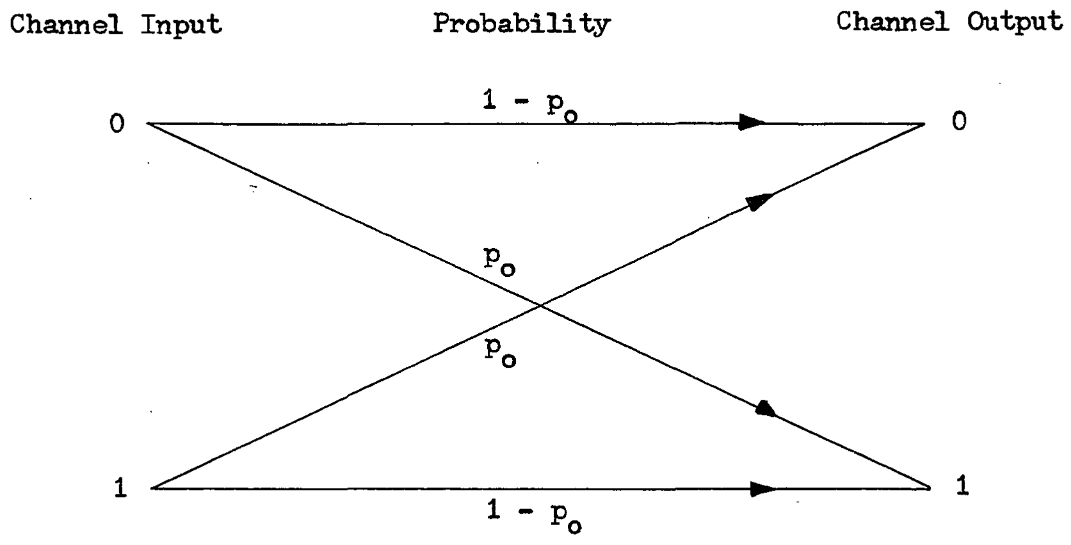


Figure 2.3.1 : The Binary Symmetric Channel.

2.5. Elements of Finite Algebra

Many aspects of coding theory are based upon the properties of groups and fields in finite algebra. Gallager [1], Berlekamp [3], and Peterson [4] provide a thorough introduction to this field. Here we give only those few definitions necessary to establish the fundamental properties of codes.

Definition 2.5.1. A group is a set of elements $\{g_1, g_2, g_3, \ldots\}$ under a rule of composition, denoted +, for which the following four axioms are satisfied:

(a) For any g_1, g_2 in the set, $g_1 + g_2$ is in the set.

(b) The associative law is satisfied; i.e., for any g_1, g_2, g_3 in the set,

$$(g_1 + g_2) + g_3 = g_1 + (g_2 + g_3).$$

(c) There is a unique identity element i in the set such that

$$g + i = i + g = g, \text{ for all } g \text{ in the set.}$$

(d) For each element g, there is a unique inverse element, -g, in the set such that

$$g + (-g) = (-g) + g = i.$$

Definition 2.5.2. An Abelian group is a group for which the commutative law is also satisfied:

$$g_1 + g_2 = g_2 + g_1, \text{ for all } g_1, g_2 \text{ in the set.}$$

Definition 2.5.3. A subgroup is a subset of elements of a group which itself forms a group under the same rule of composition.

Definition 2.5.4. For any subgroup $\{s_1, s_2, s_3, \ldots\}$ of an Abelian group and any fixed element g in the group, the subset of elements of the group given by

$$\{g + s_1, g + s_2, g + s_3, \ldots\}$$

is defined as a coset of the subgroup.

Definition 2.5.5. The order of a group or subgroup is the number of distinct elements in the group or subgroup.

A simple but important theorem [1] follows from the above five definitions.

Theorem 2.5.6. (a) If a subgroup is of finite order, each coset contains the same number of elements as the subgroup. (b) The coset containing the identity element is the subgroup itself. (c) No two cosets of the same subgroup may have any elements in common. (d) The order of a group, if finite, is a multiple of the order of each subgroup.

Definition 2.5.7. A field is a set of at least two elements $\{f_1, f_2, \ldots\}$ under two rules of composition, denoted addition (+) and multiplication ($\cdot$), for which the following four axioms are satisfied:

(a) For any f_1, f_2 in the set, $f_1 + f_2$ and $f_1 \cdot f_2$ are in the set.

(b) The set of elements is an Abelian group under addition.

(c) Where zero is the identity element of the group under addition, the set of nonzero elements is an Abelian group under multiplication.

(d) The distributive law is satisfied; i.e., for any f_1, f_2, f_3 in the set,

$$(f_1 + f_2) \cdot f_3 = (f_1 \cdot f_3) + (f_2 \cdot f_3).$$

Definition 2.5.8. A Galois field, denoted GF(q), is a field containing a finite number, q, of elements. q must be of the form

$$q = p^k,$$

where p is any prime number and k is any positive integer.

It is easy to show that the set of integers $\{0, 1, \ldots, q-1\}$ is a Galois field under modulo-q addition and modulo-q multiplication, where the mod-q sum, or product, of any two elements of the set is the remainder after ordinary division by q of the ordinary sum, or product, of the two numbers. This remainder must also be an element of the set. In particular, the set of binary digits $\{0, 1\}$ is the Galois field GF(2) under mod-2 addition and mod-2 multiplication.

3. BLOCK CODES

3.1. Introduction

Many burst correcting codes, which are designed for use on the compound channel, either employ directly or are logical extensions of random error correcting codes. This chapter is intended as an introduction to the general class of random error correcting codes known as block codes, particularly that sub-class known as parity-check codes. For block codes, the channel encoder treats each incoming binary message sequence $\underline{m}^j$ as a distinct unit, or block, or group. The encoder adds redundancy to $\underline{m}^j$ by forming mod-2 combinations of its elements and thereby produces a unique binary channel sequence, or codeword, $\underline{t}^j$. Since $\underline{t}^j$ has length n and $\underline{m}^j$ has length k, we refer to such a code as an (n,k) block code with block length n.

Section 3.2. defines an (n,k) parity-check code and describes some of its properties. Section 3.3. describes a decoding algorithm for parity-check codes, called minimum distance decoding, and develops expressions for the performance of these codes in correcting errors and in detecting errors.

3.2. Parity-Check Codes

A class of block codes known as linear codes is useful for digital data whose values are the elements of the arbitrary Galois field GF(q). Parity-check codes [1], [3], [4] are binary linear codes. This section describes the structure and properties of parity-check codes and is based largely on the work of Gallager [1].

Definition 3.2.1. The codeword digits of an (n,k) parity-check code are defined by the relation

$$t_h^j \equiv \sum_{i=1}^{k} m_i^j \, g_{ih} \, , \; h = 1, 2, \ldots, n,$$

where the summation is mod-2 and the binary set $\{g_{ih}\}$ is arbitrary but fixed independent of $\underline{m}^j$ and $\underline{t}^j$.

A parity-check code may appear in systematic form in which the first k codeword digits, called information digits or message digits, correspond exactly to the message sequence $\underline{m}^j$. The remaining n - k codeword digits, called parity digits or check digits, represent the redundancy added by the encoder to protect the message against errors.

Definition 3.2.2. The codeword digits of an (n,k) systematic parity-check code are defined by the relations

$$t_i^j = m_i^j \ , \ i = 1, 2, \ldots, k,$$

$$t_h^j = \sum_{i=1}^{k} m_i^j \, g_{ih} \ , \ h = k+1, \ldots, n,$$

where the binary set $\{g_{ih}\}$ is arbitrary but fixed independent of $\underline{m}^j$ and $\underline{t}^j$.

The set $\{g_{ih}\}$ may be considered to be the elements of a k-by-n matrix G shown in Fig. 3.2.1 and known as the generator matrix. Treating $\underline{m}^j$ and $\underline{t}^j$ strictly as row vectors, we may write

$$\underline{t}^j = \underline{m}^j \, G. \tag{3.2.1}$$

Mod-2 matrix operations are the same as over the field of real numbers except that element-by-element arithmetic is mod-2.

Systematic parity-check codes are characterized by a parity-check matrix H which may be derived from the generator matrix G. From Definition 3.2.2.,

$$\sum_{i=1}^{k} t_i^j \, g_{ih} + t_h^j = 0 \ , \ h = k+1, \ldots, n. \tag{3.2.2}$$

The n - k equations in (3.2.2) may be expressed in matrix form as

$$G = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1,n} \\ g_{21} & g_{22} & \cdots & g_{2,n} \\ \vdots & \vdots & & \vdots \\ g_{k,1} & g_{k,2} & & g_{k,n} \end{bmatrix}$$

(a) Arbitrary (n,k) Parity-Check Code.

$$G = \begin{bmatrix} 1 & 0 & \cdots & 0 & g_{1,k+1} & \cdots & g_{1,n} \\ 0 & 1 & \cdots & 0 & g_{2,k+1} & \cdots & g_{2,n} \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 1 & g_{k,k+1} & \cdots & g_{k,n} \end{bmatrix}$$

(b) (n,k) Systematic Parity-Check Code.

Figure 3.2.1 : Generator Matrix of a Parity-Check Code.

$$\underline{t}^j \, H = \underline{0}, \tag{3.2.3}$$

where H is the n-by-(n - k) parity-check matrix of the code, Fig. 3.2.2, and $\underline{0}$ is a null row vector of dimension n - k. It is important to note that (3.2.3) is true if and only if the vector $\underline{t}^j$ is a valid codeword of the parity-check code.

A fundamental property of parity-check codes is that the set of codewords forms an Abelian group under mod-2 addition. This follows from the following observations:

(a) For any integer K, it is easily shown that the set of all 2^K binary sequences of length K forms an Abelian group under mod-2 addition. The all-0 sequence is the identity element of the group and each sequence is its own unique inverse.

(b) It follows that the set $\{\underline{m}^j\}$ of all 2^k message sequences forms an Abelian group under mod-2 addition. Thus, the sum of any two message sequences, $\underline{m}^1$ and $\underline{m}^2$, is also a message sequence. If $\underline{t}^1$ and $\underline{t}^2$ are the codewords corresponding to $\underline{m}^1$ and $\underline{m}^2$, then their sum,

$$\underline{t}^1 + \underline{t}^2 = \underline{m}^1 \, G + \underline{m}^2 \, G = (\underline{m}^1 + \underline{m}^2)G, \tag{3.2.4}$$

is also a codeword, and it is easily shown that the set $\{\underline{t}^j\}$ of all 2^k codewords forms an Abelian group under mod-2 addition. The all-0 codeword is the identity element and each codeword is its own unique inverse.

(c) The set of 2^k codewords is a subset of the set of all 2^n binary sequences of length n. Thus, the group formed by the subset of codewords is a subgroup of the group formed by the set of 2^n binary sequences. From Theorem 2.5.6., there are 2^{n-k} cosets of the subgroup, including the subgroup itself, and every element of every coset is distinct.

$$H = \begin{bmatrix} g_{1,k+1} & g_{1,k+2} & \cdots & g_{1,n} \\ g_{2,k+1} & g_{2,k+2} & \cdots & g_{2,n} \\ \vdots & \vdots & & \vdots \\ g_{k,k+1} & g_{k,k+2} & \cdots & g_{k,n} \\ 1 & 0 & \ldots & 0 \\ 0 & 1 & \ldots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \ldots & 1 \end{bmatrix}$$

Figure 3.2.2 : Parity-Check Matrix of a Parity-Check Code.

3.3. Minimum Distance Decoding

There are many different schemes for encoding and decoding the various sub-classes of (n,k) parity-check codes [1], [3], [4]. A description of any or all of these schemes is beyond the scope of this thesis, though we shall describe the structure and performance of a rather general decoding algorithm called minimum distance decoding.

3.3.a. Structure

Because each of the 2^k codewords of a parity-check code is distinct, there exists some nonzero distance between any two codewords in the set. In particular, if d is the minimum distance between any two codewords, then d is said to be the minimum distance of the code. For a systematic code, codewords must differ in at least one information digit, so d may be as low as one or as high as n - k, depending upon the choices of k, n, and G.

Consider that some codeword $\underline{t}^j$ from a set with minimum distance d is transmitted over a BSC with $p_o < 0.5$. The channel adds a noise sequence $\underline{e}^j$, which occurs with higher probability if it has lower weight, and from (2.2.8), the decoder receives the sequence $\underline{r}^j$,

$$\underline{r}^j = \underline{t}^j + \underline{e}^j, \tag{3.3.1}$$

where both $\underline{e}^j$ and $\underline{r}^j$ are elements of the set of all 2^n binary sequences of length n. A minimum distance decoder attempts to deduce the most probable value of $\underline{e}^j$ and thereby recover $\underline{t}^j$,

$$\underline{t}^j = \underline{e}^j + \underline{r}^j. \tag{3.3.2}$$

To do this, the decoder first calculates in some manner a syndrome $\underline{S}^j$, where

$$\underline{S}^j = \underline{r}^j H \tag{3.3.3}$$

and $$\underline{S}^j = (\underline{t}^j + \underline{e}^j)H = \underline{t}^j\,H + \underline{e}^j\,H = \underline{e}^j\,H. \qquad (3.3.4)$$

The syndrome, a binary sequence of dimension n - k, depends only upon the noise sequence $\underline{e}^j$.

Assume that $\underline{e}^j$ is an element of the set of codewords, which includes the zero-weight sequence. Then, from (3.2.3), $\underline{S}^j = \underline{0}$. In this case, the decoder knows that $\underline{e}^j$ is itself a valid codeword, and the most probable value of $\underline{e}^j$ is that value with lowest weight; i.e., the all-0 sequence. The decoder therefore deduces that $\underline{t}^j = \underline{r}^j$. If $\underline{e}^j$ is not an element of the set of codewords, then $\underline{S}^j \not\equiv \underline{0}$ and the decoder knows that $\underline{e}^j \not\equiv \underline{0}$. Let us consider the manner in which the decoder chooses the most probable value of $\underline{e}^j$ in this case.

Since $\underline{S}^j$ has dimension n - k, there are 2^{n-k} different syndromes. There are also 2^{n-k} different cosets of the subgroup of codewords and $\underline{e}^j$ must be an element of one and only one of these cosets. It is easily shown that each coset uniquely generates one of the 2^{n-k} syndromes according to (3.3.4). The following observations may be made:

(a) If d is the minimum distance of the subgroup of codewords, then d must be the minimum distance of every coset of the subgroup.

(b) Every noise sequence with equal weight is equally probable, since a sequence with weight w occurs with probability P_w,

$$P_w = \binom{n}{w}\, p_o^{\,w}\,(1 - p_o)^{n-w}, \qquad (3.3.5)$$

where $$\binom{n}{w} = \frac{n!}{w!\,(n-w)!}. \qquad (3.3.6)$$

(c) Every coset must have at least one lowest weight sequence as an element, corresponding to a most probable noise sequence. Choose one such sequence as coset leader, the specific choice being immaterial since all lowest weight sequences are equiprobable.

(d) The coset leader of the subgroup is the null sequence.

(e) Every sequence with weight less than $^d/_2$ must be a coset leader.

It follows from the above observations that the decoder deduces the most probable value of $\underline{e}^j$ if it decides that $\underline{e}^j$ is the coset leader of the coset which generates the syndrome $\underline{S}^j$. In so doing, it chooses as $\underline{t}^j$ that codeword which is closest in distance to the received sequence $\underline{r}^j$ and it decodes with minimum probability of decoding error.

The minimum distance decoder always decodes correctly when the noise sequence has weight less than $^d/_2$. It also decodes correctly for those noise sequences with weight $^d/_2$ or greater which happen to be coset leaders, if any. We say that a code has error correcting capability t if it corrects a class of error patterns which includes all noise sequences with weight at most t and, possibly, some sequences with higher weight. Thus, a minimum distance decoded parity-check code has error correcting capability given by

$$t \leq \left\lfloor \frac{d-1}{2} \right\rfloor, \qquad (3.3.7)$$

where we use the notation $\lfloor x \rfloor$ to denote the greatest integer less than or equal to x, and we use $\lceil x \rceil$ to denote the least integer greater than or equal to x.

The inequality exists in (3.3.7) because in many applications it is useful to employ an error correcting capability which is less than maximum. This preserves a portion of the minimum distance structure of the code for the detection of errors. For example, we saw that if a noise sequence occurs such that $\underline{r}^j$ is distance t or less from any codeword in the set, the minimum distance decoder chooses that particular codeword as having been transmitted. Conversely, if the noise sequence is such that $\underline{r}^j$ is distance greater than t from every codeword in the set, the decoder detects an uncorrectable error pattern.

3.3.b. Performance in Error Correction

The criterion employed as a measure of the performance of a code in correcting errors is the probability of decoding error, P(E). For a minimum distance decoded parity-check code, all error patterns with weight at most t are correctable. Also, if t is a maximum, some error patterns with weight greater than t may be correctable, a quality referred to as robustness. Since for the general case the degree of robustness of the code is undefined, the probability of a decoding error is upper-bounded by the probability that the noise sequence has weight at least t + 1. Thus, on the BSC,

$$P(E) \leq 1 - \sum_{j=0}^{t} \binom{n}{j} p_o^{\,j} (1 - p_o)^{n-j}. \qquad (3.3.8)$$

P(E) may be decreased by increasing t, decreasing n, or decreasing p_o. However, t and n are not independent parameters and p_o is fixed for any particular channel.

3.3.c. Performance in Error Detection

The criterion employed as a measure of the performance of a code in detecting errors is the probability of failure, P(F); i.e., the probability that an error pattern is neither detected nor successfully corrected. We shall develop an expression for P(F) based on the work of Tong [26].

If the parity-check code has error correcting capability $t < \left\lfloor {}^{d-1}/_{2} \right\rfloor$, then N_p different error patterns can be successfully corrected,

$$N_p = \sum_{j=0}^{t} \binom{n}{j}. \qquad (3.3.9)$$

Each of these error patterns must be a coset leader, so that the probability that any coset chosen at random has a correctable error pattern as its leader is P_N,

$$P_N = N_p/2^{n-k} = 2^{k-n} \sum_{j=0}^{t} \binom{n}{j}. \qquad (3.3.10)$$

If one of the $2^n - N_p$ uncorrectable error patterns occurs, and if it is an element of a coset whose leader is not correctable, then an error detection occurs. On the other hand, if it is an element of a coset whose leader is among the N_p correctable patterns, then a failure occurs; i.e., the decoder mistakenly attempts a correction, thereby committing a decoding error.

Since a failure may occur only if $\underline{r}^j$ is within distance t of the wrong codeword, the error pattern must have weight at least d - t, an event which occurs with probability P_d,

$$P_d = 1 - \sum_{j=0}^{d-t-1} \binom{n}{j} p_o^{\,j} (1 - p_o)^{n-j}. \qquad (3.3.11)$$

Thus, P(F) is approximately given by

$$P(F) \approx P_N P_d. \qquad (3.3.12)$$

P(F) may be decreased by decreasing t, increasing d, and increasing n - k.

4. CONVOLUTIONAL CODES

4.1. Introduction

In this chapter we shall study another general class of random error correcting codes known as convolutional codes. Like block codes, the information digits in the codeword $\underline{t}^j$ of a convolutional code correspond to the digits of a message sequence $\underline{m}^j$. Unlike block codes, however, the parity digits of $\underline{t}^j$ are linear mod-2 combinations of the elements of $\underline{m}^j$ and of the preceding u message sequences, $\underline{m}^{j-1}, \ldots, \underline{m}^{j-u}$. Because of their application to burst correcting codes for the compound channel, we shall be particularly interested in convolutional codes which are feedback decodable, where feedback decoding is a method of implementing a more general algorithm called majority decoding.

Section 4.2. develops the majority decoding algorithm and discusses the difficulties encountered in applying it to both block codes and convolutional codes. Section 4.3. defines the structure of convolutional codes and then, through an example, describes how a convolutional encoder and a feedback decoder are implemented, defines the complexity of implementation, and defines optimality criteria for convolutional codes. The section ends by developing useful expressions for the performance of feedback decoded convolutional codes in correcting errors and in detecting errors. Section 4.4. is a note on the comparative complexities of implementation for convolutional codes and block codes.

4.2. Majority Decoding

Massey [15] introduced two useful and simple decoding algorithms, applicable to both block codes and convolutional codes, known as majority decoding and "a posteriori" probability (APP) decoding. Both algorithms are also known by the generic term threshold decoding. Majority decoding is most commonly implemented with a feedback loop in the decoder, its purpose being primarily to remove the effects of previously decoded errors, and in this form it is known as feedback decoding. Robinson [30] has proposed a majority decoding scheme without feedback which is called definite decoding. In this section, we shall develop the majority decoding rule and describe the major difficulties encountered in its implementation.

From (3.3.3) and (3.3.4), the syndrome obtained at a decoder is given by the matrix relations

$$\underline{S}^j = \underline{r}^j H, \tag{4.2.1}$$

$$\underline{S}^j = (S^j_{k+1}, S^j_{k+2}, \ldots, S^j_n) = (\underline{t}^j + \underline{e}^j)H = \underline{e}^j H. \tag{4.2.2}$$

H is the parity-check matrix of the code, derived from (3.2.2),

$$\sum_{i=1}^{k} t^j_i g_{ih} + t^j_h = 0 \ , \ h = k+1, \ldots, n. \tag{4.2.3}$$

It follows that the syndrome bits may be expressed as

$$S^j_h = \sum_{i=1}^{k} (t^j_i + e^j_i)g_{ih} + (t^j_h + e^j_h) \ , \ h = k+1, \ldots, n, \tag{4.2.4}$$

or, equivalently, as

$$S^j_h = \sum_{i=1}^{k} e^j_i g_{ih} + e^j_h \ , \ h = k+1, \ldots, n. \tag{4.2.5}$$

The n - k equations above are known as parity-check equations.

Definition 4.2.1. Massey [15] defines a composite parity-check, A_m, as a linear mod-2 combination of syndrome bits,

$$A_m = \sum_{h=k+1}^{n} a_{mh} S^j_h,$$

where the set of coefficients $\{a_{mh}\}$ are arbitrary binary elements.

It follows from (4.2.5) that a composite parity-check is a linear mod-2 combination of noise digits,

$$A_m = \sum_{i=1}^{k} \sum_{h=k+1}^{n} a_{mh}\, g_{ih}\, e_i^j + \sum_{h=k+1}^{n} a_{mh}\, e_h^j. \qquad (4.2.6)$$

We say that a noise digit, e_p^j, is checked by a composite parity-check, A_m, if and only if e_p^j appears in the equation for that composite parity-check with a nonzero coefficient.

Definition 4.2.2. A set of J composite parity-checks, $\{A_m: m = 1, 2, \ldots, J\}$, is said to be orthogonal on the noise digit e_p^j if e_p^j is checked by every equation in the set and every other noise digit is checked by at most one equation in the set.

Massey [15] gives an important theorem from which the majority decoding rule can be derived.

Theorem 4.2.3. If there are at most $\lfloor J/2 \rfloor$ nonzero noise digits in the set $\{e_h^j\}$ checked by a set of J composite parity-checks, $\{A_m\}$, orthogonal on e_p^j, then $e_p^j = 1$ if more than $\lceil J/2 \rceil$ of the A_m have value 1 and $e_p^j = 0$ if at least $\lceil J/2 \rceil$ of the A_m have value 0.

Proof of Theorem 4.2.3. Suppose that all of the e_h^j in the set $\{e_h^j\}$ are zero with the possible exception of e_p^j. Thus, $A_m = e_p^j$, for all A_m in the set $\{A_m\}$. If $e_p^j = 0$, then at most $\lfloor J/2 \rfloor$ of the other elements of $\{e_h^j\}$ can be nonzero and at most $\lfloor J/2 \rfloor$ of the A_m can have value 1. Thus, at least $\lceil J/2 \rceil$ of the A_m still have value 0. If $e_p^j = 1$, then at most $\lfloor J/2 \rfloor - 1$ of the other elements of $\{e_h^j\}$ can be nonzero and at most $\lfloor J/2 \rfloor - 1$ of the A_m can have value 0. Thus, more than $\lceil J/2 \rceil$ of the A_m still have value 1. Therefore, the theorem always gives the value of e_p^j correctly.

The majority decoding rule may be stated as follows. Given a set of J composite parity-checks, $\{A_m\}$, orthogonal on the noise digit e_p^j, choose $e_p^j = 1$ if and only if at least $\lceil J/2 \rceil + 1$ of the A_m have value 1.

It is useful to consider the class of error patterns which are correctly decoded by the majority decoding rule. The members of this class will be described by the following two corollaries of Theorem 4.2.3., the first of which is a theorem given by Tong [23].

Corollary 4.2.4. If $e_p^j = 1$, then at least $\lceil J/2 \rceil + 1$ of the A_m will have value 1 if no more than $\lfloor J/2 \rfloor - 1$ of the A_m check other nonzero noise digits. If $e_p^j = 0$, then at least $\lceil J/2 \rceil$ of the A_m will have value 0 if no more than $\lfloor J/2 \rfloor$ of the A_m check any nonzero noise digits.

This corollary differs from Massey's theorem in that it does not limit the number of nonzero noise digits to $\lfloor J/2 \rfloor$. Rather, it limits the number of composite parity-checks which may check an unspecified number of errors. The theorem may be further generalized as follows.

Corollary 4.2.5. If $e_p^j = 1$, then at least $\lceil J/2 \rceil + 1$ of the A_m will have value 1 if no more than $\lfloor J/2 \rfloor - 1$ of the A_m check an even number of nonzero noise digits. If $e_p^j = 0$, then at least $\lceil J/2 \rceil$ of the A_m will have value 0 if no more than $\lfloor J/2 \rfloor$ of the A_m check an odd number of nonzero noise digits.

A code which is majority decoded has error correcting capability given by

$$t = \lfloor J/2 \rfloor \tag{4.2.7}$$

since the class of error patterns which is correctly decoded includes every error pattern with weight $\lfloor J/2 \rfloor$ or less. However, from Corollary 4.2.5., a large number of error patterns with weight greater than t are also correctly decoded.

A decoder using the majority decoding rule recovers each codeword digit t_p^j by adding the decoded value of e_p^j to the corresponding received bit r_p^j,

$$t_p^j = e_p^j + r_p^j. \tag{4.2.8}$$

If the implementation is as a feedback decoder, and if $e_p^j = 1$, then the decoder complements each of the J composite parity-checks which check e_p^j in order to remove the effect of the error. However, the use of feedback in the decoder leads to the possibility of error propagation. This may occur when the logic, or threshold, element of the decoder incorrectly decides that $e_p^j = 1$. Since the composite parity-checks are complemented, new errors are artificially introduced into the system, and these in turn may be responsible for further incorrect decisions at the logic element. With certain convolutional codes, such self-generating error propagation might continue as long as there is continuous data transmission. Other convolutional codes, known as self-orthogonal codes, have limited error propagation properties, and for block codes, error propagation cannot continue beyond the limits of the block being decoded.

Definite decoding [30] does not use feedback and therefore does not complement composite parity-checks. Although this avoids error propagation, the set of composite parity-checks checks more noise digits than with feedback decoding, and previous errors, even though they have been decoded, may affect the correction of several subsequent noise digits. Sullivan [31] shows intuitively and experimentally that feedback decoding generally results in fewer decoding errors than definite decoding in spite of error propagation.

Another difficulty with majority decoding is to find a set of orthogonal composite parity-checks. There exists a class of convolutional codes which can be easily orthogonalized in one step. Another class of convolutional codes cannot be orthogonalized at all and must be decoded by a complex algorithm called sequential decoding [17]. There are at least three classes of block codes which may be orthogonalized in one step: the Reed-Muller codes [3], [4], [15]; the self-orthogonal quasi-cyclic codes discovered by Townsend and Weldon [32]; and the difference-set cyclic codes discovered by Rudolph [33] and by Weldon [34]. Many other block codes may be orthogonalized in L steps, $L > 1$, and Massey [15] describes procedures for L-step orthogonalization. Rudolph [35] and Gore [36], [37], by employing what may be called "non-orthogonal composite parity-checks," have shown that all binary block codes are majority decodable.

4.3. Feedback Decodable Convolutional Codes

In this section, we shall consider that sub-class of convolutional codes which is feedback decodable. We shall describe the structure of these codes, their optimality criteria, their construction and implementation, and their performance in error correction and in error detection. Because of their relative simplicity, we shall consider mainly rate $\frac{1}{2}$ codes. Our approach is based on the work of Massey [15].

4.3.a. Code Structure

For each message sequence $\underline{m}^j$, a channel sequence $\underline{t}^j$ is produced at the encoder. Using the delay operator D, where D^m corresponds to a delay of mT seconds, these sequences may be represented as sets of polynomials,

$$M_i(D) = m_i^0 + m_i^1 D + m_i^2 D^2 + \dots, \quad i = 1, 2, \dots, k, \qquad (4.3.1)$$

$$T_i(D) = t_i^0 + t_i^1 D + t_i^2 D^2 + \dots, \quad i = 1, 2, \dots, n. \qquad (4.3.2)$$

Each polynomial is the sum of message or channel bits which appear in the same location of succeeding sequences.

If the convolutional code is in systematic form, then the first k bits in each channel sequence are information bits. Thus,

$$T_i(D) = M_i(D) \ , \ i = 1, 2, \ldots, k. \tag{4.3.3}$$

The n - k parity bits in each channel sequence are defined by a linear combination of the message sequences,

$$T_i(D) = G_i(D)\, M_1(D) + H_i(D)\, M_2(D) + \ldots + Z_i(D)\, M_k(D),$$
$$i = k + 1, \ldots, n. \tag{4.3.4}$$

The k(n - k) polynomials $G_i(D)$, $H_i(D)$, ..., $Z_i(D)$, $i = k + 1, \ldots, n$, are known as code-generating polynomials and play a role analogous to the generator matrix of an (n,k) parity-check code. These polynomials are of maximum degree u and are of the form

$$G_i(D) = g_i^0 + g_i^1\, D + \ldots + g_i^u\, D^u, \tag{4.3.5}$$

where the set of coefficients $\{g_i^j\}$ are binary elements.

A convolutional code has a characteristic matrix, which we denote as P^c, analogous to the parity-check matrix H of a systematic parity-check code. To find P^c, we have from (4.3.3) and (4.3.4) that

$$[G_i(D)\, T_1(D) + \ldots + Z_i(D)\, T_k(D)] - T_i(D) = 0 \ , \ i = k + 1, \ldots, n. \tag{4.3.6}$$

We may express the channel noise sequences $\underline{e}^j$, the received sequences $\underline{r}^j$, and the syndromes $\underline{S}^j$ by the sets of polynomials $E_i(D)$, $R_i(D)$, and $S_i(D)$ respectively, where

$$E_i(D) = e_i^0 + e_i^1\, D + e_i^2\, D^2 + \ldots, \ i = 1, 2, \ldots, n, \tag{4.3.7}$$

$$R_i(D) = T_i(D) + E_i(D)$$
$$= (t_i^0 + e_i^0) + (t_i^1 + e_i^1)\, D + \ldots$$

$$R_i(D) = r_i^0 + r_i^1 D + r_i^2 D^2 + \ldots,\ i = 1, 2, \ldots, n, \qquad (4.3.8)$$

$$\begin{aligned} S_i(D) &= [G_i(D)\, R_1(D) + \ldots + Z_i(D)\, R_k(D)] - R_i(D) \\ &= [G_i(D)\, E_1(D) + \ldots + Z_i(D)\, E_k(D)] - E_i(D) \\ &= S_i^0 + S_i^1 D + S_i^2 D^2 + \ldots,\ i = k+1, \ldots, n. \end{aligned} \qquad (4.3.9)$$

Expanding the polynomials, whose coefficients are binary elements, we obtain

$$\begin{aligned} S_i^j = {} & [g_i^j e_1^0 + g_i^{j-1} e_1^1 + \ldots + g_i^0 e_1^j] + \ldots \\ & + [z_i^j e_k^0 + z_i^{j-1} e_k^1 + \ldots + z_i^0 e_k^j] + e_i^j, \\ & i = k+1, \ldots, n,\quad j = 0, 1, \ldots, u. \end{aligned} \qquad (4.3.10)$$

The set of equations above are the parity-check equations of the convolutional code, and they may be expressed in matrix form as

$$\underline{S}^c = \underline{e}^c P^c = \underline{e}^c \begin{bmatrix} H^c \\ I^c \end{bmatrix}, \qquad (4.3.11)$$

where $\underline{S}^c$ and $\underline{e}^c$ are row vectors of dimension $(n - k)(u + 1)$ and $n(u + 1)$ respectively,

$$\underline{S}^c = (S_{k+1}^0, \ldots, S_{k+1}^u, \ldots, S_n^0, \ldots, S_n^u), \qquad (4.3.12)$$

$$\underline{e}^c = (e_1^0, \ldots, e_1^u, \ldots, e_n^0, \ldots, e_n^u), \qquad (4.3.13)$$

I^c is the identity matrix of dimension $(n - k)(u + 1)$, and H^c is the $k(u + 1)$-by-$(n - k)(u + 1)$ matrix in Fig. 4.3.1. Each upper-triangular submatrix of H^c is known as a parity triangle, one for each code-generating polynomial.

4.3.b. Optimality and Implementation

In order to develop optimality criteria for a convolutional code and to show how the encoder and feedback decoder are implemented, it is convenient to use a numerical example. To maintain simplicity, we choose

$$H^c = \left[\begin{array}{ccccccccc}
g^0_{k+1} & g^1_{k+1} & g^2_{k+1} & \cdots & g^u_{k+1} & & g^0_n & \cdots & g^u_n \\
0 & g^0_{k+1} & g^1_{k+1} & \cdots & g^{u-1}_{k+1} & & \vdots & \ddots & \vdots \\
0 & 0 & g^0_{k+1} & \cdots & g^{u-2}_{k+1} & \cdots & 0 & \cdots & g^0_n \\
\vdots & \vdots & \vdots & \ddots & \vdots & & & & \\
0 & 0 & 0 & \cdots & g^0_{k+1} & & & \vdots & \\
 & & \vdots & & & & & & \\
z^0_{k+1} & \cdots & z^u_{k+1} & & & & z^0_n & \cdots & z^u_n \\
\vdots & \ddots & \vdots & & & \cdots & \vdots & \ddots & \vdots \\
0 & \cdots & z^0_{k+1} & & & & 0 & \cdots & z^0_n
\end{array}\right]$$

Figure 4.3.1 : The H^c-Matrix of a Convolutional Code.

a code with $k = 1$, $n = 2$, requiring $k(n - k) = 1$ code-generating polynomial.

Example 4.3.1.

The code-generating polynomial is given by

$$G_2(D) = 1 + D + D^4 + D^6, \tag{4.3.14}$$

with maximum degree $u = 6$. The information bits are the coefficients of the equivalent polynomials $M_1(D)$ and $T_1(D)$, and the parity bits are the coefficients of $T_2(D)$,

$$T_2(D) = G_2(D)\ M_1(D). \tag{4.3.15}$$

Each parity bit is the mod-2 sum of message bits delayed by zero, one, four, and six time units, the delays being represented by the coefficients of $G_2(D)$. The encoder can therefore be modelled as in Fig. 4.3.2. Note that any message digit can affect parity digits for $u + 1 = 7$ time units, in which time $n(u + 1) = 14$ channel digits are transmitted. The convolutional code is said to have (memory) constraint length n_A, where

$$n_A = n(u + 1). \tag{4.3.16}$$

As we shall see later, constraint length is one of two important optimality criteria of feedback decodable convolutional codes.

The P^c-matrix of the code, and especially the parity triangle, is of paramount importance in determining the orthogonalizability of the code. Since $G_2(D)$ is known, we may write P^c directly as in Fig. 4.3.3. Now, from (4.3.11), (4.3.12), and (4.3.13), we know that

$$\underline{S}^c = \underline{e}^c\ P^c,$$

$$\underline{S}^c = (S_2^0,\ S_2^1,\ \ldots,\ S_2^6),$$

$$\underline{e}^c = (e_1^0,\ \ldots,\ e_1^6,\ e_2^0,\ \ldots,\ e_2^6).$$

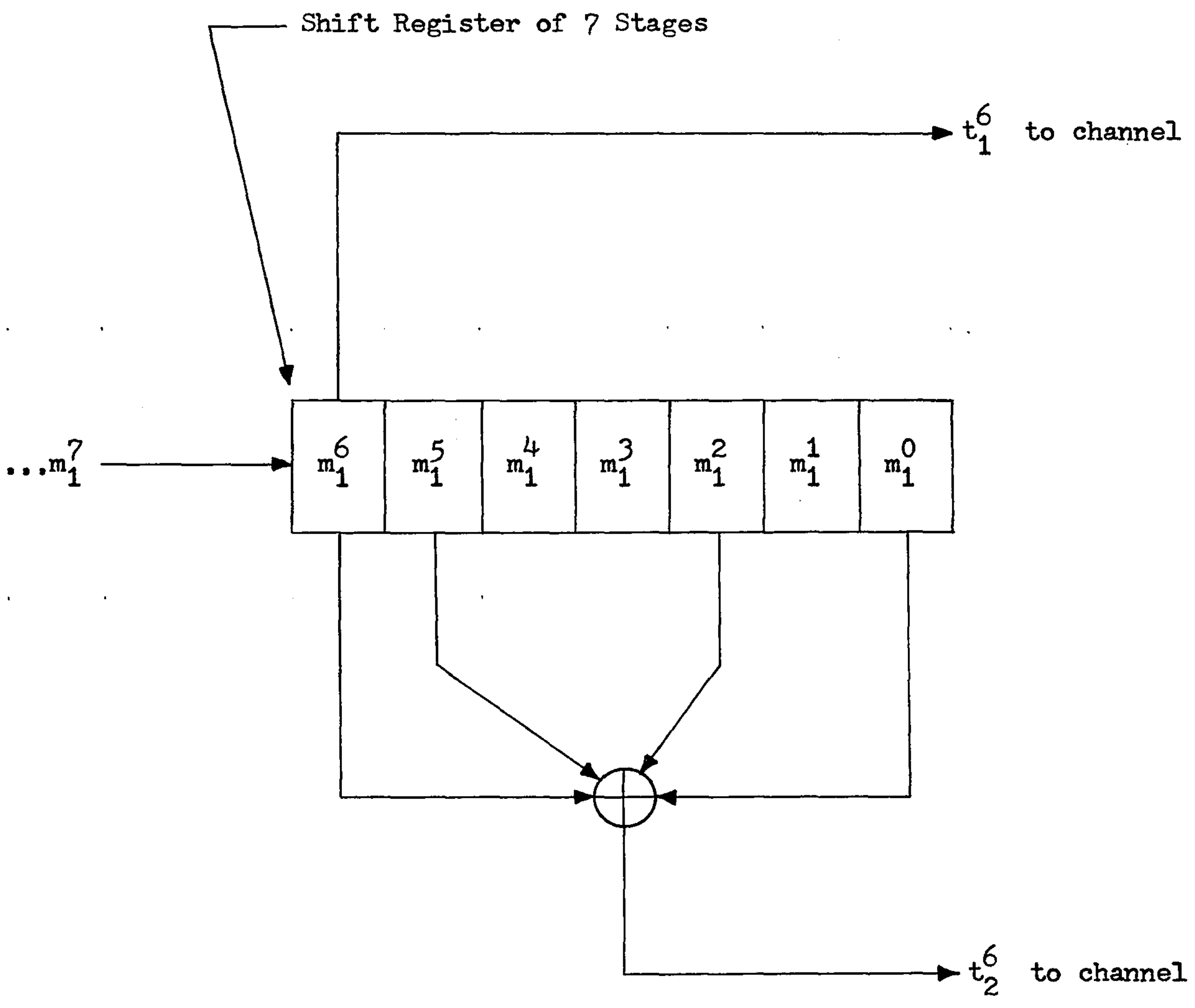


Figure 4.3.2 : Encoder for a Rate $\frac{1}{2}$ Convolutional Code.

$$P^c = \left[\begin{array}{c} H^c \\ \hdashline I^c \end{array}\right] = \left[\begin{array}{ccccccc}
1 & 1 & 0 & 0 & 1 & 0 & 1 \\
 & 1 & 1 & 0 & 0 & 1 & 0 \\
 & & 1 & 1 & 0 & 0 & 1 \\
 & & & 1 & 1 & 0 & 0 \\
 & 0 & & & 1 & 1 & 0 \\
 & & & & & 1 & 1 \\
 & & & & & & 1 \\
\hdashline
1 & & & & & & \\
 & 1 & & & & & \\
 & & 1 & & 0 & & \\
 & & & 1 & & & \\
 & 0 & & & 1 & & \\
 & & & & & 1 & \\
 & & & & & & 1
\end{array}\right]$$

Figure 4.3.3 : P^c-Matrix of a Rate $\frac{1}{2}$ Convolutional Code.

In the matrix multiplication, the sequence of parity noise digits $(e_2^0, \ldots, e_2^6)$, corresponding to the parity bits in the channel sequences $\underline{t}^j$, are multiplied by the identity matrix I^c. Clearly, then, each syndrome bit S_2^j checks the parity noise bit e_2^j and no other parity noise bit. On the other hand, the sequence of information noise digits $(e_1^0, \ldots, e_1^6)$, corresponding to the information bits in the channel sequences $\underline{t}^j$, are multiplied by the parity triangle H^c. Thus, the 1's in the jth column, $j = 0, 1, \ldots, u$, of the parity triangle determine which information noise bits are checked by the syndrome bit S_2^j. For example, if the ith and jth columns both contain a 1 in the kth row, then S_2^i and S_2^j both check the information noise bit e_1^k.

The first row of the parity triangle is uniquely defined by the coefficients of the code-generating polynomial $G_2(D)$, and each subsequent row is simply a right shift of the row above it. Thus, if $G_2(D)$ has J nonzero coefficients, there are J 1's in the first row and J syndrome bits check the noise bit e_1^0. These $J = 4$ parity-check equations are given by

$$\begin{aligned} S_2^0 &= e_1^0 + e_2^0, \\ S_2^1 &= e_1^0 + e_1^1 + e_2^1, \\ S_2^4 &= e_1^0 + e_1^3 + e_1^4 + e_2^4, \\ S_2^6 &= e_1^0 + e_1^2 + e_1^5 + e_1^6 + e_2^6. \end{aligned} \qquad (4.3.17)$$

When a convolutional code is majority decoded, the decoder attempts only to recover the information bits in the received sequences and therefore decodes only the information noise digits. This is a logical strategy since parity bits have no inherent usefulness except as a check on information bits. In order to decode the first noise bit e_1^0, the decoder must obtain a set of composite parity-checks orthogonal on e_1^0. Since e_1^0 is checked by J syndrome bits, there can be at most J (and codes are generally designed so that there are exactly J) composite parity-checks in this set.

This example was deliberately chosen so that the set of J syndrome bits $\{S_2^0, S_2^1, S_2^4, S_2^6\}$ in (4.3.17) constitutes a set of J composite parity-checks $\{A_1, A_2, A_3, A_4\}$ orthogonal on e_1^0, a fact easily verified by inspection. A code with this property is known as self-orthogonal. In Chapter 7 we shall give an example of a feedback decodable convolutional code which is not self-orthogonal.

The feedback decoder for this code is modelled in Fig. 4.3.4. Note that the decoder contains a replica of the encoder in Fig. 4.3.2. It is used to implement the parity-check equations and thereby form the syndrome bits S_2^j,

$$\begin{aligned} S_2^j &= r_1^{j-6} + r_1^{j-4} + r_1^{j-1} + r_1^j + r_2^j \\ &= e_1^{j-6} + e_1^{j-4} + e_1^{j-1} + e_1^j + e_2^j. \end{aligned} \qquad (4.3.18)$$

Note also that since $J = 4$, $\lceil J/2 \rceil = 2$ and at least three inputs to the logic element must have value 1 before the logic element can decide that $e_1^0 = 1$.

Throughout this thesis we shall be concerned with the complexity of implementation of decoders and we shall employ three parameters as measures of this complexity:

(a) the storage requirement N; i.e., the total number of stages of shift register.

(b) the total number of shift register stages which are tapped, N_T.

(c) the total number of mod-2 adders with two inputs, N_A, where it is easily shown that a mod-2 adder with K inputs, $K > 2$, is equivalent to $K - 1$ adders with two inputs.

The complexity of a feedback decoder follows from Fig. 4.3.4. The

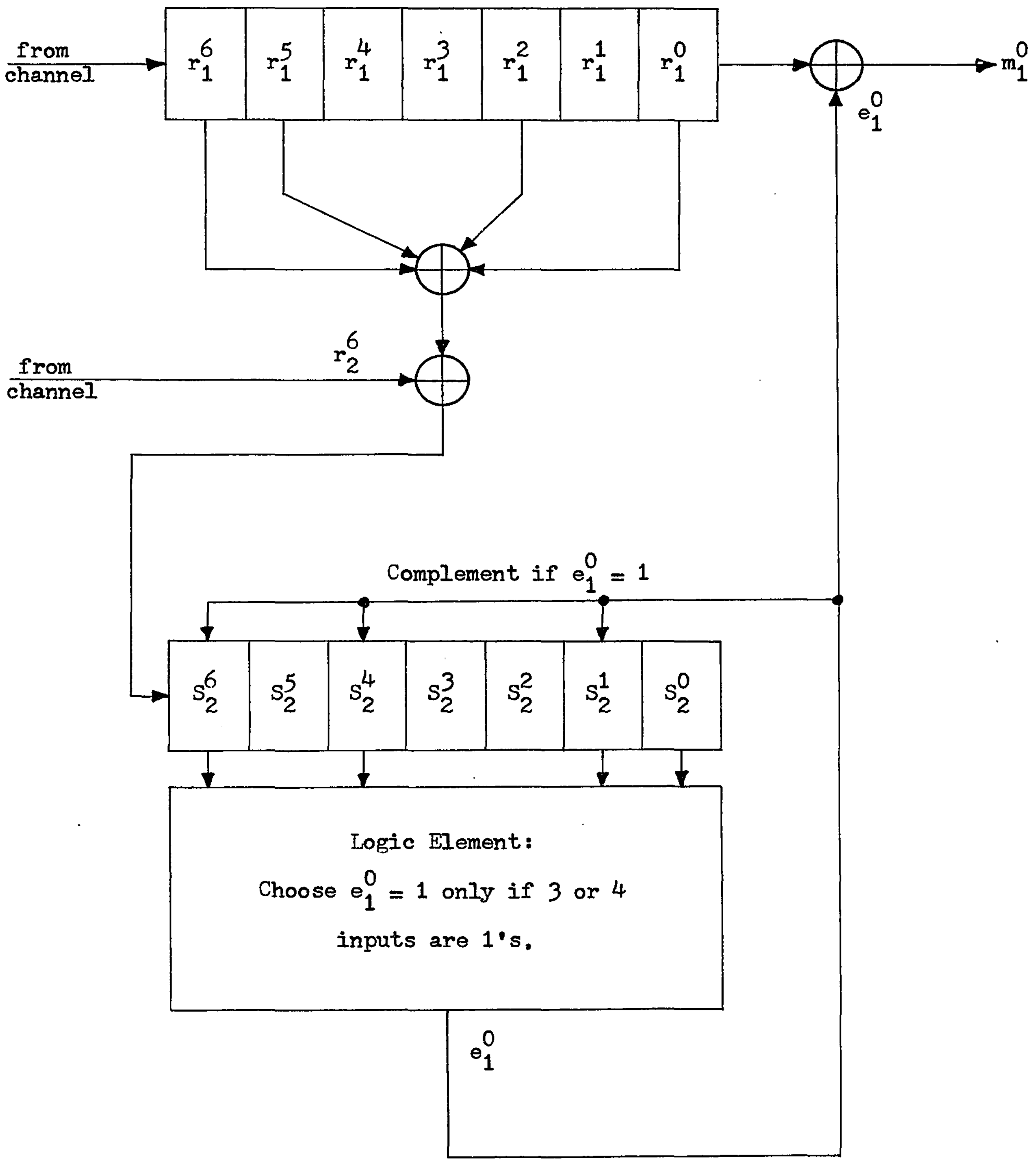


Figure 4.3.4 : Decoder for a Rate $\frac{1}{2}$ Convolutional Code.

encoder replica contains $k(u + 1) = 7$ stages of shift register, while the shift register of syndrome bits, or syndrome register, contains $(n - k)(u + 1) = 7$ stages. Thus, the storage requirement is

$$N = k(u + 1) + (n - k)(u + 1) = n(u + 1) = n_A. \tag{4.3.19}$$

We see that storage requirement and constraint length are equivalent, and this explains why constraint length is an optimality criterion.

The encoder replica is tapped in $J = 4$ locations. In addition, from Fig. 4.3.3, the tap locations correspond exactly to the positions of the 1's in the first row of the parity triangle, or, equivalently, to the nonzero coefficients in the code-generating polynomial. The syndrome register is also tapped in $J = 4$ locations. However, since the code is self-orthogonal, this represents only a lower limit. Thus,

$$N_T \geq 2J. \tag{4.3.20}$$

The decoder contains three mod-2 adders, two with two inputs and one with J inputs. Thus,

$$N_A = 2 + (J - 1) = J + 1. \tag{4.3.21}$$

If the logic element decides that $e_1^0 = 1$, then the feedback decoder complements the composite parity-checks which check e_1^0, thereby "cancelling" the error in these equations. It is unnecessary, however, to complement S_2^0 since that bit is shifted out of the syndrome register on the next cycle and no longer plays a role in decoding. The next set of inputs to the logic element is

$$S_2^1 = e_1^1 + e_2^1 \quad (e_1^0 \text{ has been removed}),$$

$$S_2^2 = e_1^1 + e_1^2 + e_2^2,$$

$$S_2^5 = e_1^1 + e_1^4 + e_1^5 + e_2^5,$$

$$S_2^7 = e_1^1 + e_1^3 + e_1^6 + e_1^7 + e_2^7. \qquad (4.3.22)$$

Clearly, the set $\{S_2^1, S_2^2, S_2^5, S_2^7\}$ constitutes a set of J composite parity-checks orthogonal on e_1^1, so decoding for e_1^1 is exactly the same as for e_1^0. Since the set of equations in (4.3.22) orthogonal on e_1^1 has precisely the same form as the set in (4.3.17) orthogonal on e_1^0, we say that the set orthogonal on e_1^0 is typical of the code. Thus, in future, we shall not say that e_1^0 is necessarily the first information noise digit. Rather, we shall allow e_1^0 to fall anywhere in the actual noise sequence and say that in decoding e_1^0, we are always decoding for a first error, provided that all previous decoding has been correct.

We can now introduce the second optimality criterion of convolutional codes, called the effective length. Quite generally, each composite parity-check A_m in the set $\{A_m\}$ orthogonal on e_1^0 checks e_1^0 plus n_m other noise bits, where n_m is called the size of A_m. Thus, the total number of distinct noise bits checked by the set $\{A_m\}$ is n_E,

$$n_E = 1 + \sum_{m=1}^{J} n_m, \qquad (4.3.23)$$

and n_E is the effective length. We shall see later that n_E is an important parameter in determining the performance of the code and that an optimal code has minimum effective length. Since the set $\{A_m\}$ is formed from at most $(n - k)(u + 1)$ syndrome bits, which together check at most n_A noise bits, it follows that

$$n_E \leq n_A. \qquad (4.3.24)$$

For rate $\frac{1}{2}$ codes, Massey [15] has shown that

$$n_E \geq \tfrac{1}{2}J^2 + \tfrac{1}{2}J + 1. \qquad (4.3.25)$$

If J is even, then the error correcting capability of the code is

$$t = \left\lfloor {}^{J}/_{2} \right\rfloor = {}^{J}/_{2}, \quad J \text{ even,}$$

and
$$n_E \geq 2t^2 + t + 1. \qquad (4.3.26)$$

The code of this example, with $J = 4$ and $t = 2$, is optimal because both n_E and n_A are minimal. To show that n_E is a minimum, we have from (4.3.26) that $n_E \geq 11$, while a simple count in (4.3.17) or (4.3.22) gives $n_E = 11$. Proving that n_A is minimal is less straightforward, but an exhaustive search shows that no self-orthogonal code exists with $J = 4$ and $n_A < 14$.

Robinson and Bernstein [38], based on the work of Singer [39], have demonstrated a good systematic method for the construction of optimal self-orthogonal codes. They have also shown that self-orthogonal codes have limited error propagation properties; i.e., a decoder always recovers from a decoding error if an error-free sequence of at least wn_A channel digits follows the decoding error, where

$$wn_A \leq x(n_A - n) + 2n_A, \qquad (4.3.27)$$

and x is the least integer such that

$$(u - J + y)\left(\frac{J - y}{y}\right)^x \leq \binom{y+1}{2} + \binom{2y - J}{2},$$

$$y = \left\lceil {}^{J}/_{2} \right\rceil + 1. \qquad (4.3.28)$$

Other optimal convolutional codes exist which are not self-orthogonal but can be orthogonalized in one step. Massey [15] describes a trial-and-error method for finding some of these codes. They generally have smaller constraint lengths than equivalent self-orthogonal codes, but as a class they do not exhibit the property that error propagation in the decoder is guaranteed to cease upon reception of an error-free sequence of known finite length. However, in most cases, an error-free sequence of some finite length, unknown in general, will act to terminate propagation. For such codes, error propagation can be controlled by periodically interrupting transmission to clear the decoder shift registers, or by requesting a retransmission of data if a counter detects more corrections over a certain interval of time than the decoder can reliably produce.

4.3.c. Performance in Error Correction

A feedback decoded convolutional code has error correcting capability $t = \lfloor J/2 \rfloor$. Thus, the probability of a decoding error, P(E), is upper-bounded by the probability that more than t errors occur among the n_E noise bits checked at the decoder. For the BSC, then,

$$P(E) \leq 1 - \sum_{j=0}^{t} \binom{n_E}{j} p_o^{\,j} (1 - p_o)^{n_E - j}. \qquad (4.3.29)$$

P(E) may be reduced by increasing t or by decreasing n_E, though these are not independent parameters. For any t, n_E may be minimized according to (4.3.26),

$$n_E \geq 2t^2 + t + 1.$$

The effect of n_E on P(E) explains why effective length is an optimality criterion of convolutional codes.

There are two important factors which affect the probability of a decoding error and which are not included in the performance expression (4.3.29). First, from Corollary 4.2.5., we know that in many cases the code will correct a large number of error patterns with weight greater than t; i.e., many convolutional codes tend to be very robust. For this reason, the right-hand side of (4.3.29) is too large. On the other hand, (4.3.29) does not consider the effect of error propagation, which makes the right-hand side too small. For simplicity, we shall assume that the effects of robustness and of error propagation on the probability of decoding error are approximately equal and opposite, so that

$$P(E) \approx 1 - \sum_{j=0}^{t} \binom{n_E}{j} p_o^{\,j} (1 - p_o)^{n_E - j}. \qquad (4.3.30)$$

In many cases this approximation is quite good.

4.3.d. Performance in Error Detection

A convolutional code with error detecting capability may simultaneously have some error correcting capability t. Suppose, for example, that t is a maximum, $t = \lfloor J/2 \rfloor$. In this case, J, the number of orthogonal composite parity-checks, must be an odd number,

$$J = 2t + 1. \qquad (4.3.31)$$

The reason for this is that the logic element of the decoder may choose $e_1^0 = 1$ only if $\lceil J/2 \rceil + 1 = t + 2$ or more composite parity-checks have value 1, and it may choose $e_1^0 = 0$ only if $\lceil J/2 \rceil = t + 1$ or more composite parity-checks have value 0. However, if exactly t + 1 composite parity-checks have value 1, neither of these conditions is met and the decoder detects an uncorrectable error pattern.

On the other extreme, we might choose t = 0. In this case the decoder decides that $e_1^0 = 0$ if and only if all J (odd or even) composite parity-checks have value 0. Otherwise, an uncorrectable error pattern is detected.

In the more general case, we say that

$$1 \leq t \leq \lfloor J/2 \rfloor, \qquad (4.3.32)$$

and we may choose J either odd or even. Then the decoder decides that $e_1^0 = 1$ only if J - t + 1 or more composite parity-checks have value 1 and that $e_1^0 = 0$ only if t - 1 or fewer composite parity-checks have value 1. Otherwise, an uncorrectable error pattern is detected.

A failure can occur only if at least J - t + 1 composite parity-checks have value 1 when $e_1^0 = 0$. Thus, the probability of failure is upper-bounded by the probability of $J - t + 1$ or more errors among the n_E noise bits checked by the decoder. For the BSC,

$$P(F) \leq 1 - \sum_{j=0}^{J-t} \binom{n_E}{j} p_o^{\,j} (1 - p_o)^{n_E - j}. \tag{4.3.33}$$

The above expression does not account for robustness or for error propagation, but again we assume that these effects approximately cancel each other, so that

$$P(F) \approx 1 - \sum_{j=0}^{J-t} \binom{n_E}{j} p_o^{\,j} (1 - p_o)^{n_E - j}. \tag{4.3.34}$$

4.4. A Note on Complexity of Implementation

In Section 4.3. we introduced the three measures of complexity which we shall use throughout this thesis: storage requirement, number of shift register tap locations, and number of mod-2 adders. We also calculated the complexity of a feedback decoder. However, we did not calculate the complexity of a minimum distance decoder for block codes. This is because there are so many different decoding algorithms available, and to study these algorithms would require a more extensive examination of parity-check codes than is needed to understand error correcting codes for the compound channel. The interested reader might consult Gallager [1], Berlekamp [3], Peterson [4], Massey [40], and Savage [41]. We shall say simply that, in general, block encoders and convolutional encoders have comparable complexities, while, except for one-step orthogonalizable parity-check codes, block decoders are much more complex than majority decoders for roughly equivalent codes.

5. BURST CORRECTING CODES

5.1. Introduction

Before going on to describe specific coding schemes for error control on the compound channel, known generally as burst correcting codes, we shall first examine in this chapter some important concepts relevant to the correction of bursts. Section 5.2. formally defines a burst and its related guard space and then relates bursts to the behaviour of the compound channel. Section 5.3. defines the burst correcting capability of a code and presents two known bounds on this capability. Section 5.4. describes the two general classes of burst correcting codes and then outlines the specific codes to be studied in the remainder of this thesis.

5.2. Burst Errors and the Compound Channel

Before we can describe a burst correcting code, we must define precisely what is meant by a burst. Using Gallager's approach [1], we postulate an arbitrary binary sequence of B consecutive noise digits $\underline{e}^B$, where

$$\underline{e}^B = (e_{j+1}, \ldots, e_{j+B}). \qquad (5.2.1)$$

Definition 5.2.1. The binary noise sequence $\underline{e}^B$ of length B is defined to be a burst of errors relative to an error-free, or clean, guard space of length G if it satisfies the following criteria:

(a) The first and last noise bits in the sequence are errors,

$$e_{j+1} = e_{j+B} = 1.$$

(b) G consecutive noise bits on each side of the sequence are error-free.

(c) There is no consecutive sequence of G error-free digits in $\underline{e}^B$.

With G fixed and B variable, we can use the above definition to segment a noise sequence of any length into a unique set of bursts. Thus, even a single error or two isolated errors with separation less than G, provided that they are bracketed by two error-free sequences of length G, are defined as bursts. However, from the point of view of the behaviour of the compound channel, it is very much more likely that these are examples of random errors embedded in a relatively long, otherwise error-free noise sequence.

Informally, bursts tend to be reasonably well-defined noise sequences with high error density, say from 1 percent to 50 percent. Bursts are separated by generally much longer sequences, or guard spaces, where errors may occur in the random mode and are therefore comparatively rare.

5.3. Burst Correcting Capability

Analogous to the error correcting capability t of random error correcting codes, burst correcting codes have a burst correcting capability B_m relative to some clean guard space G_m. Various bounds exist on the relationship of B_m to G_m. Formally, Gallager [1] defines burst correcting capability in the following way.

Definition 5.3.1. A code, or alternatively, an encoder-decoder pair, is said to have burst correcting capability B_m relative to a clean guard space G_m if every noise sequence containing only bursts of length B_m or less relative to the guard space G_m is correctly decoded, and B_m is the largest integer for which this is true.

Gallager also shows that for codes with rate R,

$$G_m \geq \left(\frac{1 + R}{1 - R}\right) B_m. \qquad (5.3.1)$$

Hereafter, we refer to (5.3.1) as the Gallager bound. This bound was originally proved for convolutional codes by Wyner and Ash [18], but Gallager generalized the proof for all codes. A somewhat stronger bound exists for the special case of (n,k) block codes with burst length smaller than the block length.

Definition 5.3.2. An (n,k) block code is said to have burst correcting capability b if it corrects every burst of length b or less located anywhere in the noise sequence of length n, provided that all other n - b noise digits are error-free.

In this case, we may consider the guard space to be of length n - b, so that the Gallager bound is

$$n - b \geq \left(\frac{1 + {}^{k}/_{n}}{1 - {}^{k}/_{n}}\right) b. \tag{5.3.2}$$

This expression reduces to a form known as the Reiger bound [19],

$$b \leq \tfrac{1}{2}(n - k). \tag{5.3.3}$$

The difference between the Reiger bound and the Gallager bound is that, in the former, the burst and the guard space may be distributed arbitrarily within a sequence of length n, while in the latter, the burst must be bracketed by two guard spaces. The Gallager bound is, of course, more general because no assumptions are made about the code or the burst length.

According to Definition 5.3.1., there are three instances when a code with burst correcting capability B_m relative to a guard space G_m is not guaranteed to decode reliably: when two bursts are separated by a clean guard space smaller than G_m; when a guard space is not clean; and when a burst is longer than B_m. Any code which will sometimes correct reliably despite the occurrence of any of the above instances is said to be robust.

5.4. Classes of Burst Correcting Codes

Broadly speaking, burst correcting codes can be classified as either adaptive or non-adaptive. The non-adaptive codes decode all received data according to a single algorithm, irrespective of the mode of the compound channel. They can reliably correct every burst within their burst correcting capability so that, at best, their guard space requirements are given by either the Gallager bound or the Reiger bound. Adaptive codes, on the other hand, attempt to deduce the channel mode and thereby decode random errors and bursts according to different algorithms. They achieve guard space requirements generally much smaller than predicted by the Gallager bound at the expense of not being able to correct every burst within their burst correcting capability. That is, adaptive codes may attempt to decode a burst according to the wrong algorithm if and when the burst is not detected.

Adaptive codes may operate in a random mode or in a burst mode. The random mode is a subcode, either an (n,k) block code or a convolutional code, with random error correcting capability t and with enough error detecting capability to detect most bursts. The burst mode has burst correcting capability B_m relative to a guard space G_m, where G_m is generally much smaller than given by the Gallager bound.

In Chapters 6, 7, 8, and 9, we shall describe four well-known coding schemes which have proved to be very effective in controlling errors on the compound channel and have manageable, in cost and hardware, implementation complexity. Interleaved block codes (Chapter 6) are non-adaptive codes whose guard space requirements at best can meet the Reiger bound. Diffuse codes (Chapter 7) are non-adaptive convolutional codes whose guard space requirements at best are asymptotic to the Gallager bound. Gallager codes (Chapter 8) are adaptive convolutional codes and Tong burst-trapping codes (Chapter 9) are adaptive block codes. Comparative performance evaluations for interleaved block codes, diffuse codes, and Tong burst-trapping codes on the real telephone channel are given by Burton [42] and by Burton and Pehlert [43].

In Chapters 10 and 11, two new methods, original with the author, will be presented. One is a concatenation scheme to allow reliable burst correction even when the guard spaces are not clean, the other, a modified burst-trapping scheme with adaptive guard space requirements immediately adjacent to the bursts.

6. INTERLEAVED BLOCK CODES

6.1. Introduction

In this chapter, we shall describe the structure, complexity, and performance of a burst correcting technique known as an interleaved, or interlaced, block code [1], [3]. Section 6.2. discusses the encoding and decoding of these codes and determines their guard space requirement. Sections 6.3. and 6.4. develop useful expressions for their complexity and performance respectively.

6.2. Structure and Guard Space Requirement

Throughout the remainder of this thesis, we shall assume that all burst correcting codes are designed for use on a compound channel on which channel bursts very rarely exceed some length B_c. In particular, an interleaved block code has burst correcting capability B_m such that

$$B_m = B_c = rb \text{ , } r \text{ and } b \text{ integers.} \tag{6.2.1}$$

The interleaved block code employs an (n,k) block code with burst correcting capability b. From the Reiger bound,

$$b \leq \tfrac{1}{2}(n - k). \tag{6.2.2}$$

The block encoder produces codewords $\underline{t}^j$, each of length n. Before transmission over the channel, r such codewords, say $\underline{t}^0, \ldots, \underline{t}^{r-1}$, are stored as a "superblock" of length rn. Conceptually, this "superblock" may be considered to be arranged in an r-by-n matrix, Fig. 6.2.1. The j^{th} row of this matrix is the j^{th} codeword $\underline{t}^j$, $j = 0, 1, \ldots, r - 1$. The i^{th} column consists of all digits t_i^j, $i = 1, 2, \ldots, n$.

Once the matrix has been completely filled by the block encoder,

t_1^0	t_2^0	$\cdots$	t_n^0
t_1^1	t_2^1	$\cdots$	t_n^1
$\vdots$	$\vdots$	$\cdots$	$\vdots$
t_1^{r-1}	t_2^{r-1}	$\cdots$	t_n^{r-1}

Figure 6.2.1 : Matrix Array for Interleaving a Block Code.

the stored digits are transmitted by column. Thus, each digit t_i^j in the codeword $\underline{t}^j$ is transmitted r channel time units after the preceding codeword digit t_{i-1}^j, where one channel time unit is the time required to transmit one channel digit. The process of separating codeword digits by r channel time units is called interleaving, or interlacing, or scrambling, to degree r; hence the term "interleaved block code."

The received digits are stored in the decoder in a "superblock" or matrix corresponding exactly to that of Fig. 6.2.1. Decoding is performed by a block decoder on the received sequences $\underline{r}^j$,

$$\underline{r}^j = \underline{t}^j + \underline{e}^j. \qquad (6.2.3)$$

Thus, decoding is executed by row on the stored matrix. Within any "superblock," a burst of length B_m or less can fill at most b consecutive complete columns of the matrix with errors, so that at most b consecutive digits in any row can be affected by the burst. Since the block code has burst correcting capability b, the burst of length B_m or less is guaranteed to be correctly decoded if no other errors occur within the "superblock."

The guard space in any one block of the code has length n - b. Thus, the guard space requirement G_m of the "superblock" is

$$G_m = r(n - b) = rn - B_m. \qquad (6.2.4)$$

G_m is a minimum if b is a maximum; i.e., if b meets the Reiger bound with equality. Corresponding to G_m, the guard space requirement of the decoder, there is a guard space of length G_c following the burst in the channel. Except in cases where the received digits are compressed in some manner before being decoded by the decoder of the burst correcting code, a situation described in Chapter 10, the decoder guard space and the channel guard space are equivalent. Thus, for interleaved block codes,

$$G_c = G_m = r(n - b) = rn - B_c. \qquad (6.2.5)$$

6.3. Complexity

As measures of the complexity of the decoder we use the parameters N (the storage requirement), N_T (the number of tapped shift register stages), and N_A (the number of mod-2 adders). As a secondary parameter, we include the ratio N/G_m of storage requirement to guard space requirement.

The decoder for an interleaved block code requires a buffer, or shift register, for the "superblock" of length rn and a block decoder, whose own storage requirement may be included in the buffer. Thus, from (6.2.4),

$$N = rn = B_m + G_m, \qquad (6.3.1)$$

and

$$N/G_m = \frac{rn}{r(n - b)} = \frac{n}{n - b} > 1. \qquad (6.3.2)$$

Because the specific implementation of the block decoder is not defined, we shall not attempt to calculate N_T or N_A for this code.

6.4. Performance

The criterion which we shall employ as a measure of the performance of a burst correcting code is the probability of a decoding error given that a channel burst has occurred, P(E | burst), where

$$P(E \mid \text{burst}) = \frac{P(E,\text{burst})}{P(\text{burst})}. \qquad (6.4.1)$$

P(E,burst) is the joint probability of a decoding error and a burst, while P(burst) is the probability of a burst. We explained in Chapter 5 that decoding errors could be caused by any of three events: a guard space which is too short, a guard space which is not clean; i.e., a noisy guard space, or a burst which is too long. We shall assume throughout that the compound channel is such that the probability of an excessively short guard space or of an excessively long burst is negligible in comparison with the probability of a noisy guard space. Thus, decoding errors are caused primarily by the occurrence of random errors in the guard space, so that, neglecting robustness,

$$P(E) = P(\text{decoding error}) = P(\text{random error in guard space}).$$

Since random errors and bursts on the compound channel are statistically independent,

$$P(E,\text{burst}) = P(E)\ P(\text{burst}) \tag{6.4.2}$$

and it follows from (6.4.1) that

$$P(E\,|\,\text{burst}) = P(E). \tag{6.4.3}$$

According to the Gilbert channel model, random errors occur with probability p_o. Thus, for interleaved block codes,

$$P(E\,|\,\text{burst}) \leq 1 - (1 - p_o)^{G_m}. \tag{6.4.4}$$

The inequality exists because the code may be robust, in which case not all random error patterns in the guard space would result in a decoding error. From (6.4.4) we see that $P(E\,|\,\text{burst})$ decreases with decreasing G_m. Thus, it is important from the point of view of performance that burst correcting codes have minimum guard space requirements.

7. DIFFUSE CODES

7.1. Introduction

In this chapter, we shall describe the structure, complexity, and performance of a burst correcting technique known as a diffuse code. Diffuse codes are feedback decodable convolutional codes with error correcting capability t and are designed to treat channel bursts of length B_c or less as if they contained no more than t "random errors." They were first reported by Kohlenberg and Forney [22] and were more thoroughly investigated by Tong [23] and by Ferguson [24].

Section 7.2. defines the optimality criteria of a diffuse code and relates them to the burst correcting capability B_m and the guard space G_m. Section 7.3. describes the structure of diffuse codes and uses these general principles to construct a specific optimal code. Section 7.4. develops expressions for the decoder complexity, Section 7.5. discusses the existence of optimal diffuse codes, and Section 7.6. develops an expression for the performance of these codes.

7.2. Optimality and Guard Space Requirement

The diffuse codes of most importance have rate $\frac{1}{2}$ and burst correcting capability B_m such that

$$B_m = B_c = 2B \text{ , B an integer.} \tag{7.2.1}$$

Simultaneously, they have random error correcting capability t,

$$t = {}^{J}/_{2} \text{ , J even.} \tag{7.2.2}$$

The optimality criteria of diffuse codes, like all feedback decodable convolutional codes, are constraint length n_A and effective length n_E. From

(4.3.25) and (4.3.26),

$$n_E \geq \tfrac{1}{2}J^2 + \tfrac{1}{2}J + 1 = 2t^2 + t + 1. \qquad (7.2.3)$$

Also, from (4.3.19),

$$n_A = k(u + 1) + (n - k)(u + 1) = N, \qquad (7.2.4)$$

where $k(u + 1)$ is the length of the encoder replica in the decoder, $(n - k)(u + 1)$ is the length of the syndrome register in the decoder, and N is the total storage requirement. Since we consider only rate $\frac{1}{2}$ codes, it follows that

$$k = n - k,$$

and

$$N_s = k(u + 1) = (n - k)(u + 1) = \tfrac{1}{2}n_A. \qquad (7.2.5)$$

We shall call N_s the shift register length. Clearly, since the decoder contains two shift registers,

$$N = 2N_s. \qquad (7.2.6)$$

We shall now relate shift register length to both the burst length B_m and the guard space requirement G_m.

In order that all bursts of length B_m or less, relative to the guard space G_m, be correctly decoded, G_m is given by the Gallager bound,

$$G_m \geq \left(\frac{1 + \frac{1}{2}}{1 - \frac{1}{2}}\right) B_m = 3B_m = 6B. \qquad (7.2.7)$$

Any consecutive sequence of 6B channel bits contains exactly 3B information bits and 3B parity bits. Thus, when either the first error or the last error in a burst is the only one contained in the encoder replica, the Gallager bound demands that at least 3B error-free information bits, the guard space, also be contained in the encoder replica. It follows that the shift register length is lower-bounded by

$$N_s \geq 3B + 1, \qquad (7.2.8)$$

or, equivalently,

$$n_A = N \geq 6B + 2 = 3B_m + 2. \qquad (7.2.9)$$

If G is the guard space required in the encoder replica, then

$$N_s = G + 1,$$

and
$$G_m = G_c = 2G = N - 2 = n_A - 2, \qquad (7.2.10)$$

where G_c is the channel guard space corresponding to the decoder guard space G_m.

If N_s meets the bound (7.2.8) with equality; i.e.,

$$N_s = N_s^{opt} = 3B + 1,$$

and
$$G_m = 3B_m,$$

then the code is optimal with respect to constraint length. If the weaker condition holds that

$$N_s \longrightarrow N_s^{opt} \quad \text{as} \quad B_m \longrightarrow \infty,$$

then the code is said to be asymptotically optimal with respect to constraint length.

7.3. Structure

A diffuse code with burst correcting capability B_m and error correcting capability $t = {}^J/_2$ is structured in such a way that bursts of length B_m or less never appear at the decoder to contain more than t errors among the n_E noise bits that are checked. The manner in which this is accomplished depends on Corollary 4.2.4., paraphrased below in Corollary 7.3.1..

Corollary 7.3.1. If $e_1^0 = 1$, no more than $t - 1$ of the A_m in the set of 2t composite parity-checks $\{A_m\}$ orthogonal on e_1^0 may check other errors, and if $e_1^0 = 0$, no more than t of the A_m may check any errors.

Ferguson [24] employs a useful terminology. We say that any digit e_i^j is q-dependent if its superscript is of the form

$$j = qB + a \text{ , } 0 \leq a < B \text{ , } q \text{ an integer.} \tag{7.3.1}$$

If e_1^0 is the first error in a burst of length B_m or less, the burst can affect only the 0-dependent information noise bits, e_1^0, e_1^1, ..., e_1^{B-1}, and the 0-dependent parity noise bits, e_2^0, e_2^1, ..., e_2^{B-1}. Since the guard space is given by (7.2.7),

$$G_m \geq 3B_m \quad \text{or} \quad G \geq 3B, \tag{7.3.2}$$

this means that all 1-, 2-, and 3-dependent noise bits must be error-free.

Suppose that $e_1^0 = 1$. Then a burst may affect only 0-dependent noise bits and Corollary 7.3.1. demands that no more than t - 1 of the A_m check other 0-dependent noise bits. Alternatively, suppose that $e_1^0 = 0$. Then a burst might be embedded somewhere among the 1-, 2-, or 3-dependent noise bits. Corollary 7.3.1. demands that no more than t of the A_m check any i-dependent noise bits, i = 1, 2, 3.

To clarify the foregoing development, let us consider the construction of an (asymptotically) optimal diffuse code for, say, J = 4 and t = 2. The code-generating polynomial $G_2(D)$, or, equivalently, the first row of the parity triangle H^c, is to be found.

The parity triangle, Fig. 7.3.1, has four 1's in the first row, two of which are located in the first and last columns. The remaining two may be assigned arbitrary locations such that x, y, and z represent the number of 0's between the 1's. The length of the first row defines the shift register length N_s and the locations of the 1's correspond exactly to the tap locations on the encoder replica. From (7.2.8),

$$N_s = x + y + z + 4 \geq 3B + 1. \tag{7.3.3}$$

We may write the four parity-check equations which check e_1^0 directly as

$$\begin{tikzcd}
S_2^{0} & S_2^{x+1} & S_2^{x+y+2} & S_2^{x+y+z+3} \\
1 \arrow[r, leftrightarrow, "x" description] & 1 \arrow[r, leftrightarrow, "y" description] \arrow[d, leftrightarrow, "x" description] & 1 \arrow[r, leftrightarrow, "z" description] \arrow[d, leftrightarrow, "y" description] & 1 \arrow[d, leftrightarrow, "z" description] \\
 & 1 & 1 \arrow[d, leftrightarrow, "x" description] & 1 \arrow[d, leftrightarrow, "y" description] \\
 & & 1 & 1 \arrow[d, leftrightarrow, "x" description] \\
 & & & 1
\end{tikzcd}$$

Figure 7.3.1 : Parity Triangle for a Rate $\frac{1}{2}$ Diffuse Code, J = 4.

$$S_2^0 = e_1^0 + e_2^0,$$

$$S_2^{x+1} = e_1^0 + e_1^{x+1} + e_2^{x+1},$$

$$S_2^{x+y+2} = e_1^0 + e_1^{y+1} + e_1^{x+y+2} + e_2^{x+y+2},$$

$$S_2^{x+y+z+3} = e_1^0 + e_1^{z+1} + e_1^{y+z+2} + e_1^{x+y+z+3} + e_2^{x+y+z+3}. \qquad (7.3.4)$$

Since the syndrome bit S_2^0 checks the 0-dependent parity noise bit e_2^0, one composite parity-check must also check e_2^0. Also, since $t - 1 = 1$, no other composite parity-check may check any 0-dependent bits other than e_1^0. This means that e_1^{x+1}, e_1^{y+1}, and e_1^{z+1} may not be 0-dependent; i.e.,

$$x,\ y,\ z \geq B - 1. \qquad (7.3.5)$$

If the code were to be self-orthogonal, then x, y, and z, besides being bounded by (7.3.5), would all have to be different in order to ensure that all n_E noise bits in (7.3.4) would be different. However, Tong [23] has proved that for a rate $\frac{1}{2}$ self-orthogonal diffuse code, the shift register length is at best asymptotic to the bound

$$N_s \geq (t + 2)\ B. \qquad (7.3.6)$$

Since we wish our code to have shift register length asymptotically 3B, not 4B, we may abandon any hope that the code be self-orthogonal. It follows that x, y, and z need not all be different.

Suppose we allow two of the variables to be equal,

$$x = y \neq z,$$

and, in order to minimize shift register length, we let the variables assume minimum values as defined by (7.3.5),

$$x = B - 1\ ,\ y = B - 1\ ,\ z = B. \qquad (7.3.7)$$

The shift register length is, from (7.3.3) and (7.2.8),

$$N_s = x + y + z + 4 = 3B + 2 = N_s^{opt} + 1, \qquad (7.3.8)$$

so the code is asymptotically optimal with respect to constraint length. The set of parity-check equations in (7.3.4) can be rewritten as

$$\begin{aligned} s_2^0 &= e_1^0 + e_2^0, \\ s_2^B &= e_1^0 + e_1^B + e_2^B, \\ s_2^{2B} &= e_1^0 + e_1^B + e_1^{2B} + e_2^{2B}, \\ s_2^{3B+1} &= e_1^0 + e_1^{B+1} + e_1^{2B+1} + e_1^{3B+1} + e_2^{3B+1}. \end{aligned} \qquad (7.3.9)$$

We have already conceded that the code cannot be self-orthogonal and this is confirmed by the fact that s_2^B and s_2^{2B} above both check the noise bit e_1^B. In addition, s_2^B, s_2^{2B}, and s_2^{3B+1} all check 1-dependent bits, which is not allowed by Corollary 7.3.1.. In order to orthogonalize the set of parity-check equations, and at the same time satisfy the corollary, we remove the 1-dependence from s_2^{2B} by adding to it (mod-2) some syndrome bit s_2^j. A good choice of s_2^j is s_2^{3B}, where

$$s_2^{3B} = e_1^B + e_1^{2B} + e_1^{3B} + e_2^{3B}. \qquad (7.3.10)$$

We replace s_2^{2B} in (7.3.9) by the composite parity-check $s_2^{2B} + s_2^{3B}$, where

$$\begin{aligned} s_2^{2B} + s_2^{3B} &= (e_1^0 + e_1^B + e_1^{2B} + e_2^{2B}) + (e_1^B + e_1^{2B} + e_1^{3B} + e_2^{3B}) \\ &= e_1^0 + e_1^{3B} + e_2^{2B} + e_2^{3B}. \end{aligned} \qquad (7.3.11)$$

This new set of composite parity-checks, (7.3.9) and (7.3.11), is orthogonal on e_1^0 and satisfies all the requirements of Corollary 7.3.1.. In addition, by a simple count, $n_E = 11$, so that the effective length of the code is minimal from (7.2.3). The feedback decoder for this diffuse code is shown in Fig. 7.3.2.

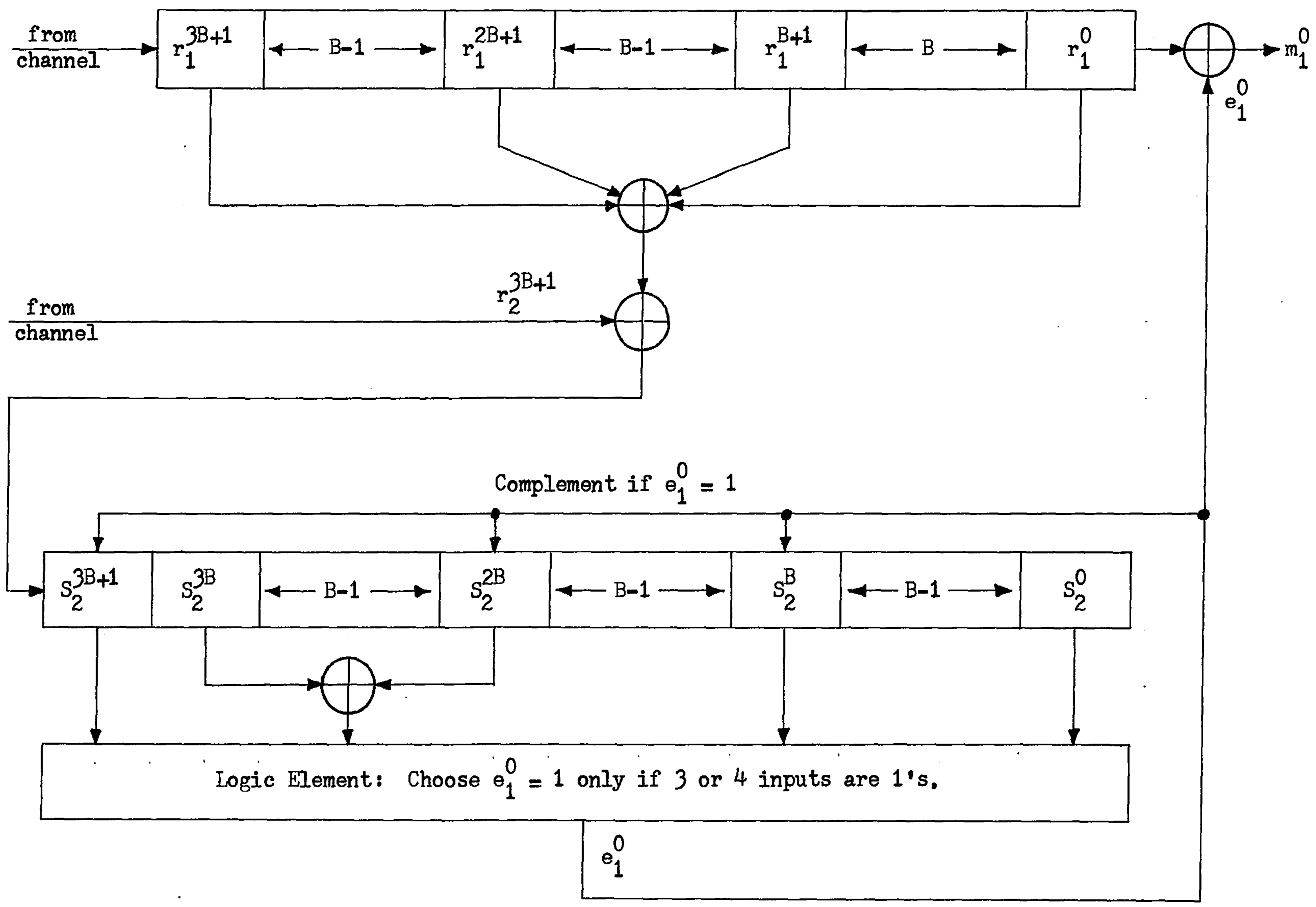


Figure 7.3.2 : Decoder for a Rate ½ Diffuse Code, t = 2.

7.4. Complexity

The three measures of the complexity of the decoder of a diffuse code, N, N_T, and N_A, may be determined easily from Fig. 7.3.2. From (7.2.10), the storage requirement is

$$N = G_m + 2, \tag{7.4.1}$$

$$N/G_m = \frac{G_m + 2}{G_m} \longrightarrow 1 \quad \text{as} \quad B_m \longrightarrow \infty . \tag{7.4.2}$$

The encoder replica is tapped in exactly J locations. Since an (asymptotically) optimal code cannot be self-orthogonal, the syndrome register is tapped in at least J + 1 locations. Thus, the number of tapped shift register stages is

$$N_T \geq 2J + 1 = 4t + 1, \tag{7.4.3}$$

At the encoder replica, the decoder requires one mod-2 adder with J inputs and two with two inputs. In addition, the syndrome register requires at least one adder because the code must be orthogonalized. Thus, the number of mod-2 adders is

$$N_A \geq (J - 1) + 2 + 1 = 2t + 2, \tag{7.4.4}$$

7.5. Existence of Optimal Codes

The diffuse code which we derived in Section 7.3. was the only example given in the first published description of diffuse codes by Kohlenberg and Forney [22]. They implied that, although no unified theory for the design of diffuse codes existed, this example for $t = 2$ with minimal effective length and asymptotically optimal constraint length was typical. Ferguson [24], [44] disagreed. Based on Tong's treatment of self-orthogonal diffuse codes [23], he developed the rules of a trial-and-error procedure to design asymptotically optimal diffuse codes for $t \geq 2$. He found that for $t \geq 3$, it was not evident that codes existed with simultaneously minimal effective length. For example, for $t = 3$ and asymptotically optimal constraint length, the best codes known to the author were found by Tabak [45] with $n_E = 24$ and by Ferguson with $n_E = 25$, whereas, from (7.2.3), $n_E^{opt} = 22$. However, codes with minimal effective length were found [24] which have shift register length asymptotically 4B rather than 3B. Ferguson therefore conjectured that diffuse codes with both minimal effective length and asymptotically optimal constraint length do not, in fact, exist for $t \geq 3$.

7.6. Performance

Our criterion of performance is the probability of decoding error given that a burst has occurred, $P(E \mid \text{burst})$. As in Section 6.4., we assume that decoding errors are caused primarily by random errors in the guard space G_m. Thus, analogous to (6.4.4),

$$P(E \mid \text{burst}) \leq 1 - (1 - p_o)^{G_m}. \tag{7.6.1}$$

The above expression neglects both robustness and error propagation in the feedback decoder. We again assume that these effects are approximately equal and opposite, so that

$$P(E \mid \text{burst}) \approx 1 - (1 - p_o)^{G_m}. \tag{7.6.2}$$

Since n_A and G_m are linearly related by (7.2.10), we see that it is important from the point of view of performance that diffuse codes be designed with minimal constraint length.

A secondary criterion of performance of a diffuse code is the probability of a decoding error given that the compound channel is in its random mode, $P(E \mid \text{random})$. In this case, the compound channel behaves roughly like a BSC, so that P(E | random) is given by (4.3.30),

$$P(E \mid \text{random}) \approx 1 - \sum_{j=0}^{t} \binom{n_E}{j} p_o^{\,j} (1 - p_o)^{n_E - j}. \qquad (7.6.3)$$

8. GALLAGER CODES

8.1. Introduction

In this chapter, we shall describe the structure, complexity, and performance of an adaptive burst correcting technique called a Gallager code by Kohlenberg and Forney [22] and called a time-diversity code by Gallager [1]. The random mode of a Gallager code is a feedback decodable convolutional code, while the burst mode is provided by a simple extension of the convolutional code.

Section 8.2. describes the structure of a Gallager code, including decoding procedures in both modes and the mechanism of decoder transitions between modes. Section 8.3. calculates the guard space requirement of the code and compares it to the Gallager bound. Section 8.4. determines the decoder complexity, and Section 8.5. develops expressions for the performance of the code.

8.2. Structure

The Gallager codes which we shall consider have rate $\frac{1}{2}$ and burst correcting capability B_m such that

$$B_m = B_c = 2B \text{ , B an integer.} \tag{8.2.1}$$

Since Gallager codes are adaptive, they include a random mode to correct random errors and to detect bursts and a burst mode to correct the detected bursts.

The random mode is provided by a rate $\frac{1}{2}$ feedback decodable convolutional code with code-generating polynomial $G_2(D)$. If $G_2(D)$ has maximum degree u and has J nonzero coefficients, then from (4.3.16) and (7.2.5) the constraint length is

$$n_A^* = n(u + 1) = 2(u + 1) = 2N_s^*, \tag{8.2.2}$$

and from (4.3.25) the effective length is

$$n_E \geq \tfrac{1}{2}J^2 + \tfrac{1}{2}J + 1. \tag{8.2.3}$$

Because the convolutional code might be required to have some error detecting capability, the error correcting capability t of the code is given by (4.3.32),

$$1 \leq t \leq \lfloor J/2 \rfloor . \tag{8.2.4}$$

There are two very simple ways in which the random mode can be used to detect bursts. In the first, choose t small in (8.2.4) so that the convolutional code has a large amount of error detecting capability. Then, at the logic element of the decoder, if J - t + 1 or more of the J composite parity-checks orthogonal on e_1^0 have value 1, choose $e_1^0 = 1$. If t - 1 or fewer composite parity-checks have value 1, choose $e_1^0 = 0$. Otherwise, do not decide on e_1^0 and defer to the burst mode of the decoder.

In the second burst detection scheme, the convolutional code need not have any explicit error detecting capability so that t may be chosen a maximum in (8.2.4). Now, if the compound channel were in its random mode, it would be highly unlikely that the logic element would repeatedly decide that a channel error had occurred. In the channel burst mode, however, such a situation would appear to be highly likely due to the large probability of both channel errors and error propagation. Thus, a counter at the logic element output could be used to detect a burst if some minimum density of 1's were detected.

For the burst mode of the Gallager code, a minimum of B stages of shift register are appended to the right of the encoder of the convolutional code, with a tap location at the rightmost stage. Thus, the shift register length N_s of the Gallager code is at least

$$N_s = B + N_s^* = B + u + 1, \tag{8.2.5}$$

and the constraint length n_A is

$$n_A = 2N_s = B_m + 2(u + 1) = B_m + n_A^*. \tag{8.2.6}$$

The encoder for a Gallager code is shown in Fig. 8.2.1.

To demonstrate the decoding procedures of a Gallager code, we assume that the decoder is initially in its random mode and that the received bit r_1^0,

$$r_1^0 = t_1^0 + e_1^0, \tag{8.2.7}$$

has just entered the encoder replica. As r_1^0 is shifted through the first $N_s^* = u + 1$ stages of the encoder replica, a set of J syndrome bits $\{S_2^0, \ldots, S_2^u\}$ is formed, each of which checks e_1^0, where

$$\begin{aligned} S_2^0 &= e_1^0 + e_2^0, \\ &\vdots \\ S_2^u &= e_1^0 + \ldots + e_1^u + e_2^u. \end{aligned} \tag{8.2.8}$$

These J syndrome bits are stored in the first N_s^* stages of the syndrome register.

When r_1^0 is in the last, or N_sth, stage of the encoder replica, the syndrome bit S_2^{B+u} is formed,

$$S_2^{B+u} = e_1^0 + e_1^B + \ldots + e_1^{B+u} + e_2^{B+u}, \tag{8.2.9}$$

and is stored in the first stage of the syndrome register. At this point, the J syndrome bits $S_2^0, \ldots, S_2^u$ occupy the last u + 1 stages of the syndrome register.

Assume that the convolutional code of the random mode has large error detecting capability. Then the set of syndrome bits $\{S_2^0, \ldots, S_2^u\}$ is used to form a set of J composite parity-checks orthogonal on e_1^0. If e_1^0 is

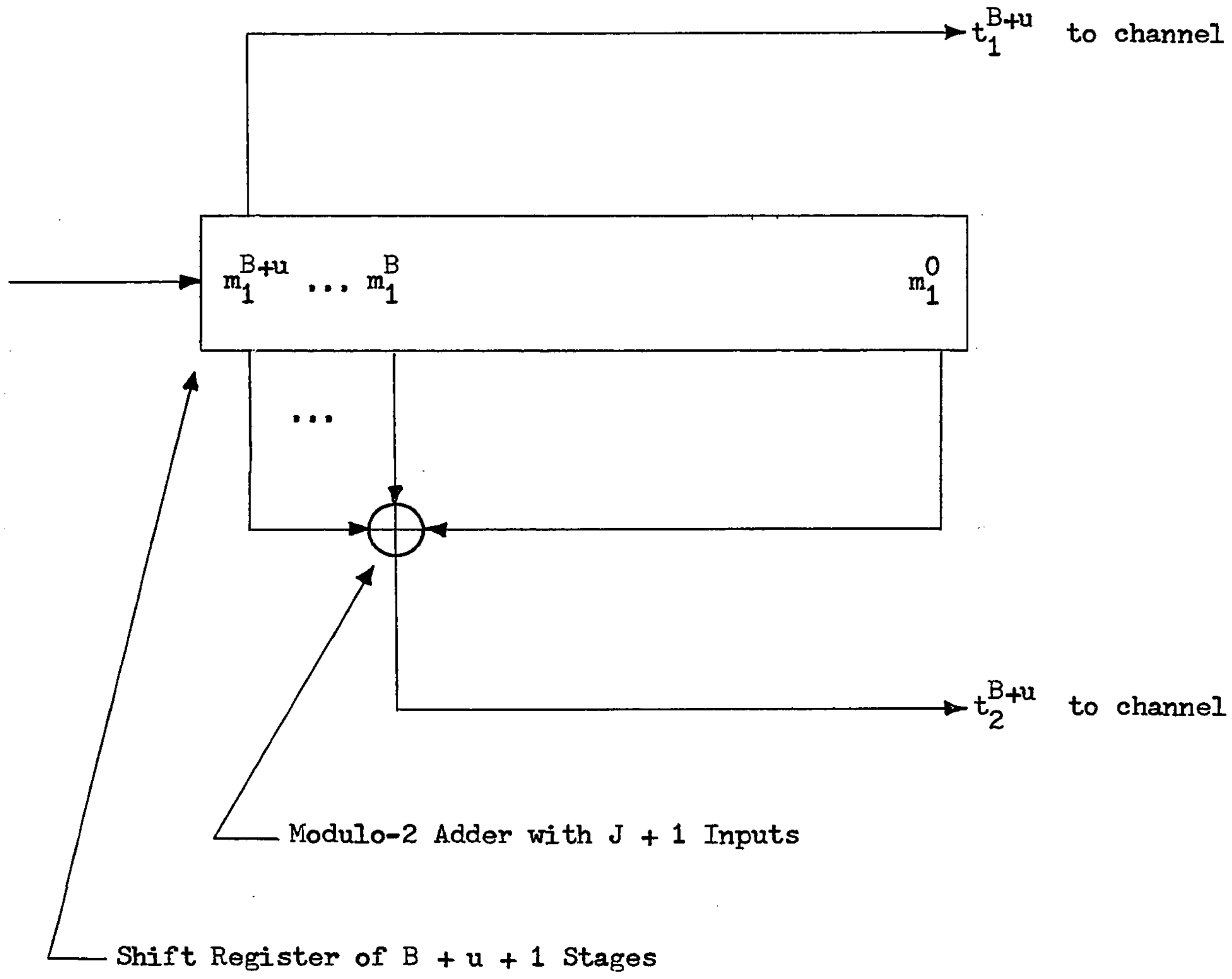

t_1^{B+u} to channel
m_1^{B+u} ... m_1^{B}
m_1^{0}
...
t_2^{B+u} to channel
Modulo-2 Adder with J + 1 Inputs
Shift Register of B + u + 1 Stages

Figure 8.2.1 : Encoder for a Rate $\frac{1}{2}$ Gallager Code.

correctable, the decoder continues operation in the random mode. If e_1^0 is not correctable, the decoder switches to the burst mode.

When e_1^0 is the first error in a burst of length B_m or less, the burst can affect at most the B information noise bits $e_1^0, e_1^1, \ldots, e_1^{B-1}$ and the corresponding B parity noise bits $e_2^0, e_2^1, \ldots, e_2^{B-1}$. The clean guard space of length G_m following the burst must span the u + 1 pairs of noise bits $e_1^B, e_2^B, \ldots, e_1^{B+u}, e_2^{B+u}$. Thus, from (8.2.9),

$$S_2^{B+u} = e_1^0 + e_1^B + \ldots + e_1^{B+u} + e_2^{B+u} = e_1^0. \qquad (8.2.10)$$

Thus, the burst mode of the decoder simply decides that e_1^0 has the value of the syndrome bit S_2^{B+u}. The decoder is shown in Fig. 8.2.2.

The decoder remains in its burst mode until it receives some indication that the burst has ended. Once a burst has passed completely through the encoder replica, only digits which are part of the guard space will remain. If the guard space is clean, all syndrome bits will have value 0. In particular, if a certain number y of consecutive 0's appear in the first stage of the syndrome register, denoted S_2^{B+u} in Fig. 8.2.2, then the decoder switches back to its random mode.

8.3. Guard Space Requirement

To determine the decoder guard space requirement G_m, we assume that e_1^0 is the first error in a burst of length 2b,

$$2b \leq B_m = 2B. \qquad (8.3.1)$$

The burst affects the b pairs of noise bits $e_1^0, e_2^0, \ldots, e_1^{b-1}, e_2^{b-1}$.

To decode e_1^0, the burst mode uses the syndrome bit S_2^{B+u}. From (8.2.10),

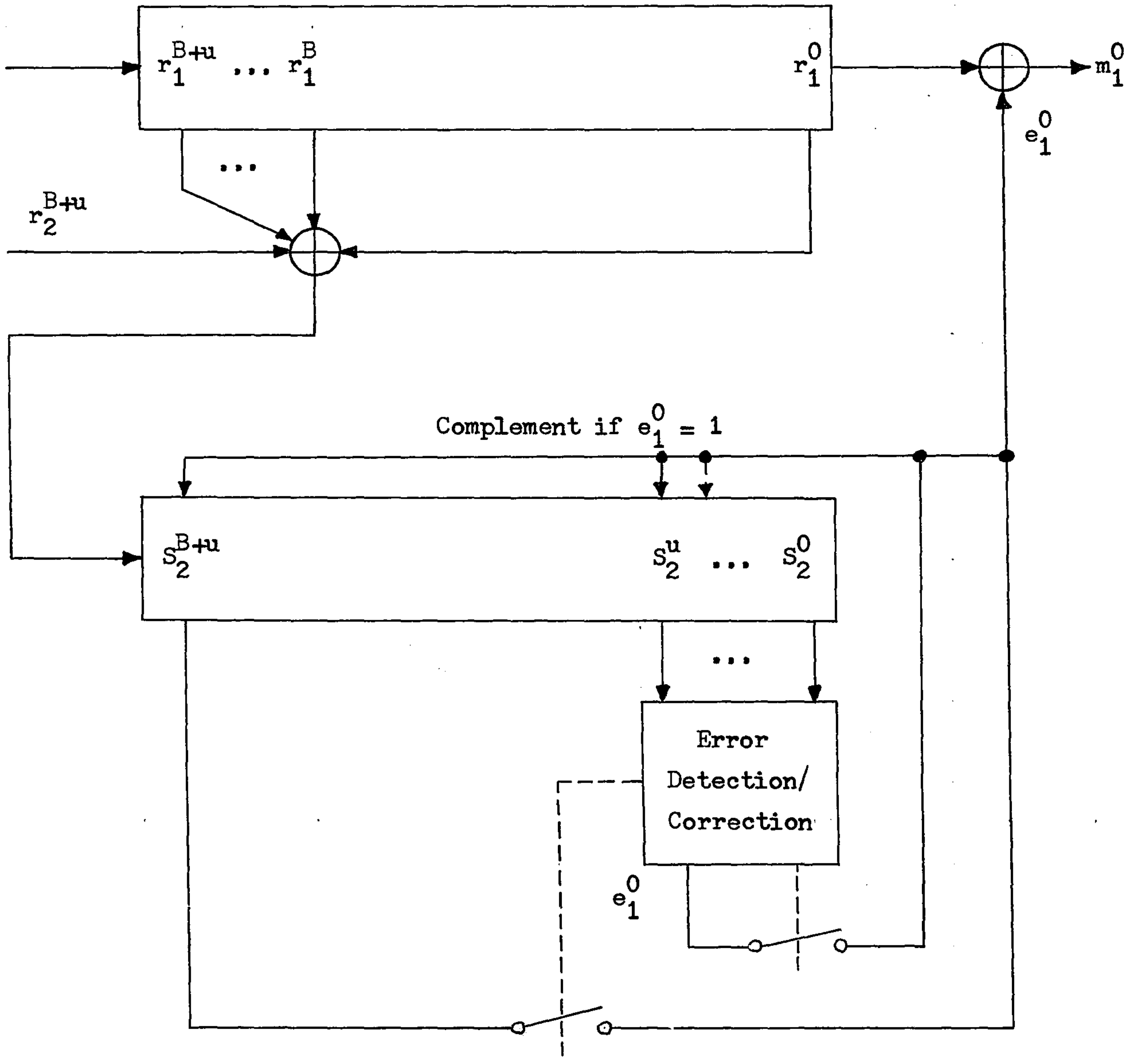


Figure 8.2.2 : Decoder for a Rate ½ Gallager Code.

$$S_2^{B+u} = e_1^0 + e_1^B + \dots + e_1^{B+u} + e_2^{B+u}. \qquad (8.3.2)$$

To decode e_1^{b-1}, the burst mode must use $S_2^{B+u+b-1}$,

$$S_2^{B+u+b-1} = e_1^{b-1} + e_1^{B+b-1} + \dots + e_1^{B+u+b-1} + e_2^{B+u+b-1}. \qquad (8.3.3)$$

In order that the decoder switch out of the burst mode, the succeeding y syndrome bits, S_2^{B+u+b}, ..., $S_2^{B+u+b-1+y}$, must be zero-valued, where

$$\begin{aligned} S_2^{B+u+b} &= e_1^b + e_1^{B+b} + \dots + e_1^{B+u+b} + e_2^{B+u+b}, \\ &\vdots \\ S_2^{B+u+b-1+y} &= e_1^{b-1+y} + e_1^{B+b-1+y} + \dots + e_1^{B+u+b-1+y} + e_2^{B+u+b-1+y}. \end{aligned} \qquad (8.3.4)$$

From (8.3.2) and (8.3.3), in order that the burst be corrected, the decoder guard space must span the noise bits e_1^B, e_2^B, ..., $e_1^{B+u+b-1}$, $e_2^{B+u+b-1}$. From (8.3.4), in order that the decoder exit from the burst mode, the guard space must span the noise bits e_1^b, e_2^b, ..., e_1^{b-1+y}, e_2^{b-1+y} and e_1^{B+b}, e_2^{B+b}, ..., $e_1^{B+u+b-1+y}$, $e_2^{B+u+b-1+y}$. Taking into account the overlap in these guard space requirements, we see that the guard space must span the 2y consecutive digits immediately following the burst, e_1^b, e_2^b, ..., e_1^{b-1+y}, e_2^{b-1+y}, and the 2(b + u + y) consecutive digits displaced a length 2B from the beginning of the burst, e_1^B, e_2^B, ..., $e_1^{B+u+b-1+y}$, $e_2^{B+u+b-1+y}$. In addition, as b approaches its maximum value B, there may be overlap between the adjacent guard space of length 2y and the displaced guard space of length 2(b + u + y).

The adjacent and displaced guard spaces first become a continuous guard space when $b = B - y$, where the total guard space length G is

$$G = 2y + 2(B - y + u + y) = 2(B + u + y). \qquad (8.3.5)$$

When $b = B$, there is complete overlap of the two guard spaces so that

$$G = 2(B + u + y). \qquad (8.3.6)$$

We may therefore write the decoder guard space requirement as

$$G = G_{db} = 2(b + u + 2y)\ ,\ b < B - y,$$

$$G = G_m = 2(B + u + y) = B_m + 2(u + y)\ ,\ B - y \leq b \leq B. \qquad (8.3.7)$$

Because the guard space depends on the length of the actual burst and not on the length of the maximum correctable burst, the guard space is called adaptive. The corresponding adaptive channel guard space, G_{cb} or G_c, is

$$G_{cb} = G_{db}\ ,\ b < B - y,$$

$$G_c = G_m\ ,\ B - y \leq b \leq B. \qquad (8.3.8)$$

Since we consider Gallager codes with rate $\frac{1}{2}$, the Gallager bound is

$$G_{db},\ G_m \geq \left(\frac{1 + \frac{1}{2}}{1 - \frac{1}{2}}\right) 2b = 6b. \qquad (8.3.9)$$

From (8.3.7), when $b < B - y$, we know that $G_{db} = 2b + 2u + 4y$. The parameters u and y are typically small compared to B and are independent of B, so that G_{db} is smaller than the Gallager bound except for small values of b,

$$b \leq y + \tfrac{1}{2}u. \qquad (8.3.10)$$

For large values of b, G_{db} or G_m approaches one-third the limit predicted by the Gallager bound.

This apparent paradox is easily explained. The Gallager bound applies to non-adaptive codes, such as diffuse codes, which are guaranteed to correct all bursts of length B_m or less. Adaptive codes, such as Gallager codes, can only correct detected bursts of length B_m or less. Any burst which is not detected is certain to cause a decoding error.

8.4. Complexity

The three parameters of complexity, N, N_T, and N_A, may be easily determined from Fig. 8.2.2. First, from (4.3.19), (8.2.6), and (8.3.7), the storage requirement N is

$$N = n_A = B_m + 2(u + 1) = G_m - 2(y - 1). \tag{8.4.1}$$

It follows that

$$N/G_m = \frac{G_m - 2(y - 1)}{G_m} \longrightarrow 1 \quad \text{as} \quad B_m \longrightarrow \infty . \tag{8.4.2}$$

The number of tapped shift register stages N_T and the number of mod-2 adders N_A are both dependent on the choice of convolutional code for the random mode. Since N_T and N_A are both minimized if this code is self-orthogonal, we use this case as a lower bound. The decoder contains two shift registers, each with at least J + 1 tap locations. Thus,

$$N_T \geq 2J + 2. \tag{8.4.3}$$

Similarly, the decoder contains at least one mod-2 adder with J + 2 inputs and one with two inputs, so that

$$N_A \geq (J + 1) + 1 = J + 2. \tag{8.4.4}$$

8.5. Performance

We consider the probability of a decoding error given that a burst has occurred, P(E | burst), where

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)\, P(\text{no } F) + P(E \mid F)\, P(F). \tag{8.5.1}$$

P(F) is the probability of failure; i.e., the probability that the burst is neither corrected nor detected by the random mode. P(no F) is the probability that there is no failure, so that

$$P(\text{no } F) = 1 - P(F). \tag{8.5.2}$$

P(E | F) is the probability of a decoding error given a failure by the random mode. Since a burst which is not detected is certain to cause a decoding error,

$$P(E \mid F) = 1. \tag{8.5.3}$$

P(E | no F) is the probability of a decoding error when the burst is detected and is therefore analogous to the probability $P(E \mid \text{burst})$ of a non-adaptive code. It follows that for an adaptive code,

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)[1 - P(F)] + P(F). \qquad (8.5.4)$$

Thus, the probability of decoding error is bounded away from zero by the probability $P(F)$.

If the convolutional code of the random mode has error detecting capability, we know that $P(F)$ is given by an equation analogous to (4.3.34). Since, in the channel burst mode, errors are produced with probability q_o, we have

$$P(F) \approx 1 - \sum_{j=0}^{J-t} \binom{n_E}{j} q_o^{\,j} (1 - q_o)^{n_E - j}. \qquad (8.5.5)$$

$P(F)$ may be decreased by increasing J, decreasing t, and decreasing n_E. However, J and n_E are related by (8.2.3),

$$n_E \geq \tfrac{1}{2}J^2 + \tfrac{1}{2}J + 1.$$

The probability $P(E \mid \text{no } F)$ is primarily the probability of a random error in the guard space. Analogous to (6.4.4) and (7.6.1),

$$P(E \mid \text{no } F) \leq 1 - (1 - p_o)^{G_{db}}. \qquad (8.5.6)$$

If the effects of robustness and error propagation are approximately equal and opposite, then

$$P(E \mid \text{no } F) \approx 1 - (1 - p_o)^{G_{db}}. \qquad (8.5.7)$$

A secondary criterion of performance is the probability of a decoding error in the channel random mode, $P(E \mid \text{random})$, where

$$P(E \mid \text{random}) = P(E \mid \text{no } A)\, P(\text{no } A) + P(E \mid A)\, P(A). \qquad (8.5.8)$$

$P(A)$ is the probability of false alarm; i.e., the probability that the decoder switches to its burst mode while the channel is acting in its random mode.

P(no A) is the probability of no false alarm, so that

$$P(\text{no } A) = 1 - P(A). \tag{8.5.9}$$

$P(E \mid A)$ is the probability of a decoding error given a false alarm by the burst detector, and $P(E \mid \text{no } A)$ is the probability of a decoding error when no false alarm occurs.

There can be a false alarm only if between t and J - t of the J composite parity-checks have value 1 while the channel is in its random mode. Thus, P(A) is lower-bounded by the probability of between t and J - t random errors among n_E noise bits,

$$P(A) \leq \sum_{j=t}^{J-t} \binom{n_E}{j} p_o^{\,j} (1 - p_o)^{n_E - j}. \tag{8.5.10}$$

If the decoder is in its burst mode due to a false alarm, we assume that decoding errors occur primarily because of random errors in the guard space of length G_{db}. Thus, from (8.5.7),

$$P(E \mid A) \approx P(E \mid \text{no } F) \approx 1 - (1 - p_o)^{G_{db}}. \tag{8.5.11}$$

If there is no false alarm, then a decoding error can occur only if there are at least J - t + 1 random errors among n_E noise bits,

$$P(E \mid \text{no } A) \leq 1 - \sum_{j=0}^{J-t} \binom{n_E}{j} p_o^{\,j} (1 - p_o)^{n_E - j}. \tag{8.5.12}$$

Sullivan [25] has developed a generalization of the Gallager codes which features improved performance by being very tolerant of random errors in the channel guard space. Sullivan's scheme employs a convolutional code within a convolutional code. The outermost code is used as the random mode of the Gallager code, while the innermost code is used to correct random errors in the channel guard space before they can affect the decoder guard space of the

Gallager code. A somewhat similar scheme for extending or generalizing all burst correcting codes is described in Chapter 10.

9. TONG BURST-TRAPPING CODES

9.1. Introduction

In this chapter, we shall describe the structure, complexity, and performance of an adaptive burst correcting technique discovered by Tong [26], called a burst-trapping code. The random mode of a burst-trapping code is a systematic (n,k) parity-check code, while the burst mode is provided by an extension of the block code.

Section 9.2. describes the structure and decoding procedures of a burst-trapping code. Section 9.3. calculates and discusses the guard space requirement of the code. Section 9.4. determines the decoder complexity, and Section 9.5. develops expressions for the performance of the code.

9.2. Structure

Burst-trapping codes are fairly straightforward extensions of the systematic (n,k) parity-check codes which provide their random mode. They have rate R_o such that

$$R_o = \frac{k}{n} = \frac{x-1}{x}\ , \ x \text{ an integer,} \qquad (9.2.1)$$

and they have burst correcting capability B_m such that

$$B_m = B_c = vn\ , \ v \text{ an integer.} \qquad (9.2.2)$$

The block code of the random mode has minimum distance d and error correcting capability t,

$$t < \left\lfloor \frac{d-1}{2} \right\rfloor . \qquad (9.2.3)$$

The particular choice of t is a trade-off between the random error correcting capability and the error detecting capability required of the code.

To show how the burst-trapping code is obtained from the block code, we note that the block code has 2^k codewords $\underline{t}^j$,

$$\underline{t}^j = (t_1^j, t_2^j, \ldots, t_n^j). \tag{9.2.4}$$

Since the block code is systematic, each message sequence $\underline{m}^j$ is given by

$$\underline{m}^j = (m_1^j, \ldots, m_k^j) = (t_1^j, \ldots, t_k^j). \tag{9.2.5}$$

However, from (9.2.1), it follows that

$$k = (x - 1)(n - k) \ , \ n = x(n - k), \tag{9.2.6}$$

so that $\underline{m}^j$ may be divided into $x - 1$ segments of length $n - k$. Each such segment, $\underline{I}_i^j$, $i = 1, 2, \ldots, x - 1$, is called an information sub-block, and

$$\underline{m}^j = (\underline{I}_1^j, \underline{I}_2^j, \ldots, \underline{I}_{x-1}^j), \tag{9.2.7}$$

$$\underline{I}_i^j = (t^j_{(n-k)(i-1)+1}, \ldots, t^j_{(n-k)i}). \tag{9.2.8}$$

In addition, the $n - k$ parity bits of $\underline{t}^j$ may be represented by a sequence $\underline{P}^j$, called the parity sub-block. Thus,

$$\underline{P}^j = (t_{k+1}^j, \ldots, t_n^j), \tag{9.2.9}$$

and

$$\underline{t}^j = (\underline{m}^j, \underline{P}^j) = (\underline{I}_1^j, \ldots, \underline{I}_{x-1}^j, \underline{P}^j). \tag{9.2.10}$$

The basic principle of Tong's burst-trapping scheme is as follows. Any burst of length B_m or less can affect at most v codewords, $\underline{t}^j$, $\underline{t}^{j+1}$, ..., $\underline{t}^{j+v-1}$. Thus, $\underline{t}^{j+v}$ is unaffected, and if some of the information bits in $\underline{t}^j$ could also be included in $\underline{t}^{j+v}$, then those bits could always be recovered from at least one of the two codewords. This can in fact be done without altering the rate R_o of the code simply by adding (mod-2) one of the information sub-blocks of $\underline{t}^j$ to the parity sub-block of $\underline{t}^{j+v}$. Specifically, the i^{th} information sub-block of $\underline{t}^j$ is added to the parity sub-block of $\underline{t}^{j+iv}$, $i = 1, 2, \ldots, x - 1$.

The burst-trapping code has codewords $\underline{T}^j$ of the form

$$\underline{T}^j = (\underline{I}_1^j, \ldots, \underline{I}_{x-1}^j, \underline{Q}^j), \tag{9.2.11}$$

$$\underline{Q}^j = \underline{P}^j + \underline{I}_1^{j-v} + \underline{I}_2^{j-2v} + \dots + \underline{I}_{x-1}^{j-(x-1)v}. \qquad (9.2.12)$$

We shall call this process of substituting the sequence $\underline{Q}^j$ for the parity sub-block $\underline{P}^j$ time-diversification to degree v. Tong [26] calls this process interleaving to degree v, but this terminology could lead to confusion with the distinct process of interleaving described in Chapter 6.

For each codeword $\underline{T}^j$, the decoder receives a block $\underline{R}^j$,

$$\underline{R}^j = \underline{T}^j + \underline{e}^j, \qquad (9.2.13)$$

where $\underline{e}^j$ is the binary channel noise sequence of length n. Corresponding to $\underline{R}^j$, there is a minimum distance decodable sequence $\underline{r}^j$ such that

$$\underline{r}^j = \underline{t}^j + \underline{e}^j. \qquad (9.2.14)$$

Distinguishing received sub-blocks by a left superscript r, we may write

$$\underline{R}^j = ({}^r\underline{I}_1^j, \dots, {}^r\underline{I}_{x-1}^j, {}^r\underline{Q}^j),$$

$$\underline{r}^j = ({}^r\underline{I}_1^j, \dots, {}^r\underline{I}_{x-1}^j, {}^r\underline{P}^j). \qquad (9.2.15)$$

Thus, $\underline{r}^j$ may be recovered from $\underline{R}^j$ simply by forming the mod-2 sum, from (9.2.12),

$${}^r\underline{P}^j = {}^r\underline{Q}^j + \underline{I}_1^{j-v} + \underline{I}_2^{j-2v} + \dots + \underline{I}_{x-1}^{j-(x-1)v}. \qquad (9.2.16)$$

It is important in (9.2.16) that the information sub-blocks be error-free in order to maintain the same noise sequence $\underline{e}^j$ in both $\underline{R}^j$ and $\underline{r}^j$. The decoder, therefore, must store the decoded message sequences of the preceding (x - 1)v received blocks.

In order to demonstrate fully the decoding procedures of a burst-trapping code and to show how the decoder is implemented, it is convenient to use a numerical example.

Example 9.2.1.

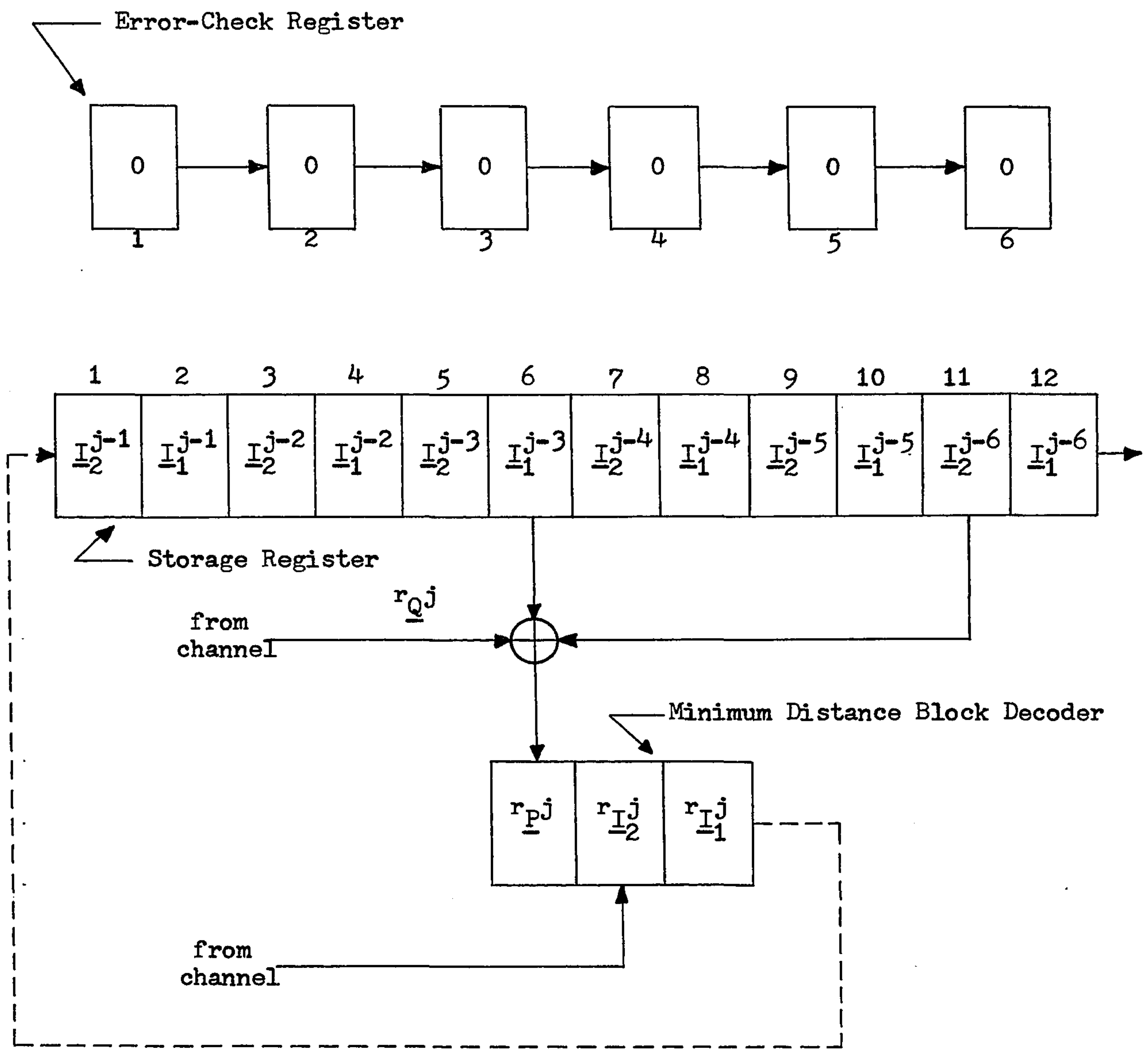
The code parameters are x = 3, v = 3, and the block $\underline{R}^j$ has just been received by the decoder. Assume that the decoder is initially in its random mode and that all previous decoding has been correct.

The decoder, Fig. 9.2.1, contains three shift registers. The first is part of the minimum distance decoder of the (n,k) parity-check code and, initially, is used to store the x - 1 information sub-blocks, ${}^{r}\underline{I}_1^j$ and ${}^{r}\underline{I}_2^j$, of the received block $\underline{R}^j$. The second, called the storage register, is used to store the decoded message sequences of the previous (x - 1)v blocks. This implies that $(x - 1)^2 v = 12$ information sub-blocks are stored, and these twelve sub-block locations are numbered in the figure. The third shift register, called the error-check register, is used to label the (x - 1)v message sequences in the storage register as either presumably correct or definitely unreliable by means of 0's and 1's respectively. As we shall see, it is the state of the error-check register that controls the mode of the decoder. The $(x - 1)v = 6$ stages of the error-check register are numbered in the figure. Because previous decoding has been correct, the error-check register is initially in the state (0, 0, 0, 0, 0, 0).

The decoder can operate in its random mode only if the minimum distance decodable sequence $\underline{r}^j$ can be recovered from $\underline{R}^j$ according to (9.2.16). This requires that the x - 1 decoded message sequences $\underline{m}^{j-v}$, $\underline{m}^{j-2v}$, ..., $\underline{m}^{j-(x-1)v}$ be correct. Equivalently, the error-check register stages numbered v, 2v, ..., (x - 1)v must contain 0's. If any of these stages contains a 1, then $\underline{r}^j$ cannot be recovered and the decoder will operate in one of x - 1 burst modes, one for each of the x - 1 information sub-blocks.

In Fig. 9.2.1, since error-check register stages number 3 and number 6 contain 0's, the decoder operates in the random mode. The sequence $\underline{r}^j$ is recovered from $\underline{R}^j$ by forming the mod-2 sum

$${}^{r}\underline{P}^j = {}^{r}\underline{Q}^j + \underline{I}_1^{j-3} + \underline{I}_2^{j-6}. \qquad (9.2.17)$$



Error-Check Register
0 0 0 0 0 0
1 2 3 4 5 6
1 2 3 4 5 6 7 8 9 10 11 12
$\underline{I}_2^{j-1}$ $\underline{I}_1^{j-1}$ $\underline{I}_2^{j-2}$ $\underline{I}_1^{j-2}$ $\underline{I}_2^{j-3}$ $\underline{I}_1^{j-3}$ $\underline{I}_2^{j-4}$ $\underline{I}_1^{j-4}$ $\underline{I}_2^{j-5}$ $\underline{I}_1^{j-5}$ $\underline{I}_2^{j-6}$ $\underline{I}_1^{j-6}$
Storage Register
$r_{\underline{Q}}j$
from channel
Minimum Distance Block Decoder
$r_{\underline{P}}j$ $r_{\underline{I}_2}j$ $r_{\underline{I}_1}j$
from channel

Figure 9.2.1 : Decoder for a Rate 2/3 Burst-Trapping Code: Random Mode.

This is accomplished by adding ${}^{r}\underline{Q}^{j}$, taken directly from the channel, to the contents of storage register sub-block locations number 6 and number 11. The result ${}^{r}\underline{P}^{j}$ is stored in the minimum distance decoder along with ${}^{r}\underline{I}_1^{j}$ and ${}^{r}\underline{I}_2^{j}$.

If the minimum distance decoder decides that $\underline{r}^{j}$ contains t or fewer errors, it makes the necessary corrections to the information bits. Otherwise a burst is detected in the j th block. Then the decoder shifts $\underline{I}_1^{j-6}$ and $\underline{I}_2^{j-6}$ out of the storage register to the source decoder, shifts $\underline{I}_1^{j}$ and $\underline{I}_2^{j}$ (or ${}^{r}\underline{I}_1^{j}$ and ${}^{r}\underline{I}_2^{j}$) into the storage register, right-shifts the error-check register, inserting a 0 (or a 1) in stage number 1, and accepts the next block $\underline{R}^{j+1}$.

The decoder operates in the random mode unless a 1 appears in any one of the error-check register stages numbered v, 2v, ..., (x - 1)v. In particular, a 1 in stage number iv results in decoder operation in a burst mode such as to correct the i th information sub-block, i = 1, 2, ..., x - 1.

Suppose, for example, that a burst is detected in $\underline{r}^{j}$. Then a 1 first appears in stage number 3 when $\underline{R}^{j+3}$; i.e., $\underline{R}^{j+v}$, is received by the decoder. This is shown in Fig. 9.2.2. $\underline{R}^{j+3}$ is assumed to be part of a clean guard space of the burst in $\underline{R}^{j}$, so that

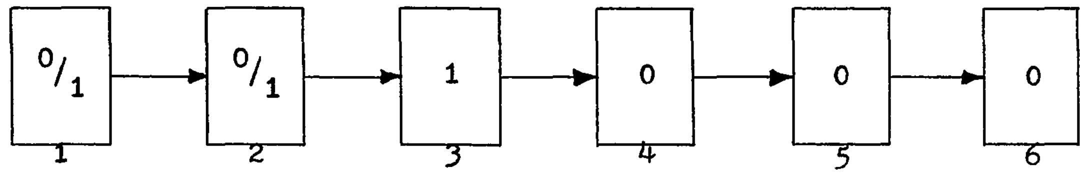
$$\underline{R}^{j+3} = \underline{T}^{j+3} = (\underline{I}_1^{j+3}, \underline{I}_2^{j+3}, \underline{Q}^{j+3}). \qquad (9.2.18)$$

Since the information sub-blocks are known, the decoder attempts to recover $\underline{I}_1^{j}$. Analogous to (9.2.12),

$$\underline{I}_1^{j} = \underline{Q}^{j+3} + \underline{P}^{j+3} + \underline{I}_2^{j-3}. \qquad (9.2.19)$$

The decoder calculates $\underline{P}^{j+3}$ from $\underline{I}_1^{j+3}$ and $\underline{I}_2^{j+3}$ by using a replica of the encoder of the (n,k) parity-check code. It adds $\underline{P}^{j+3}$ to $\underline{Q}^{j+3}$, taken directly from the channel, and to the contents of storage register sub-block location number 11. The result $\underline{I}_1^{j}$ is used to replace ${}^{r}\underline{I}_1^{j}$ in sub-block location number 6.

A 1 first appears in error-check register stage number 6 when $\underline{R}^{j+6}$,



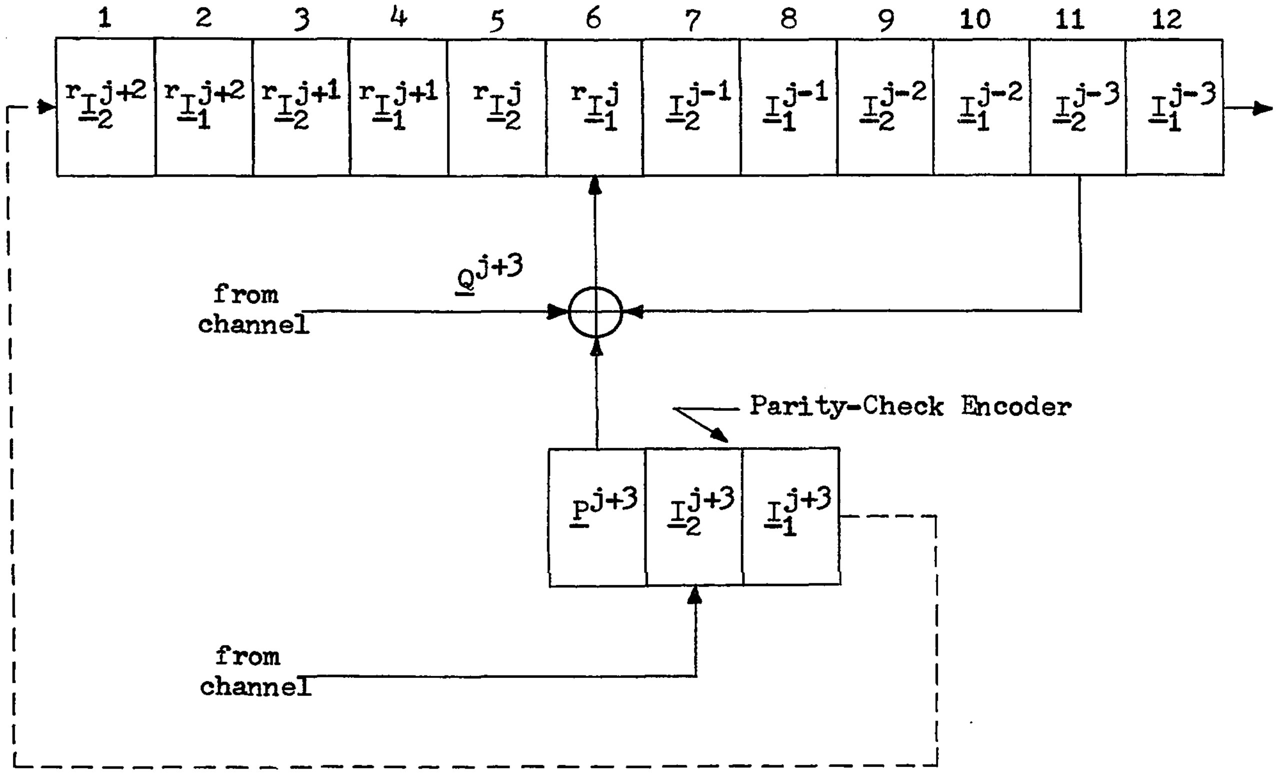


Figure 9.2.2 : Decoder for a Rate 2/3 Burst-Trapping Code: Burst Mode.

or $\underline{R}^{j+2v}$, is received by the decoder. At the same time, stage number 3 must contain a 0 since $\underline{R}^{j+3}$, or $\underline{R}^{j+v}$, is part of the clean guard space. This is shown in Fig. 9.2.3. $\underline{R}^{j+6}$ is also assumed to be part of the clean guard space of the burst in $\underline{R}^{j}$. Thus, the decoder attempts to recover $\underline{I}_2^j$ from

$$\underline{I}_2^j = \underline{Q}^{j+6} + \underline{P}^{j+6} + \underline{I}_1^{j+3}. \tag{9.2.20}$$

$\underline{P}^{j+6}$ is obtained from the parity-check encoder replica, $\underline{Q}^{j+6}$ is taken directly from the channel, and $\underline{I}_1^{j+3}$ is taken from sub-block location number 6. $\underline{I}_2^j$ replaces ${}^{r}\underline{I}_2^j$ in sub-block location number 11 and the message sequence $\underline{m}^j$ is completely recovered.

9.3. Guard Space Requirement

The guard space requirement of a burst-trapping code may be determined by generalizing Example 9.2.1.. If a burst is detected in some arbitrary block $\underline{R}^j$, then the clean guard space must include only the x - 1 succeeding blocks $\underline{R}^{j+v}$, $\underline{R}^{j+2v}$, ..., $\underline{R}^{j+(x-1)v}$. That is, for each block in the burst, the guard space contains x - 1 blocks at intervals of v blocks. Thus, analogous to the Gallager codes of Chapter 8, the guard space does not necessarily span consecutive blocks and its total length is proportional only to the length of the actual burst, not to the length of the longest correctable burst B_m.

If a burst affects y blocks, $y \leq v$, then the decoder guard space G_{dy} contains (x - 1)y clean blocks,

$$G_{dy} = (x - 1)yn \ , \ y = 1, 2, \ldots, v. \tag{9.3.1}$$

Because G_{dy} is proportional to yn, the guard space is called adaptive. The corresponding adaptive channel guard space G_{cy} is

$$G_{cy} = G_{dy}. \tag{9.3.2}$$

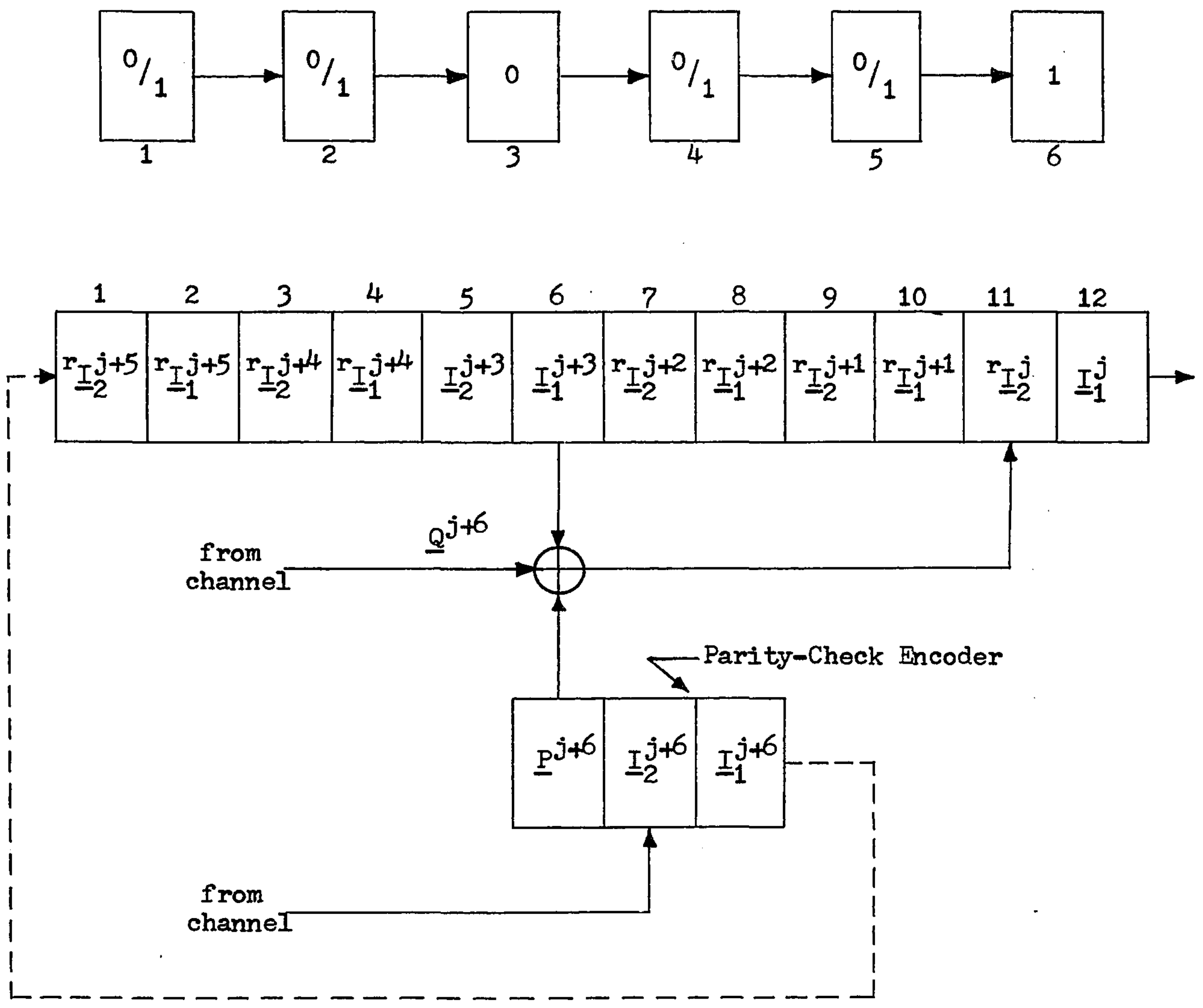


Figure 9.2.3 : Decoder for a Rate 2/3 Burst-Trapping Code: Burst Mode.

If $y = v$, then

$$G_{dy} = G_m = (x - 1)vn = (x - 1)\, B_m, \tag{9.3.3}$$

$$G_{cy} = G_c = G_m = (x - 1)\, B_c. \tag{9.3.4}$$

Since the burst-trapping code has rate $R_o = {}^{x-1}/_x$, the Gallager bound on guard space requirement is

$$G_{dy} \geq \frac{1 + R}{1 - R}\, yn = (2x - 1)yn. \tag{9.3.5}$$

From (9.3.1), the burst-trapping code has guard space requirement between one-half and one-third that predicted by the Gallager bound.

9.4. Complexity

The parameters of decoder complexity, N, N_T, and N_A, are easily determined from Figs. 9.2.1, 9.2.2, and 9.2.3. In the decoder, the minimum distance decoder, or encoder replica, contains n stages, the storage register contains $(x - 1)^2\, v\, (n - k)$ or $(x - 1)vk$ stages, and the error-check register contains $(x - 1)v$ stages. The storage requirement N is therefore given by

$$N = (x - 1)v(k + 1) + n. \tag{9.4.1}$$

From (9.3.3) it follows that

$$N/G_m = \frac{(x - 1)v(k + 1) + n}{(x - 1)vn}. \tag{9.4.2}$$

If the (n,k) parity-check code is to be effective in detecting bursts, it must have a large minimum distance d. This implies that $n - k$ must be large,

$$n - k \gg 1\ ,\ n > k + 1. \tag{9.4.3}$$

Thus,
$$N/G_m \longrightarrow \frac{k + 1}{n} < 1 \quad \text{as} \quad B_m \longrightarrow \infty\,. \tag{9.4.4}$$

To determine the number of tapped shift register stages N_T and the number of mod-2 adders N_A, we note that the decoder forms sums analogous to (9.2.12),

$$\underline{Q}^j = \underline{P}^j + \underline{I}_1^{j-v} + \dots + \underline{I}_{x-1}^{j-(x-1)v}.$$

The sequence $\underline{Q}^j$ is always fed to an array of mod-2 adders directly from the channel. The remaining x sequences, however, are always fed directly from, or to, tapped stages of shift register. Since each sequence occupies n - k stages, and since the same x sub-block locations are used in forming every sum, we have

$$N_T = x(n - k) = n. \qquad (9.4.5)$$

If we consider the summations to be performed by an array of n - k mod-2 adders with x inputs, then

$$N_A = (n - k)(x - 1) = k. \qquad (9.4.6)$$

9.5. Performance

To determine the probability of a decoding error given that a burst has occurred, $P(\dot{E} \mid \text{burst})$, we know from (8.5.4) that for an adaptive code,

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)[1 - P(F)] + P(F). \qquad (9.5.1)$$

The probability of failure P(F) is defined by the parameters of the (n,k) parity-check code. From (3.3.10), (3.3.11), and (3.3.12), since channel errors occur in the burst mode with probability q_o,

$$P(F) = P_d \, P_N,$$

$$P_N = 2^{k-n} \sum_{j=0}^{t} \binom{n}{j},$$

$$P_d = 1 - \sum_{j=0}^{d-t-1} \binom{n}{j} q_o^{\,j} (1 - q_o)^{n-j}. \qquad (9.5.2)$$

The probability of a decoding error given that the burst is detected, $P(E \mid \text{no } F)$, is primarily the probability of a random error in the guard space. Burst-trapping codes, though, are not robust since the decoder assumes that all received blocks in the guard space are error-free and takes no steps to check the validity of this assumption. Thus, analogous to (8.5.7),

$$P(E \mid \text{no } F) \approx 1 - (1 - p_o)^{G_{dy}}. \tag{9.5.3}$$

The probability of a decoding error in the channel random mode is given by (8.5.8) and (8.5.9),

$$P(E \mid \text{random}) = P(E \mid \text{no } A)[1 - P(A)] + P(E \mid A)\, P(A). \tag{9.5.4}$$

A false alarm occurs when the received sequence is distance between t + 1 and d - t - 1 from the transmitted codeword, given that the channel is in its random mode. Thus, the probability P(A) of false alarm is approximately the probability of a noise sequence with weight between t + 1 and d - t - 1,

$$P(A) \approx \sum_{j=t+1}^{d-t-1} \binom{n}{j} p_o^{\,j} (1 - p_o)^{n-j}. \tag{9.5.5}$$

Analogous to (8.5.11) and (9.5.3),

$$P(E \mid A) \approx P(E \mid \text{no } F) \approx 1 - (1 - p_o)^{G_{dy}}. \tag{9.5.6}$$

Given no false alarm, a decoding error can occur only if the noise sequence has weight at least d - t. Thus,

$$P(E \mid \text{no } A) = 1 - \sum_{j=0}^{d-t-1} \binom{n}{j} p_o^{\,j} (1 - p_o)^{n-j}. \tag{9.5.7}$$

Because the decoder of a burst-trapping code contains feedback, the possibility of error propagation exists. Tong [26] shows that if a decoding error occurs, then error propagation is limited to W blocks,

$$W \le (x - 1)v + \left\lfloor \frac{(x - 2)t}{d - 2t} \right\rfloor \left\lfloor \frac{x}{2} \right\rfloor v, \tag{9.5.8}$$

provided these W blocks are error-free. However, the probability that propagation would continue to the limit of (9.5.8) is very small if d - t is large.

The fact that burst-trapping codes are not robust is of no great importance on compound channels where random errors are extremely rare. On channels where this is not true, measures must be taken to correct errors in the channel guard space before they can affect the decoder guard space. Burton, Sullivan, and Tong [27] have proposed a scheme, called generalized burst-trapping codes, which has this capability. The principle is similar to the generalized Gallager codes of Sullivan [25] and to the codes of Chapter 10.

10. COMPOUND-CONCATENATED SYSTEMS

10.1. Introduction

In this chapter, we describe a method for improving the performance of burst correcting codes when the compound channel exhibits noisy guard spaces between bursts. The method, which we call a compound-concatenated system, utilizes a random error correcting code concatenated [28] with a burst correcting code. The purpose of the random error correcting code is to control errors in the channel guard space before they can affect the decoder guard space of the burst correcting code.

Section 10.2. defines concatenated codes and explains how the error correcting code of a compound-concatenated system controls channel guard space errors. Sections 10.3. and 10.4. discuss the structure of compound-concatenated systems and calculate the guard space requirement when the error correcting code is either a block code or a convolutional code. Section 10.5. determines the complexity of compound-concatenated systems in comparison with that of burst correcting codes alone. Sections 10.6. and 10.7. derive expressions for the performance of compound-concatenated systems.

10.2. Concatenated Codes

We shall describe the principle of concatenated codes, introduced by Forney [28], and then develop more fully the concept of a compound-concatenated system.

Suppose that code X with rate R_X is used to transmit information over some arbitrary channel. Then the system described by encoder X-channel-decoder X may be considered to be a "superchannel." Depending upon the correcting capabilities of code X, the "superchannel" has a relatively clean output, only a fraction R_X of which consists of information digits. If code Y with rate R_Y is used to transmit information over the "superchannel," then code Y is said to be concatenated with code X, Fig. 10.2.1.

The overall rate R_s of the system of concatenated codes is

$$R_s = R_X R_Y < R_X, R_Y. \qquad (10.2.1)$$

Thus, the system has a rate lower than that of either code alone. However, if code X, called the inner code, and code Y, called the outer code, are chosen to correct sufficiently different classes of error patterns, then many errors not corrected by one code will be corrected by the other. It follows that the performance of the concatenated system, in terms of the probability of a decoding error, is superior to that of either code alone.

In a compound-concatenated system, the inner code is a random error correcting code, either a minimum distance decodable (n,k) parity-check code or a feedback decodable convolutional code. When the compound channel is in its random mode, the inner code is designed to decode correctly with very high probability, so that the output of the "superchannel" is very clean. When the channel is in its burst mode, the inner code is ineffective and the output of the "superchannel" is a burst of decoding errors. The length of the burst of decoding errors is related to the length of the channel burst by the properties of the inner code.

The outer code of a compound-concatenated system is a burst correcting code such as those described in Chapters 6, 7, 8, and 9. The input to the outer decoder is simply the output of the "superchannel." Thus, the burst correcting capability of the outer code is determined by the maximum length of the burst of decoding errors. In addition, guard spaces at the outer decoder are very clean despite the fact that guard spaces in the channel may be quite

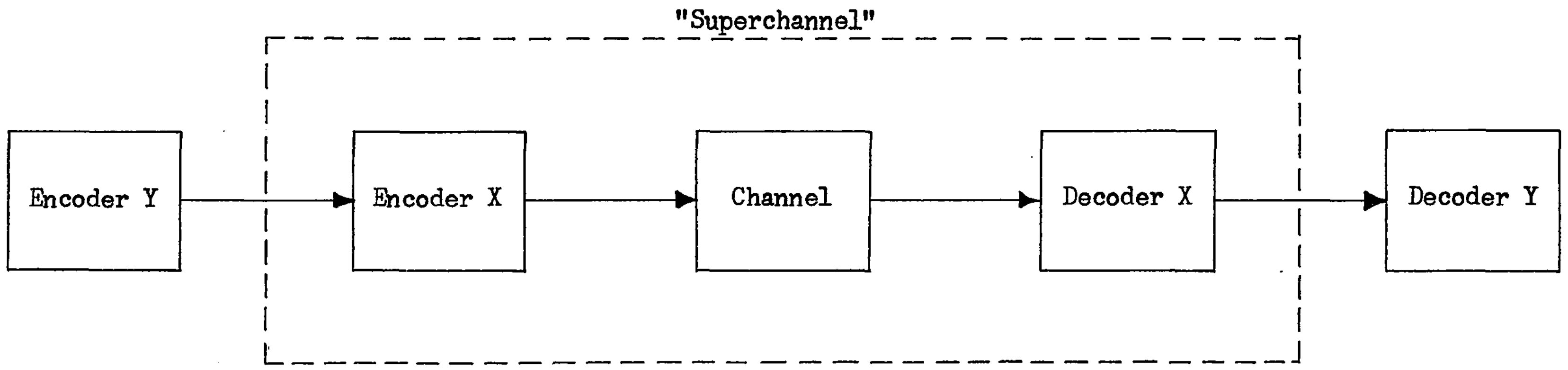


Figure 10.2.1 : Concatenated Codes.

noisy. Thus, the performance of the outer code is improved over that of the same code used alone on the compound channel.

The remainder of this chapter describes the structure, complexity, and performance of compound-concatenated systems for both classes of inner code. As far as is known, this material is new with this thesis.

10.3. Inner Block Code: Structure and Guard Space Requirement

The inner (n,k) parity-check code has minimum distance d and error correcting capability t. Since the sole purpose of the inner code is to correct channel guard space errors, we choose t a maximum,

$$t = \left\lfloor \frac{d-1}{2} \right\rfloor . \qquad (10.3.1)$$

On the compound channel, bursts very rarely exceed length B_c, where

$$B_c = fn \text{ , f an integer.} \qquad (10.3.2)$$

Since the inner decoder decodes each received block independently of all others, a burst of decoding errors cannot affect more blocks than the channel burst itself. Also, for each block, the output of the inner decoder consists only of the k decoded information bits in that block. It follows that the burst correcting capability B_m of the outer code need only be

$$B_m = fk = \frac{k}{n} B_c . \qquad (10.3.3)$$

In general, the outer code has burst correcting capability B_m relative to a clean guard space G_m, where

$$G_m = h\ B_m . \qquad (10.3.4)$$

The factor h is easily determined from the parameters of the outer code. From (10.3.3), the guard space requirement of the outer decoder is

$$G_m = h\ B_m = hfk. \qquad (10.3.5)$$

Although hfk is an integer, hf need not be. Thus, the guard space G_c in the channel spans $\lceil hf \rceil$ blocks,

$$G_c = \lceil hf \rceil n \approx h\, B_c. \tag{10.3.6}$$

It follows that

$$G_m \approx \frac{k}{n} G_c. \tag{10.3.7}$$

10.4. Inner Convolutional Code: Structure and Guard Space Requirement

The inner convolutional code may in theory have any rate $^k/_n$. In previous chapters, however, we have restricted our consideration to rate $\frac{1}{2}$ codes, the simplest though most important case, so we shall do so again. The code-generating polynomial $G_2(D)$ has maximum degree u with J nonzero coefficients, the code constraint length is n_A, the effective length is n_E, and the error correcting capability t is a maximum,

$$t = \left\lfloor {}^J/_2 \right\rfloor. \tag{10.4.1}$$

Channel bursts very rarely exceed length B_c, where

$$B_c = sn_A\ , \quad s \text{ an integer.} \tag{10.4.2}$$

During a channel burst, the inner feedback decoder will almost certainly propagate errors. By choosing a self-orthogonal convolutional code, this propagation is limited to wn_A digits, given by (4.3.27) and (4.3.28),

$$wn_A \leq (n_A - n)x + 2n_A, \tag{10.4.3}$$

where x is the least integer for which

$$(u - J + y)\left(\frac{J - y}{y}\right)^x \leq \binom{y+1}{2} + \binom{2y - J}{2}$$

and where $y = \left\lceil {}^J/_2 \right\rceil + 1$. This requires that the channel burst be followed by wn_A clean digits; i.e., if the first wn_A digits in the guard space following the burst are not all error-free, then error propagation is not guaranteed to be limited as in (10.4.3).

Because compound-concatenated systems are especially intended for compound channels with noisy guard spaces, the probability that all digits in a sequence of length wn_A will be error-free is not satisfactorily high. However, the probability P_R of an error-free run of length wn_A or greater anywhere in a sequence of length W, where $W \geq wn_A$, is an increasing function of W. Feller [46] shows that

$$P_R \approx 1 - \frac{K}{c^{W+1}} , \qquad (10.4.5)$$

where
$$K = \frac{1 - (1 - p_o)c}{(wn_A + 1 - wn_A c)p_o}, \qquad (10.4.6)$$

$$c \approx 1 + p_o(1 - p_o)^{wn_A} + (wn_A + 1)[p_o(1 - p_o)^{wn_A}]^2. \qquad (10.4.7)$$

Error propagation can be limited to W digits simply by allowing the run of wn_A error-free digits to occur anywhere within the W guard space digits immediately following the channel burst. The probability that this limit will not be exceeded is at least P_R, and P_R can be made arbitrarily close to one by increasing W. From (10.4.5),

$$W = \frac{\log(K) - \log(1 - P_R)}{\log(c)} - 1. \qquad (10.4.8)$$

Since the output of the inner decoder consists only of the decoded information digits, the burst correcting capability B_m of the outer code need only be

$$B_m = \tfrac{1}{2}(sn_A + W) = \tfrac{1}{2}(B_c + W). \qquad (10.4.9)$$

From (10.3.4), the guard space requirement G_m at the outer decoder is

$$G_m = h\, B_m = \frac{h}{2}(B_c + W). \qquad (10.4.10)$$

This corresponds to a guard space in the channel of length $h(B_c + W)$. Thus, the total channel guard space G_c is

$$G_c = W + h(B_c + W) = h\ B_c + (h + 1)W. \tag{10.4.11}$$

It follows that

$$G_m = \tfrac{1}{2}(G_c - W). \tag{10.4.12}$$

10.5. Comparative Complexities

For any particular compound channel on which bursts very rarely exceed length B_c, we consider three systems which can be used to control errors. One is a burst correcting code alone, another is a burst correcting code concatenated with a block code, and the last is a burst correcting code concatenated with a convolutional code. If the same class and rate of burst correcting code is used in all three cases, then the last two systems have lower rate than the first, but we would expect them to have better performance in return. The three systems can also be compared with respect to the complexity of their decoders. Using storage requirement N as the criterion, we shall make this comparison for interleaved block codes, diffuse codes, Gallager codes, and burst-trapping codes. The left superscripts "1" and "2" will refer to compound-concatenated systems with inner block codes and inner convolutional codes respectively.

10.5.a. Interleaved Block Codes

From (6.3.1), the storage requirement of an interleaved block code is

$$N = B_m + G_m. \tag{10.5.1}$$

When the code is used alone, we know from (6.2.1) and (6.2.5) that $B_m = B_c$ and $G_m = G_c$. It follows that

$$N = B_c + G_c. \tag{10.5.2}$$

Also, if $G_m = h\ B_m$, then $G_c = h\ B_c$.

When the code is concatenated with a block code, then from (10.3.3) and (10.3.7),

$$^{1}B_m = \frac{^{1}k}{^{1}n}(^{1}B_c) \;,\; ^{1}G_m \approx \frac{^{1}k}{^{1}n}(^{1}G_c). \tag{10.5.3}$$

Because the compound channel is the same,

$$^{1}B_c \approx B_c, \tag{10.5.4}$$

and because $^{1}G_c \approx h\ ^{1}B_c$ from (10.3.6),

$$^{1}G_c \approx G_c. \tag{10.5.5}$$

It follows that

$$^{1}N \approx \frac{^{1}k}{^{1}n}(B_c + G_c) = \frac{^{1}k}{^{1}n}(N). \tag{10.5.6}$$

When the code is concatenated with a convolutional code, then from (10.4.9) and (10.4.12),

$$^{2}B_m = \tfrac{1}{2}(^{2}B_c + W) \;,\; ^{2}G_m = \tfrac{1}{2}(^{2}G_c - W). \tag{10.5.7}$$

Again, the channel is the same so that

$$^{2}B_c \approx B_c. \tag{10.5.8}$$

However, $^{2}G_c = h\ ^{2}B_c + (h + 1)W$ from (10.4.11), so that

$$^{2}G_c \approx G_c + (h + 1)W. \tag{10.5.9}$$

It follows that

$$^{2}N \approx \tfrac{1}{2}[B_c + G_c + (h + 1)W] = \tfrac{1}{2}N + \frac{h + 1}{2} W. \tag{10.5.10}$$

Although the complexities of the inner block decoder and the inner feedback decoder are not included in the above calculations, (10.5.6) and (10.5.10) show that the complexity of a compound-concatenated system can very well be less than the complexity of the burst correcting code alone.

Because $B_m = B_c$ and $G_m = G_c$ for all the burst correcting codes that we consider, all expressions derived above relating these four parameters are valid throughout this section.

10.5.b. Diffuse Codes

From (7.4.1), the storage requirement of a diffuse code is

$$N = G_m + 2. \tag{10.5.11}$$

Thus, when the code is used alone,

$$N = G_c + 2. \tag{10.5.12}$$

When the code is concatenated with a block code,

$$^1N \approx \frac{^1k}{^1n}(G_c) + 2 = \frac{^1k}{^1n}(N - 2) + 2. \tag{10.5.13}$$

When the code is concatenated with a convolutional code,

$$^2N \approx \tfrac{1}{2}(G_c + hW) + 2 = \tfrac{1}{2}(N + hW) + 1. \tag{10.5.14}$$

10.5.c. Gallager Codes

From (8.4.1), the storage requirement of a Gallager code is

$$N = G_m - 2(y - 1), \tag{10.5.15}$$

where y is the number of consecutive zero-valued syndrome bits required to switch the decoder from the burst mode to the random mode. Thus, when the code is used alone,

$$N = G_c - 2(y - 1). \tag{10.5.16}$$

When the code is concatenated with a block code,

$$^1N \approx \frac{^1k}{^1n}(G_c) - 2(y - 1) = \frac{^1k}{^1n}[N + 2(y - 1)] - 2(y - 1). \tag{10.5.17}$$

When the code is concatenated with a convolutional code,

$$^{2}N \approx \tfrac{1}{2}(G_c + hW) - 2(y - 1) = \tfrac{1}{2}(N + hW) - (y - 1). \qquad (10.5.18)$$

10.5.d. Burst-Trapping Codes

From (9.4.1) and (9.4.2), the storage requirement of a burst-trapping code is

$$N = \frac{k+1}{n} G_m + n, \qquad (10.5.19)$$

where the code is obtained from an (n,k) systematic parity-check code. Thus, when the code is used alone,

$$N = \frac{k+1}{n} G_c + n. \qquad (10.5.20)$$

When the code is concatenated with a block code,

$$^{1}N \approx \frac{k+1}{n} \frac{^{1}k}{^{1}n}(G_c) + n = \frac{^{1}k}{^{1}n}(N - n) + n. \qquad (10.5.21)$$

When the code is concatenated with a convolutional code,

$$^{2}N \approx \frac{k+1}{n} \tfrac{1}{2}(G_c + hW) + n = \tfrac{1}{2}(N + n + \frac{k+1}{n} hW). \qquad (10.5.22)$$

10.6. Inner Block Code: Performance

The criterion of performance of a compound-concatenated system is the probability of a decoding error given that a channel burst has occurred, $P(E \mid \text{burst})$. From (8.5.4),

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)[1 - P(F)] + P(F). \qquad (10.6.1)$$

If the outer burst correcting code is non-adaptive, then the probability of failure $P(F)$ is zero. The probability of a decoding error given that the burst is detected, $P(E \mid \text{no } F)$, is primarily the probability of an error in the guard space of the outer decoder.

The compound channel model produces independent errors in the random mode with probability p_o and in the burst mode with probability q_o. In the random mode, the inner block decoder commits decoding errors with probability p_1, given approximately by (3.3.8),

$$p_1 \approx 1 - \sum_{j=0}^{t} \binom{n}{j} p_o^{\,j} (1 - p_o)^{n-j}. \qquad (10.6.2)$$

Similarly in the burst mode, the inner block decoder commits decoding errors with probability q_1, where

$$q_1 \approx 1 - \sum_{j=0}^{t} \binom{n}{j} q_o^{\,j} (1 - q_o)^{n-j}. \qquad (10.6.3)$$

Because t is chosen as large as possible and because p_o is small, it is generally true that

$$p_1 \ll p_o. \qquad (10.6.4)$$

However, q_1 may be either larger than or smaller than q_o. Although in actual fact a decoding error at the inner decoder results in a cluster of errors at its output, we may say roughly that on the average the "superchannel" produces independent errors in the random mode with probability p_1 and in the burst mode with probability q_1. These, then, are the parameters used in determining P(F) when the outer code is adaptive.

To find $P(E \mid \text{no } F)$, we must find the probability of an error in the decoder guard space, or, equivalently, the probability of a decoding error in the channel guard space. From (10.3.6), the channel guard space spans $\lceil hf \rceil$ blocks, each of which is decoded independently. Since the probability of a decoding error in any one block is p_1, $P(E \mid \text{no } F)$ is given by

$$P(E \mid \text{no } F) \leq 1 - (1 - p_1)^{\lceil hf \rceil}. \qquad (10.6.5)$$

The inequality exists because, in general, the outer code may be robust.

10.7. Inner Convolutional Code: Performance

In the random mode of the compound channel, the inner feedback decoder commits decoding errors with probability p_2, given approximately by (4.3.29),

$$p_2 \approx 1 - \sum_{j=0}^{t} \binom{n_E}{j} p_o^{\,j} (1 - p_o)^{n_E - j}. \qquad (10.7.1)$$

Similarly in the burst mode, the inner decoder commits decoding errors with probability q_2, where

$$q_2 \approx 1 - \sum_{j=0}^{t} \binom{n_E}{j} q_o^{\,j} (1 - q_o)^{n_E - j}. \qquad (10.7.2)$$

It is generally true that

$$p_2 \ll p_o, \qquad (10.7.3)$$

but q_2 may be larger than or smaller than q_o. We again consider that p_2 and q_2 approximately represent independent error probabilities of the "superchannel," so that these parameters are used in determining P(F) if the outer code is adaptive.

$P(E \mid \text{no } F)$ is again primarily the probability of an error in the decoder guard space, and we can say that

$$P(E \mid \text{no } F) \leq (1 - P_R) + P_e. \qquad (10.7.4)$$

From (10.4.5), P_R is the probability of a run of wn_A error-free digits in the first W digits of the channel guard space. Thus, $1 - P_R$ is the limiting probability that a channel burst is propagated by the feedback decoder beyond the burst correcting capability of the outer code. P_e is the probability of a decoding error in the last $h(B_c + W)$ digits of the channel guard space G_c, where from (10.4.11),

$$G_c = W + h(B_c + W). \qquad (10.7.5)$$

In decoding any one noise bit, the inner feedback decoder checks n_E distinct noise bits. In decoding a second noise bit, the effective length may contain some noise bits in common with the first. Thus, decoding is not independent in blocks of length n_E as it would be in a block decoder. However, because random errors in the channel guard space are relatively rare, a calculation for P_e based on the assumption that decoding is independent in blocks of n_E noise bits gives a result that is approximately correct.

If all the parameters of the system are known, it is easy to calculate the quantity g such that

$$g \triangleq \left\lceil \frac{h(B_c + W)}{n_E} \right\rceil . \tag{10.7.6}$$

The guard space can then be segmented into g distinct and independent blocks of length n_E. The probability of a decoding error in any one such block is p_2, so that P_e is given by

$$P_e \approx 1 - (1 - p_2)^g . \tag{10.7.7}$$

By increasing W sufficiently, P_R can be made so close to one that

$$1 - P_R \ll P_e . \tag{10.7.8}$$

In this case, then,

$$P(E \mid \text{no } F) \approx 1 - (1 - p_2)^g . \tag{10.7.9}$$

11. GUARD-SPACE-ADAPTIVE BURST-TRAPPING CODES

11.1. Introduction

In this chapter, we shall describe the structure, complexity, and performance of a burst correcting compound-concatenated system which we call a guard-space-adaptive burst-trapping code, or GSA code. GSA codes are adaptive. The random mode is provided by the inner code of the compound-concatenated system, a minimum distance decodable (n_i,k_i) parity-check code. The burst mode is provided by the outer code of the system, a modified burst-trapping code obtained from a systematic (n_o,k_o) parity-check code.

GSA codes are similar to Tong's burst-trapping codes, having an adaptive guard space requirement proportional to the length of the actual burst. However, the guard space of a GSA code is immediately adjacent to the burst, which is not always the case for burst-trapping codes.

Section 11.2. combines the structure of a compound-concatenated system, the general principle of a burst-trapping code, and the requirements for an adaptive guard space adjacent to the burst in order to determine the structure of a GSA code. Section 11.3. defines an optimality criterion for GSA codes, called minimum effective length, and lists a number of optimal codes for various rates and burst correcting capabilities. Section 11.4. shows that by interleaving, GSA codes may have any desired burst correcting capability, and then outlines the general properties of interleaved GSA codes. Section 11.5. employs an example to demonstrate the decoding procedures of a GSA code and to determine the decoder implementation. Section 11.6. discusses the decoder complexity, and Section 11.7. derives expressions for the performance of GSA codes.

As far as is known, the principles of GSA codes and all material in this chapter are new with this thesis.

11.2. Structure and Guard Space Requirement

The inner (n_i,k_i) parity-check code of the GSA code has minimum distance d and error correcting capability t. Since it is also the random mode of the GSA code, it must have some error detecting capability. Thus, t is chosen less than maximum,

$$t < \left\lfloor \frac{d-1}{2} \right\rfloor . \qquad (11.2.1)$$

The outer code is a modified burst-trapping code based upon a systematic (n_o,k_o) parity-check code for which

$$\frac{k_o}{n_o} = \frac{x-1}{x} \text{ , x an integer,} \qquad (11.2.2)$$

and
$$k_o = (x-1)(n_o - k_o) \text{ , } n_o = x(n_o - k_o). \qquad (11.2.3)$$

Each codeword $\underline{t}^j$ of the parity-check code may be divided into x - 1 information sub-blocks $\underline{I}_i^j$, i = 1, 2, ..., x - 1, and one parity sub-block $\underline{P}^j$, where each sub-block has length $n_o - k_o$,

$$\underline{t}^j = (\underline{m}^j, \underline{P}^j) = (\underline{I}_1^j, \underline{I}_2^j, \ldots, \underline{I}_{x-1}^j, \underline{P}^j). \qquad (11.2.4)$$

Since the outer code provides only the burst mode of the GSA code, it need have no random error correcting capability or error detecting capability. Thus, it can be chosen to be the trivial code for which $\underline{P}^j$ is always the null sequence,

$$\underline{P}^j = \underline{0}. \qquad (11.2.5)$$

Like a burst-trapping code, the outer code is obtained from the (n_o,k_o) parity-check code by forming codewords $\underline{T}^j$ such that

$$\underline{T}^j = (\underline{m}^j, \underline{Q}^j) = (\underline{I}_1^j, \ldots, \underline{I}_{x-1}^j, \underline{Q}^j), \qquad (11.2.6)$$

where
$$\underline{Q}^j = \underline{P}^j + f(\underline{I}) = f(\underline{I}). \qquad (11.2.7)$$

From (9.2.12), for a Tong burst-trapping code,

$$f(\underline{I}) = \underline{I}_1^{j-v} + \underline{I}_2^{j-2v} + \ldots + \underline{I}_{x-1}^{j-(x-1)v}. \tag{11.2.8}$$

For a GSA code, $f(\underline{I})$ is some mod-2 sum of information sub-blocks which imparts burst correcting capability to the outer code in such a way that the guard space requirement is both adaptive and immediately adjacent to the burst.

Bursts on the compound channel very rarely exceed length B_c such that

$$B_c = fn_i \ , \ f \text{ an integer.} \tag{11.2.9}$$

For each block at the inner decoder, the output consists of the k_i decoded information bits in that block. Since each block is decoded independently, a burst cannot be propagated. Thus, the burst correcting capability B_m of the outer code need only be

$$B_m = fk_i = \frac{k_i}{n_i} B_c. \tag{11.2.10}$$

In addition, since the block length at the outer decoder is n_o,

$$B_m = fk_i = bn_o, \tag{11.2.11}$$

where b is an integer such that f/b is an integer.

If a burst at the outer decoder affects y blocks, y = 1, 2, ..., b, then there are y(x - 1) unreliable information sub-blocks, which we call the y(x - 1) unknowns. The sequences $\underline{Q}^j$ or $f(\underline{I})$ in the codewords of the outer code are chosen in such a way that the set of y(x - 1) sequences immediately following the burst constitutes a set of y(x - 1) linearly independent equations in the y(x - 1) unknowns. In this way, for a burst of length yn_o, the guard space G_{oy} at the outer decoder is

$$G_{oy} = (x - 1)yn_o = (x - 1)y \frac{f}{b} k_i \ , \ y = 1, 2, \ldots, b. \tag{11.2.12}$$

This corresponds to an adaptive guard space G_{iy} spanning (x - 1)yf/b blocks at the inner decoder,

$$G_{iy} = (x - 1)y \frac{f}{b} n_i \text{ , } y = 1, 2, \ldots, b. \qquad (11.2.13)$$

In the case where $y = b$,

$$G_{oy} = G_m = (x - 1)bn_o = (x - 1) B_m, \qquad (11.2.14)$$

$$G_{iy} = G_c = (x - 1)fn_i = (x - 1) B_c. \qquad (11.2.15)$$

To demonstrate how $f(\underline{I})$ is chosen, it is convenient to use a numerical example.

Example 11.2.1.

The parameters of the system are $x = 3$, $b = 3$. The outer code therefore corrects all detected bursts of length n_o, $2n_o$, and $3n_o$.

For bursts of length n_o, a suitable burst correcting code is a Tong burst-trapping code which is time-diversified to degree $v = 1$. From (11.2.8),

$$\underline{Q}^j = f(\underline{I}) = \underline{I}_1^{j-1} + \underline{I}_2^{j-2}. \qquad (11.2.16)$$

If we are attempting to correct a burst in the jth block, then the guard space spans the succeeding $y(x - 1) = 2$ blocks, for which

$$\underline{Q}^{j+1} = \underline{I}_1^j + \underline{I}_2^{j-1},$$

$$\underline{Q}^{j+2} = \underline{I}_1^{j+1} + \underline{I}_2^j. \qquad (11.2.17)$$

The two unknowns are recovered from two linearly independent equations,

$$\underline{I}_1^j = \underline{A} = \underline{Q}^{j+1} + \underline{I}_2^{j-1},$$

$$\underline{I}_2^j = \underline{B} = \underline{Q}^{j+2} + \underline{I}_1^{j+1}. \qquad (11.2.18)$$

To obtain a code which can correct bursts of length n_o and $2n_o$, we append $x - 1$ terms; i.e., information sub-blocks, to $f(\underline{I})$ so that

$$\underline{Q}^j = f(\underline{I}) = \underline{I}_1^{j-1} + \underline{I}_2^{j-2} + \underline{I}_c^{j-3} + \underline{I}_d^{j-4}. \qquad (11.2.19)$$

The variables c and d are to be determined. They may assume any of the values 0, 1, ..., x - 1, where we say that

$$\underline{I}_0^j = \underline{0} \text{ for all } j. \qquad (11.2.20)$$

If a burst affects the blocks j and j + 1, then the guard space spans the y(x - 1) = 4 succeeding blocks j + 2, ..., j + 5. The four unknowns are recovered from four linearly independent equations,

$$\begin{aligned}
\underline{I}_1^{j+1} + \underline{I}_2^j &= \underline{A} = \underline{Q}^{j+2} + \underline{I}_c^{j-1} + \underline{I}_d^{j-2}, \\
\underline{I}_2^{j+1} + \underline{I}_c^j &= \underline{B} = \underline{Q}^{j+3} + \underline{I}_1^{j+2} + \underline{I}_d^{j-1}, \\
\underline{I}_c^{j+1} + \underline{I}_d^j &= \underline{C} = \underline{Q}^{j+4} + \underline{I}_1^{j+3} + \underline{I}_2^{j+2}, \\
\underline{I}_d^{j+1} \qquad &= \underline{D} = \underline{Q}^{j+5} + \underline{I}_1^{j+4} + \underline{I}_2^{j+3} + \underline{I}_c^{j+2}. \qquad (11.2.21)
\end{aligned}$$

The set of equations (11.2.21) are linearly independent if c = 0, d = 1, so that

$$f(\underline{I}) = \underline{I}_1^{j-1} + \underline{I}_2^{j-2} + \underline{I}_1^{j-4}. \qquad (11.2.22)$$

If bursts can be of length n_o, $2n_o$, or $3n_o$, we again append x - 1 terms to $f(\underline{I})$. Thus,

$$\underline{Q}^j = f(\underline{I}) = \underline{I}_1^{j-1} + \underline{I}_2^{j-2} + \underline{I}_1^{j-4} + \underline{I}_c^{j-5} + \underline{I}_d^{j-6}. \qquad (11.2.23)$$

Solving for c and d such that a set of y(x - 1) = 6 equations in the guard space are linearly independent, we obtain c = 0, d = 2. Therefore,

$$f(\underline{I}) = \underline{I}_1^{j-1} + \underline{I}_2^{j-2} + \underline{I}_1^{j-4} + \underline{I}_2^{j-6}. \qquad (11.2.24)$$

11.3. Optimal GSA Codes

To find the function $f(\underline{I})$ which defines the outer code of a GSA code, we must solve for the x - 1 subscripts c, d, ... a total of b times, where

$$B_m = bn_o. \qquad (11.3.1)$$

Each time we try to solve for these subscripts, there may be no solution, a unique solution, or several solutions. If at any step there is no solution, then a GSA code cannot be found. If several solutions exist, one or more of them may be optimal.

The number of terms in the function $f(\underline{I})$ is called the effective length l_E of the GSA code. We say that the code is optimal if it has minimum effective length for, as we shall see in Section 11.6., minimizing the effective length minimizes the decoder complexity.

To determine the minimum effective length of a GSA code, we observe that $f(\underline{I})$ has the general form

$$f(\underline{I}) = \sum_{y=1}^{b} [\underline{I}^{j-(y-1)(x-1)-1} + \underline{I}^{j-(y-1)(x-1)-2} + \ldots + \underline{I}^{j-y(x-1)}]. \qquad (11.3.2)$$

We say that the x - 1 terms $\underline{I}^{j-(y-1)(x-1)-1}, \ldots, \underline{I}^{j-y(x-1)}$ constitute the y^{th} partition of the function $f(\underline{I})$, y = 1, 2, ..., b. Some of these x - 1 terms may be of the form $\underline{I}_0^j$; i.e., may be null sequences, and we minimize l_E if we maximize the number of null sequences in $f(\underline{I})$. In order that the set of $b(x - 1)$ equations in $b(x - 1)$ unknowns be linearly independent, and in order that all detected bursts of length bn_o or less be corrected, the following conditions on $f(\underline{I})$ must be met:

(a) All x - 1 terms in the first partition must have different subscripts and no term may be a null sequence.

(b) At least one term in each of the remaining b - 1 partitions may not be a null sequence.

(c) No term in the y^{th} partition may be a null sequence in a consecutive run of y or more null sequences.

We say arbitrarily that the last term, $\underline{I}^{j-y(x-1)}$, in the y^{th} partition may not be a null sequence. Then, if $y > x - 2$, the remaining x - 2 terms in the partition may all be null sequences. If $y \leq x - 2$, then at least $\left\lfloor \frac{x-2}{y} \right\rfloor$ of these $x - 2$ terms may not be null sequences. It follows that

$$l_E \geq (x-1) + (b-1) + \sum_{y=2}^{K} \left\lfloor \frac{x-2}{y} \right\rfloor$$

$$= x + b - 2 + \sum_{y=2}^{K} \left\lfloor \frac{x-2}{y} \right\rfloor , \; K = \min(x-2,\, b). \qquad (11.3.3)$$

In Table 11.3.1, we list all the optimal GSA codes found by the method of Example 11.2.1. for different values of x and b. For convenience, we have employed the notation

$$(i,u) = \underline{I}_i^{j-u}. \qquad (11.3.4)$$

Note that no GSA codes were found to exist for $b > 4$.

11.4. Interleaved GSA Codes

GSA codes may be interleaved to any degree r so that their burst correcting capability B_m may be arbitrarily long,

$$B_m = fk_i = rbn_o \, , \; b = 1, \ldots, 4 \, , \; r \text{ an integer.} \qquad (11.4.1)$$

GSA codes can correct bursts of b different lengths, n_o, $2n_o$, ..., bn_o, according to b different algorithms. Interleaved GSA codes are therefore restricted to b decoding algorithms, so that all bursts of length $r(y-1)n_o + n_o$, ..., ryn_o are decoded according to the algorithm for bursts of length yn_o, $y = 1, 2, \ldots, b$. The decoder guard space requirement for any such burst is, from (11.2.12),

$$G_{oy} = (x-1)ryn_o = (x-1)y \frac{f}{b} k_i \, , \; y = 1, 2, \ldots, b. \qquad (11.4.2)$$

The function $f(\underline{I})$ for an interleaved GSA code has the general form

$$f(\underline{I}) = \sum_{y=1}^{b} [\underline{I}^{j-(y-1)(x-1)r-r} + \ldots + \underline{I}^{j-y(x-1)r}]. \qquad (11.4.3)$$

$\frac{x-1}{x}$	b	l_E	f($\underline{I}$)
2/3	1	2	(1,1) + (2,2)
2/3	2	3	(1,1) + (2,2) + (1,4)
2/3	3	4	(1,1) + (2,2) + (1,4) + (2,6)
2/3	4	5	(1,1) + (2,2) + (1,4) + (2,6) + (2,8)
3/4	1	3	(1,1) + (2,2) + (3,3)
3/4	2	5	(1,1) + (2,2) + (3,3) + (3,5) + (1,6)
			(1,1) + (2,2) + (3,3) + (1,4) + (3,6)
			(1,1) + (2,2) + (3,3) + (2,4) + (1,6)
			(1,1) + (2,2) + (3,3) + (3,4) + (1,6)
3/4	3	6	(1,1) + (2,2) + (3,3) + (3,5) + (1,6) + (2,9)
			(1,1) + (2,2) + (3,3) + (2,4) + (1,6) + (2,9)
			(1,1) + (2,2) + (3,3) + (3,4) + (1,6) + (2,9)
3/4	4	7	(1,1) + (2,2) + (3,3) + (3,5) + (1,6) + (2,9) + (1,12)
			(1,1) + (2,2) + (3,3) + (3,4) + (1,6) + (2,9) + (3,12)
4/5	1	4	(1,1) + (2,2) + (3,3) + (4,4)
4/5	2	6	(1,1) + (2,2) + (3,3) + (4,4) + (1,6) + (3,8)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8)
4/5	3	8	(1,1) + (2,2) + (3,3) + (4,4) + (1,6) + (3,8) + (1,11) + (2,12)
			(1,1) + (2,2) + (3,3) + (4,4) + (1,6) + (3,8) + (2,11) + (1,12)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (1,11) + (2,12)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (2,11) + (4,12)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (3,11) + (2,12)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (4,10) + (2,12)

Table 11.3.1 (continued on page 114)

$\frac{x-1}{x}$	b	l_E	$f(\underline{I})$
4/5	3	8	(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (1,9) + (2,12)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (3,9) + (2,12)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (4,9) + (2,12)
4/5	4	9	(1,1) + (2,2) + (3,3) + (4,4) + (1,6) + (3,8) + (1,11) + (2,12) + (1,16)
			(1,1) + (2,2) + (3,3) + (4,4) + (1,6) + (3,8) + (2,11) + (1,12) + (2,16)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (1,11) + (2,12) + (3,16)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (3,11) + (2,12) + (3,16)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (4,10) + (2,12) + (3,16)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (1,9) + (2,12) + (3,16)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (1,9) + (2,12) + (4,16)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (3,9) + (2,12) + (3,16)
			(1,1) + (2,2) + (3,3) + (4,4) + (3,6) + (1,8) + (4,9) + (2,12) + (3,16)

Table 11.3.1

11.5. Decoding Procedures

For every block of length n_i in the compound channel, the inner minimum distance decoder of the GSA code passes on k_i decoded information bits to the outer decoder. These k_i bits contain errors either if a burst was detected or if a decoding error was committed in attempting to make a correction. From (11.4.1),

$$n_o = \frac{f}{rb} k_i \, , \; \frac{f}{rb} \text{ an integer,} \tag{11.5.1}$$

so f/rb groups of these k_i decoded bits constitute one block of length n_o at the outer decoder. The outer code has codewords $\underline{T}^j$, so that the received block is denoted $\underline{R}^j$,

$$\underline{R}^j = \underline{T}^j + \underline{e}^j, \tag{11.5.2}$$

where $\underline{e}^j$ is the error pattern passed on from the inner decoder.

To demonstrate fully the decoding procedures of the outer code and to show how the decoder is implemented, it is convenient to use a numerical example.

Example 11.5.1.

The parameters of the outer code are x = 3, b = 3, r = 1. From Table 11.3.1, an optimal code is given by

$$f(\underline{I}) = \underline{Q}^j = \underline{I}_1^{j-1} + \underline{I}_2^{j-2} + \underline{I}_1^{j-4} + \underline{I}_2^{j-6}, \tag{11.5.3}$$

which is the code derived in Example 11.2.1.. This code has codewords of the form

$$\underline{T}^j = (\underline{I}_1^j, \underline{I}_2^j, \underline{Q}^j). \tag{11,5,4}$$

Assume that the block $\underline{R}^j$ has just been received and that all previous decoding has been correct.

The outer decoder, Fig. 11.5.1, contains four shift registers. The first is used to store the received block $\underline{R}^j$. The second, called the storage register, is used to store the decoded message sequences of the previous $r(bx - 1)$ blocks. This means that $r(bx - 1)(x - 1) = 16$ information sub-blocks are stored, these sub-block locations being numbered in the figure. The third shift register, called the error-check register, is used to label the received block and the $r(bx - 1)$ stored message sequences as either presumably correct or definitely unreliable by means of 0's and 1's respectively. A digit is placed in the first stage by the inner decoder according to whether or not a burst was detected somewhere in the block $\underline{R}^j$. The state of the error-check register, whose $r(bx - 1) + 1 = 9$ stages are numbered in the figure, controls the mode of the decoder. Because previous decoding has been correct, the last eight stages of the register initially contain 0's. The fourth shift register, called the computation register, is used to store the mod-2 sums $\underline{A}$, $\underline{B}$, $\underline{C}$, ... of (11.2.18) and (11.2.21). These are the sums of the known quantities in the set of linearly independent simultaneous equations from which the unknown information sub-blocks may be recovered. The computation register therefore contains $rb(x - 1)$ sub-block locations. Initially, it is either empty or contains irrelevant data.

When the error-check register is in the state of Fig. 11.5.1, the storage register does not contain a detected burst. In this case, the decoder shifts $\underline{I}_1^{j-8}$ and $\underline{I}_2^{j-8}$ out of the storage register to the source decoder, shifts $\underline{I}_1^j$ and $\underline{I}_2^j$ into the storage register, right-shifts the error-check register one stage, right-shifts the computation register one sub-block, and accepts the next block $\underline{R}^{j+1}$.

We shall now consider the decoder behaviour for detected bursts of length ryn_o followed by clean guard spaces of length $(x - 1)ryn_o$, $y = 1, 2, 3$.

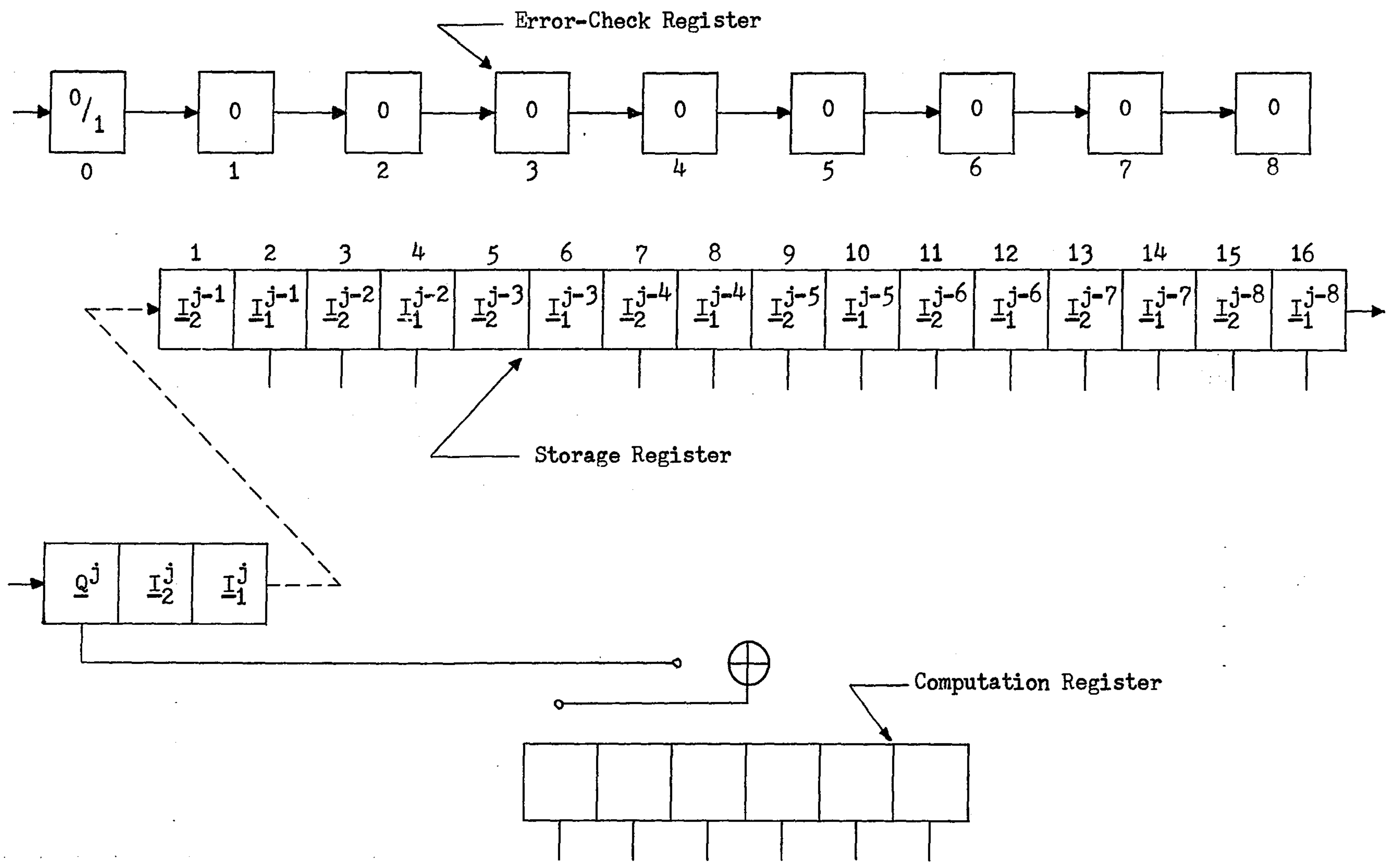


Figure 11.5.1 : Outer Decoder of a GSA Code.

First, assume $\underline{R}^{j}$ is affected by a burst and $\underline{R}^{j+1}$, $\underline{R}^{j+2}$ are clean. When $\underline{R}^{j+1}$ is received, the error-check register state (0, 1, 0, ..., 0), Fig. 11.5.2, shows that the burst spans only one block. Analogous to (11.5.3), the decoder forms the sum

$$\underline{I}_1^{j} = \underline{A} = \underline{Q}^{j+1} + \underline{I}_2^{j-1} + \underline{I}_1^{j-3} + \underline{I}_2^{j-5}, \qquad (11.5.5)$$

and stores it in the computation register. Similarly, when $\underline{R}^{j+2}$ is received, Fig. 11.5.3, the decoder forms the sum

$$\underline{I}_2^{j} = \underline{B} = \underline{Q}^{j+2} + \underline{I}_1^{j+1} + \underline{I}_1^{j-2} + \underline{I}_2^{j-4}. \qquad (11.5.6)$$

The decoder then solves the set of simultaneous equations as in Fig. 11.5.4 and resets the error-check register to zero.

Assume now that $\underline{R}^{j}$ and $\underline{R}^{j+1}$ are affected by a burst and $\underline{R}^{j+2}$, ..., $\underline{R}^{j+5}$ are clean. When $\underline{R}^{j+2}$ is received, the error-check register state (0, 1, 1, 0, ..., 0) shows that the burst spans two blocks. Each equation therefore contains two unknowns and four sums, $\underline{A}$, $\underline{B}$, $\underline{C}$, $\underline{D}$, must be formed. Figs. 11.5.5 and 11.5.6 show how the information sub-blocks are recovered.

Finally, assume that $\underline{R}^{j}$, $\underline{R}^{j+1}$, and $\underline{R}^{j+2}$ are affected by a burst and $\underline{R}^{j+3}$, ..., $\underline{R}^{j+8}$ are clean. When $\underline{R}^{j+3}$ is received, the error-check register state (0, 1, 1, 1, 0, ..., 0) shows that the burst spans three blocks. Each equation therefore contains three unknowns and six sums, $\underline{A}$, $\underline{B}$, $\underline{C}$, $\underline{D}$, $\underline{E}$, $\underline{F}$, must be formed. Figs. 11.5.7 and 11.5.8 show how the information sub-blocks are recovered.

11.6. Complexity

The three parameters of complexity, N, N_T, and N_A, may be determined from Figs. 11.5.1 to 11.5.8. In the outer decoder, the received block occupies n_o stages of shift register, the storage register contains $r(bx - 1)(x - 1)(n_o - k_o)$

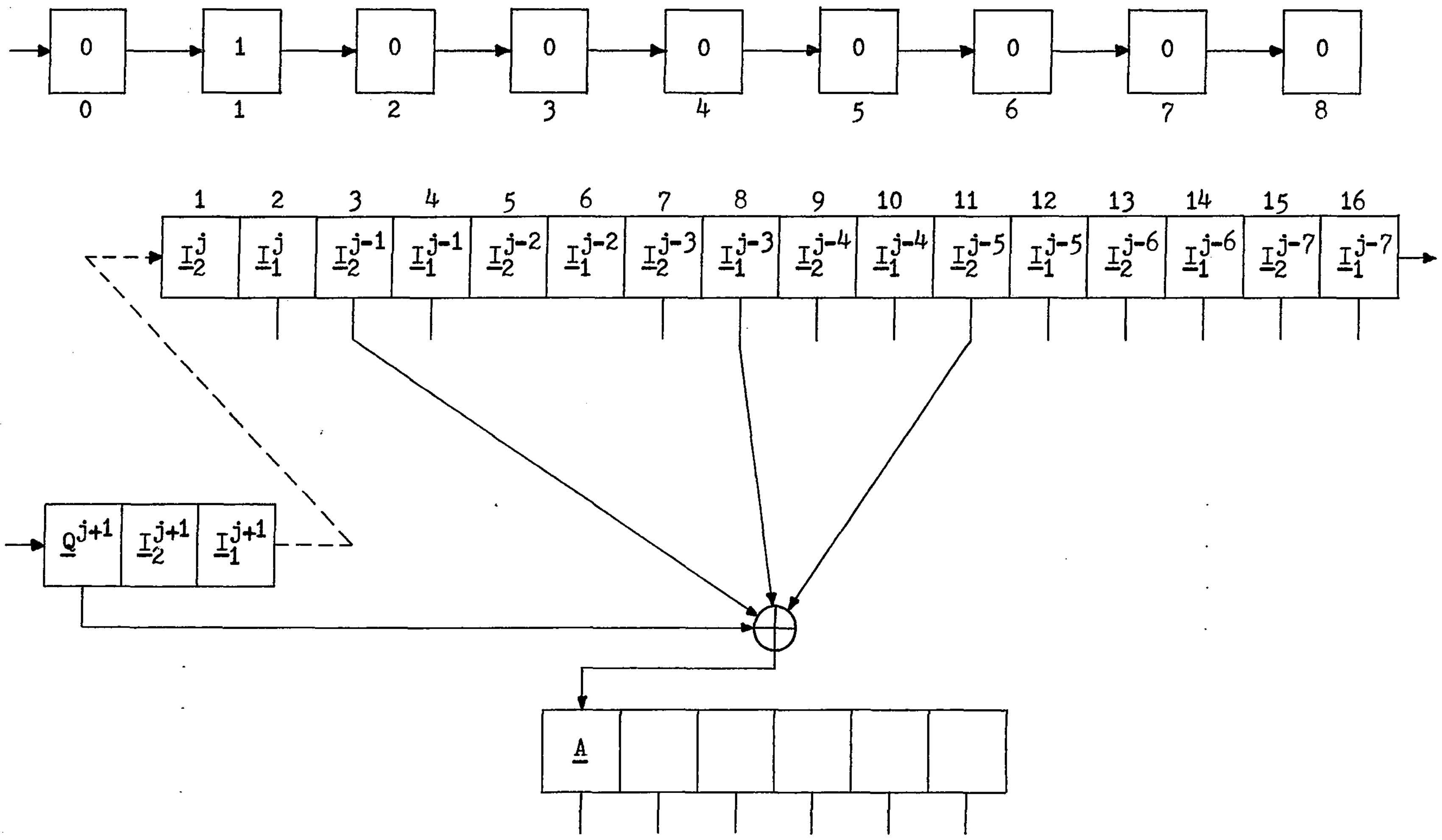

0 1 0 0 0 0 0 0 0
0 1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
$\underline{I}_2^j$ $\underline{I}_1^j$ $\underline{I}_2^{j-1}$ $\underline{I}_1^{j-1}$ $\underline{I}_2^{j-2}$ $\underline{I}_1^{j-2}$ $\underline{I}_2^{j-3}$ $\underline{I}_1^{j-3}$ $\underline{I}_2^{j-4}$ $\underline{I}_1^{j-4}$ $\underline{I}_2^{j-5}$ $\underline{I}_1^{j-5}$ $\underline{I}_2^{j-6}$ $\underline{I}_1^{j-6}$ $\underline{I}_2^{j-7}$ $\underline{I}_1^{j-7}$
$\underline{Q}^{j+1}$ $\underline{I}_2^{j+1}$ $\underline{I}_1^{j+1}$
$\underline{A}$

Figure 11.5.2 : Correcting a Burst in One Block.

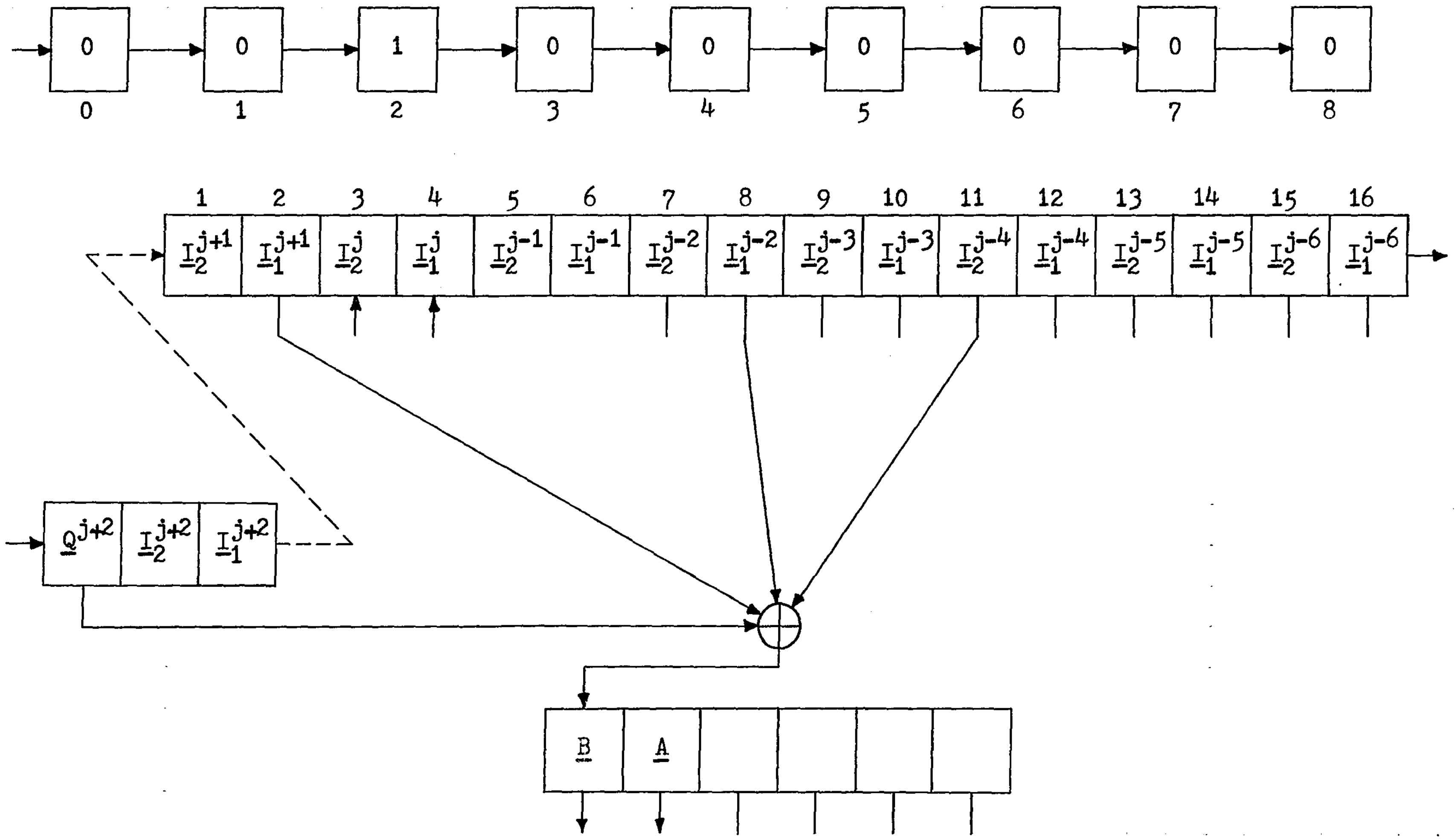
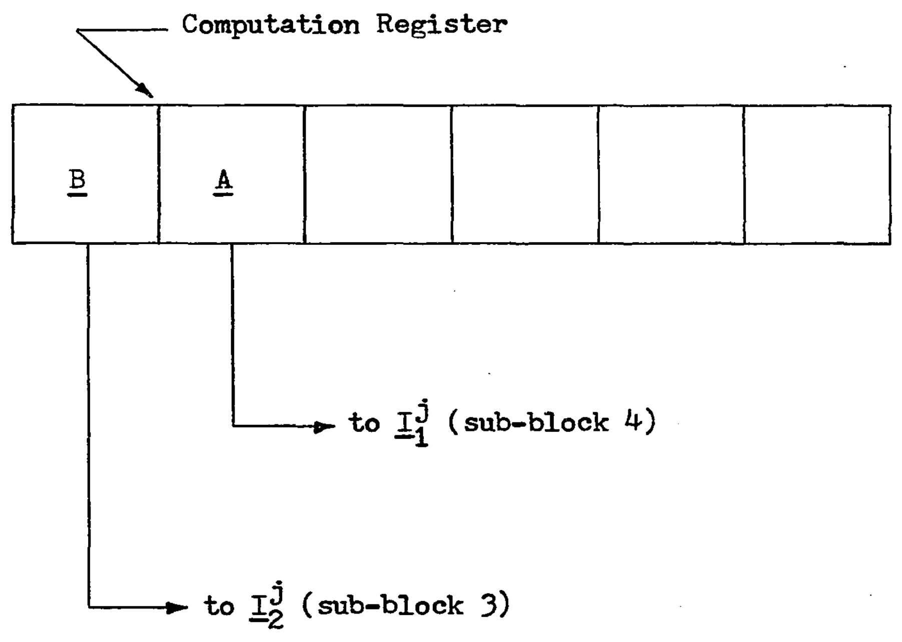


Figure 11.5.3 : Correcting a Burst in One Block.



Computation Register
B
A
to $\underline{I}_1^j$ (sub-block 4)
to $\underline{I}_2^j$ (sub-block 3)

Figure 11.5.4 : Computation Register Connections for the Correction of a Burst in One Block.

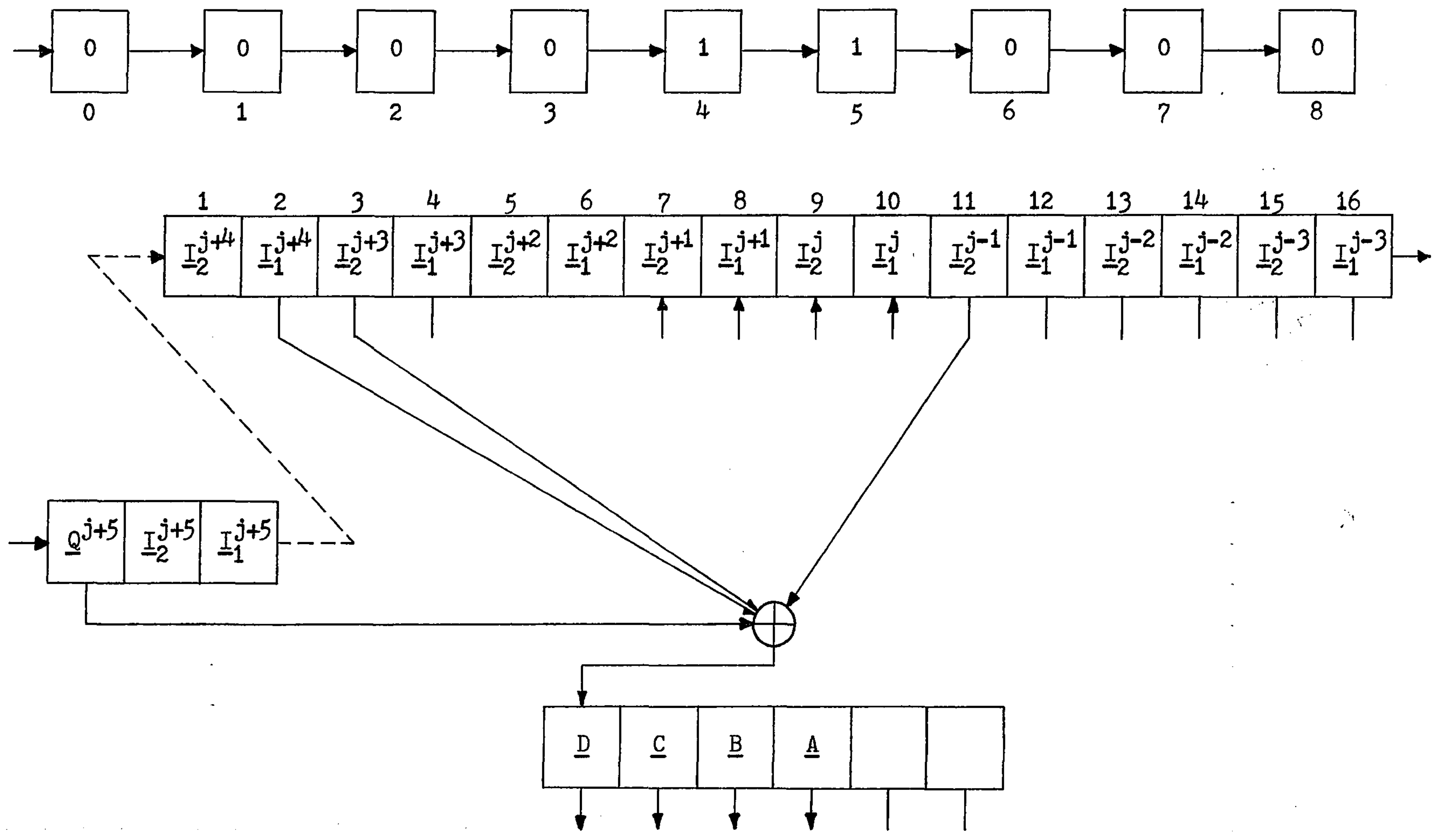

0
0
0
0
1
1
0
0
0
0 1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
I_2^{j+4} I_1^{j+4} I_2^{j+3} I_1^{j+3} I_2^{j+2} I_1^{j+2} I_2^{j+1} I_1^{j+1} I_2^{j} I_1^{j} I_2^{j-1} I_1^{j-1} I_2^{j-2} I_1^{j-2} I_2^{j-3} I_1^{j-3}
Q^{j+5} I_2^{j+5} I_1^{j+5}
D C B A

Figure 11.5.5 : Correcting a Burst in Two Blocks.

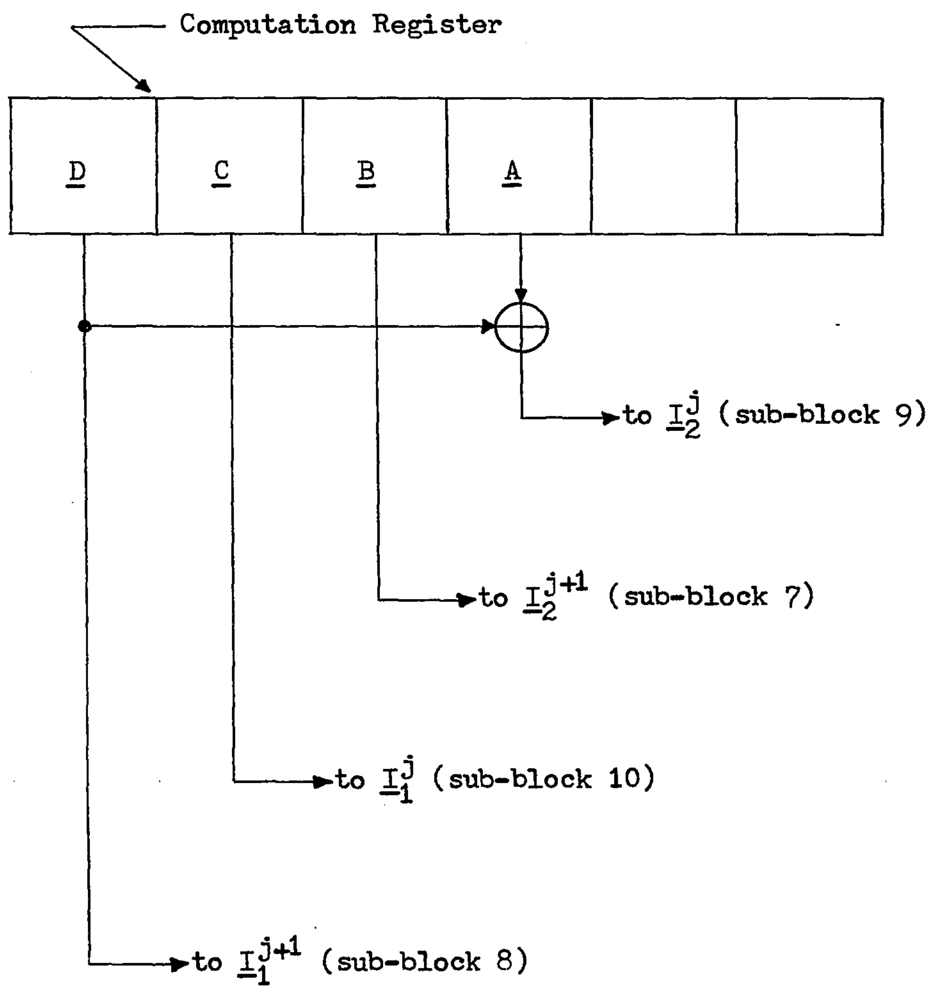
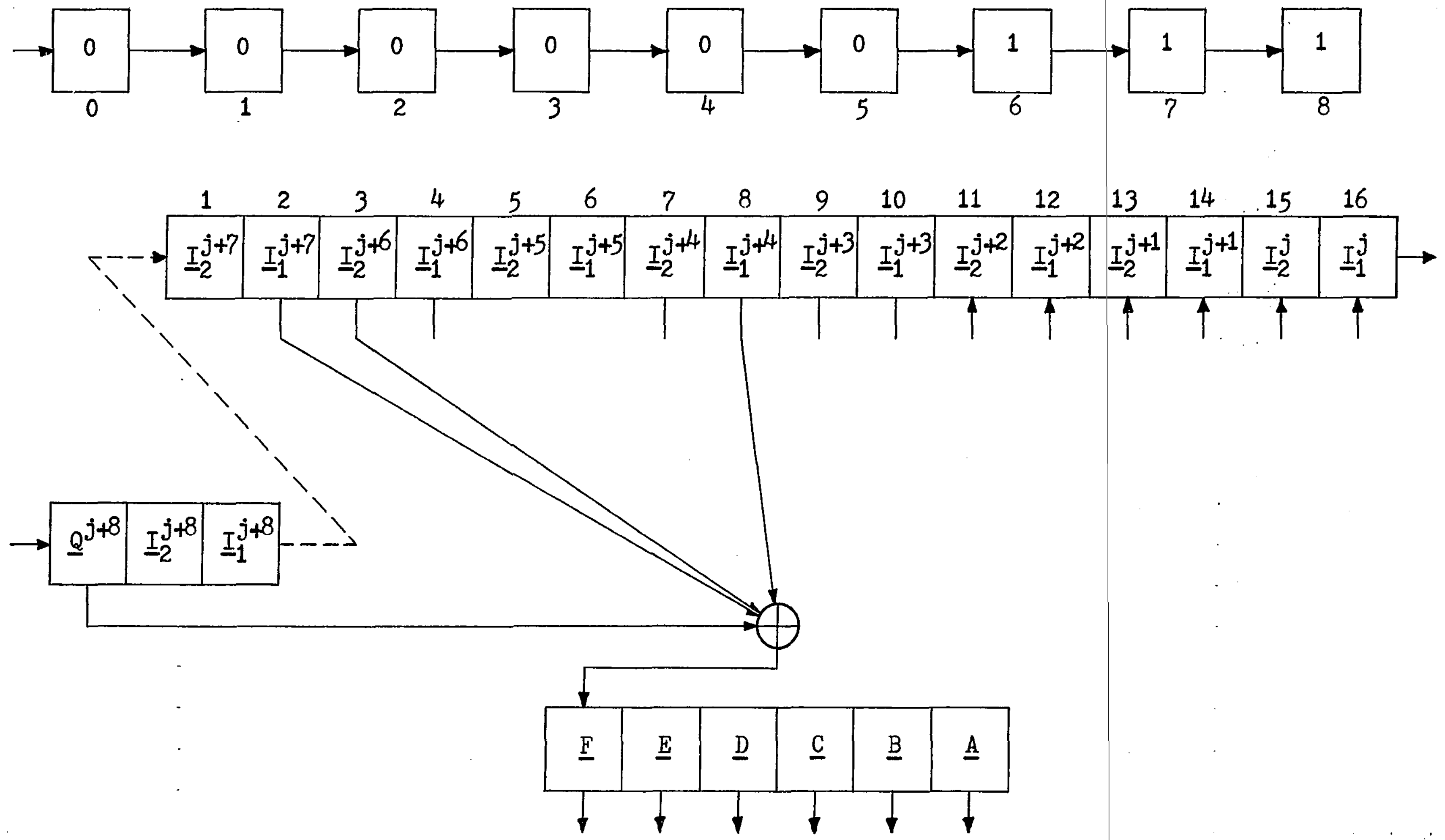

Computation Register
$\underline{D}$
$\underline{C}$
$\underline{B}$
$\underline{A}$
to $\underline{I}_2^j$ (sub-block 9)
to $\underline{I}_2^{j+1}$ (sub-block 7)
to $\underline{I}_1^j$ (sub-block 10)
to $\underline{I}_1^{j+1}$ (sub-block 8)

Figure 11.5.6 : Computation Register Connections for the Correction of a Burst in Two Blocks.



0 0 0 0 0 0 1 1 1
0 1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
I_2^{j+7} I_1^{j+7} I_2^{j+6} I_1^{j+6} I_2^{j+5} I_1^{j+5} I_2^{j+4} I_1^{j+4} I_2^{j+3} I_1^{j+3} I_2^{j+2} I_1^{j+2} I_2^{j+1} I_1^{j+1} I_2^{j} I_1^{j}
Q^{j+8} I_2^{j+8} I_1^{j+8}
F E D C B A

Figure 11.5.7 : Correcting a Burst in Three Blocks.

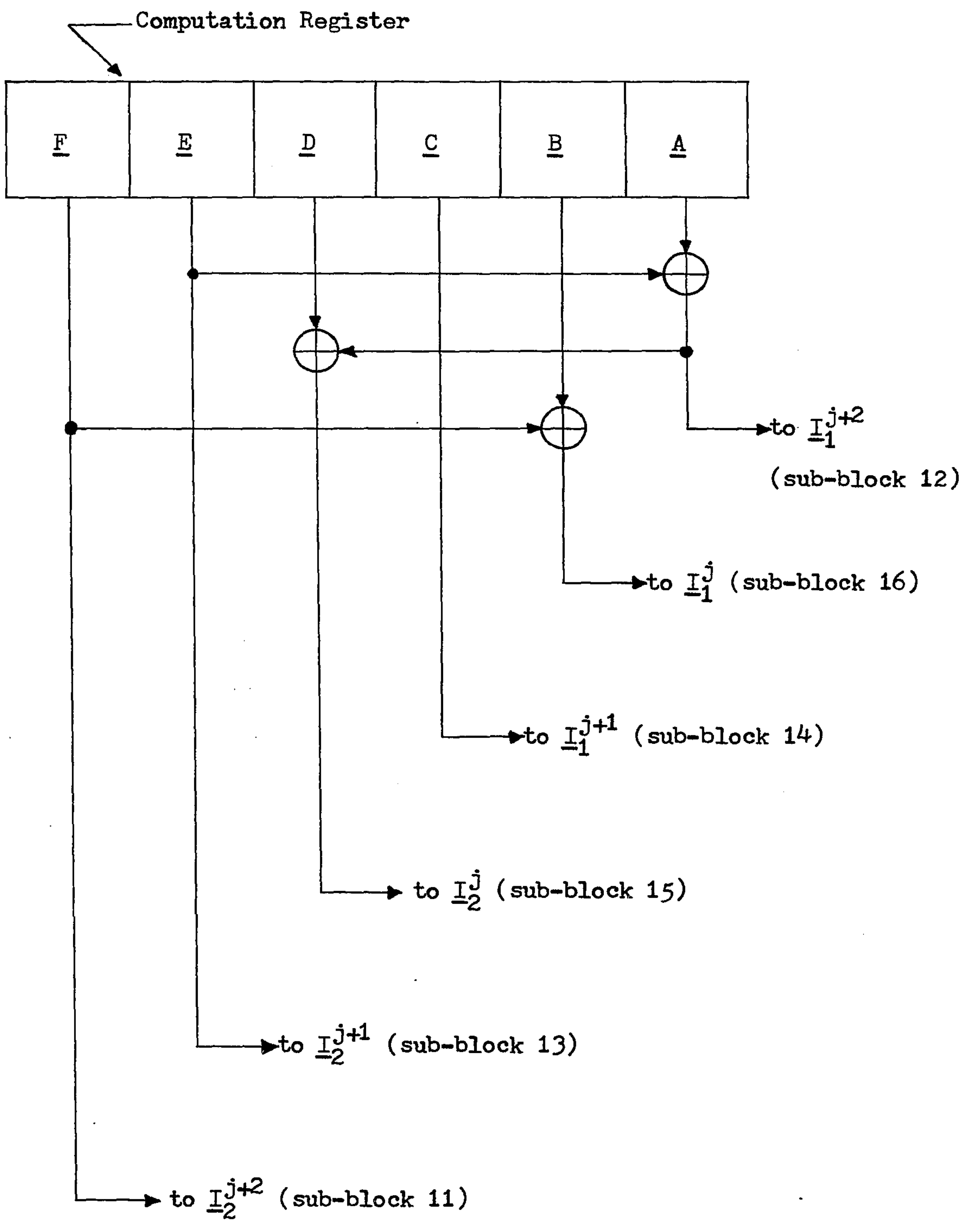

Computation Register
F
E
D
C
B
A
to $\underline{I}_1^{j+2}$ (sub-block 12)
to $\underline{I}_1^{j}$ (sub-block 16)
to $\underline{I}_1^{j+1}$ (sub-block 14)
to $\underline{I}_2^{j}$ (sub-block 15)
to $\underline{I}_2^{j+1}$ (sub-block 13)
to $\underline{I}_2^{j+2}$ (sub-block 11)

Figure 11.5.8 : Computation Register Connections for the Correction of a Burst in Three Blocks.

or $r(bx - 1)k_o$ stages, the error-check register contains $r(bx - 1) + 1$ stages, and the computation register contains $rb(x - 1)(n_o - k_o)$ or rbk_o stages. The storage requirement N is therefore gicen by

$$N = r(k_o + 1)(bx - 1) + rbk_o + n_o + 1. \tag{11.6.1}$$

From (11.2.14) and (11.4.2), it follows that

$$N/G_m = \frac{r(k_o + 1)(bx - 1) + rbk_o + n_o + 1}{(x - 1)rbn_o}. \tag{11.6.2}$$

The ratio of (11.6.2) is smaller than one if and only if

$$\frac{k_o}{n_o} = \frac{x - 1}{x} < \frac{bx - b - 1/r}{bx + b - 1} - \frac{bx + 1 - 1/r}{(bx + b - 1)n_o}. \tag{11.6.3}$$

The inequality (11.6.3) is most easily satisfied when the right hand side is a maximum; i.e., when $n_o \longrightarrow \infty$. First, if $r = 1$, then the condition becomes

$$\frac{x - 1}{x} < \frac{bx - b - 1}{bx + b - 1}, \tag{11.6.4}$$

which reduces to

$$b(x - 1) < -1. \tag{11.6.5}$$

Clearly, (11.6.5) can never be satisfied for positive values of b and x. Alternatively, as $r \longrightarrow \infty$, the inequality (11.6.3) becomes

$$\frac{x - 1}{x} < \frac{bx - b}{bx + b - 1}, \tag{11.6.6}$$

which reduces to

$$x - 1 > b(x - 1). \tag{11.6.7}$$

(11.6.7) can never be satisfied for integer values of b. Thus, because the condition (11.6.3) cannot be satisfied, the storage requirement of the outer decoder can never be smaller than its guard space requirement.

From Fig. 11.5.1, three of the decoder shift registers have tapped stages. The sub-block denoted $\underline{Q}^j$ is tapped in all of its $n_o - k_o$ stages. The entire computation register is tapped for a total of rbk_o stages. The storage register is tapped in $n_T(n_o - k_o)$ stages, where n_T is the number of tapped sub-block locations. Examination of the storage registers of a number of GSA codes shows that n_T is given by

$$n_T = l_E - b + \sum_{j=1}^{b} (x - 1)rj. \tag{11.6.8}$$

Thus, the total number of tapped shift register stages in the outer decoder is

$$N_T = [l_E - b + 1 + \sum_{j=1}^{b} (x - 1)rj](n_o - k_o) + rbk_o. \tag{11.6.9}$$

N_T is minimized by choosing a GSA code for which l_E is a minimum; i.e., an optimal code.

From (11.2.7), the burst correcting capability of the outer code is imparted by the sub-block

$$\underline{Q}^j = f(\underline{I}), \tag{11.6.10}$$

where $f(\underline{I})$ contains l_E terms. In forming the set of linearly independent simultaneous equations, each of the mod-2 sums $\underline{A}$, $\underline{B}$, $\underline{C}$, ... is given in general by

$$\underline{A} = \underline{Q}^j + \text{at most } (l_E - 1) \text{ terms from } f(\underline{I}). \tag{11.6.11}$$

If we consider the summations to be performed by an array of $n_o - k_o$ mod-2 adders with at most l_E inputs, then $(l_E - 1)(n_o - k_o)$ mod-2 adders with two inputs are required. However, at different times in the decoding process, the number of terms taken from $f(\underline{I})$, and the storage register sub-block locations tapped to supply these terms, may differ. Thus, the $(l_E - 1)(n_o - k_o)$ mod-2 adders may be given variable input leads, controlled by the state of the error-check register, or the decoder may be given additional mod-2 adders, whichever alternative appears more attractive in any particular application. In addition, from Figs. 11.5.6 and 11.5.8, a number of mod-2 adders may be required at the computation register in order to solve the set of equations for the unknown terms. The number, however, depends upon the specific GSA code and cannot be predicted in general. We therefore say that the number of mod-2 adders in the outer decoder is lower-bounded by

$$N_A \geq (l_E - 1)(n_o - k_o). \qquad (11.6.12)$$

In addition to N, N_T, and N_A, the total complexity of the GSA decoder includes the unspecified complexity of the inner minimum distance decoder.

11.7. Performance

To determine the probability of a decoding error given that a channel burst has occurred, $P(E \mid \text{burst})$, we know from (8.5.4) that for an adaptive code

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)[1 - P(F)] + P(F). \qquad (11.7.1)$$

The probability of failure P(F) is defined by the parameters of the inner (n_i, k_i) parity-check code. From (3.3.10), (3.3.11), and (3.3.12), since channel errors occur in the burst mode with probability q_o, the probability P_f that the inner code fails either to detect or to correct a burst in one block of length n_i is

$$P_f = P_d \, P_N,$$

$$P_N = 2^{k_i - n_i} \sum_{j=0}^{t} \binom{n_i}{j},$$

$$P_d = 1 - \sum_{j=0}^{d-t-1} \binom{n_i}{j} q_o^{\,j} (1 - q_o)^{n_i - j}. \qquad (11.7.2)$$

From (11.4.1), an integer number f/rb of sequences from the inner decoder is is required to form a single received block at the outer decoder. Thus, a failure occurs at the outer decoder only if there is a failure in all f/rb blocks at the inner decoder. Since a channel burst does not necessarily span all f/rb blocks, we can say only approximately that

$$P(F) \approx (P_f)^{f/rb}. \tag{11.7.3}$$

The probability of a decoding error given that the burst is detected, $P(E \mid \text{no } F)$, is primarily the probability of an error in the guard space G_{oy} of the outer decoder. This is equivalent to the probability of a decoding error in the guard space G_{iy} of the inner decoder. From (11.2.13) and (11.4.2), G_{iy} spans $(x - 1)yf/b$ blocks if the burst at the outer decoder is of length $r(y - 1)n_o + n_o, \ldots, ryn_o$, $y = 1, 2, \ldots, b$. The probability P_b of a decoding error in any one of these $(x - 1)yf/b$ channel blocks is, from (3.3.8),

$$P_b = 1 - \sum_{j=0}^{t} \binom{n_i}{j} p_o^{\,j} (1 - p_o)^{n_i - j}. \tag{11.7.4}$$

Since the decoding of blocks is independent, it follows that

$$P(E \mid \text{no } F) \approx 1 - (1 - P_b)^{(x-1)yf/b}. \tag{11.7.5}$$

From (9.5.4), the probability of a decoding error in the channel random mode is

$$P(E \mid \text{random}) = P(E \mid \text{no } A)[1 - P(A)] + P(E \mid A)\, P(A). \tag{11.7.6}$$

In order that a false alarm occur, at least one of f/rb sequences at the inner decoder must be distance between t + 1 and d - t - 1 from the corresponding transmitted codeword. Thus, the probability P(A) of a false alarm is approximately given by

$$P(A) \approx 1 - \Big[1 - \sum_{j=t+1}^{d-t-1} \binom{n_i}{j} p_o^{\,j} (1 - p_o)^{n_i - j}\Big]^{f/rb}. \tag{11.7.7}$$

Analogous to (8.5.11) and (11.7.5),

$$P(E \mid A) \approx P(E \mid \text{no } F) \approx 1 - (1 - P_b)^{(x-1)yf/b}. \tag{11.7.8}$$

Given no false alarm, a decoding error can occur only if at least one of f/rb sequences at the inner decoder contains a noise sequence with weight at least $d - t$. Thus,

$$P(E \mid \text{no } A) = 1 - \left[\sum_{j=0}^{d-t-1} \binom{n_i}{j} p_o^{\,j} (1 - p_o)^{n_i - j} \right]^{f/rb}. \qquad (11.7.9)$$

12. SUMMARY AND NUMERICAL COMPARISON OF BURST CORRECTING CODES

12.1. Introduction

In Chapters 6 to 11, a large number of error control techniques for the compound channel were described. Expressions were developed for the burst correcting capability, the guard space requirement, the complexity, and the performance of these codes. In this chapter, we shall summarize these results and evaluate the expressions for particular codes on specific channels. It will be assumed throughout this chapter that channel bursts very rarely exceed length $B_c \approx 1000$. In the channel random mode, errors will occur with probability p_o ranging from 10^{-8} to 10^{-2}, while in the channel burst mode, errors will occur with probability q_o ranging from 0.01 to 0.50.

Interleaved block codes, diffuse codes, Gallager codes, burst-trapping codes, and GSA codes are treated respectively in Sections 12.2., 12.3., 12.4., 12.5., and 12.6..

12.2. Interleaved Block Codes

Interleaved block codes, originally described in Chapter 6, are obtained by interleaving an (n,k) block code with burst correcting capability b. These codes have minimum guard space requirement, and thereby optimum performance, if b meets the Reiger bound (6.2.2) with equality,

$$b \leq \tfrac{1}{2}(n - k). \qquad (12.2.1)$$

From (6.2.1), interleaved block codes have burst correcting capability B_m such that

$$B_m = B_c = rb, \qquad (12.2.2)$$

where r is known as the interleaving degree. Their guard space requirement G_m is given by (6.2.4) and (6.2.5),

$$G_m = G_c = r(n - b) = rn - B_m. \qquad (12.2.3)$$

The complexity of the decoder is defined by the storage requirement N in (6.3.1),

$$N = rn = B_m + G_m. \qquad (12.2.4)$$

The performance of these codes in correcting bursts is bounded by (6.4.4),

$$P(E \mid \text{burst}) \leq 1 - (1 - p_o)^{G_m}. \qquad (12.2.5)$$

12.2.a. Numerical Example

Consider a (15,9) parity-check code, originally used by Weldon [47], which has optimal burst correcting capability b = 3. Interleaving this code to degree r = 334, we obtain

$$B_m = B_c = (334)(3) = 1002,$$

$$G_m = G_c = (334)(15 - 3) = 4008,$$

$$N = 1002 + 4008 = 5010,$$

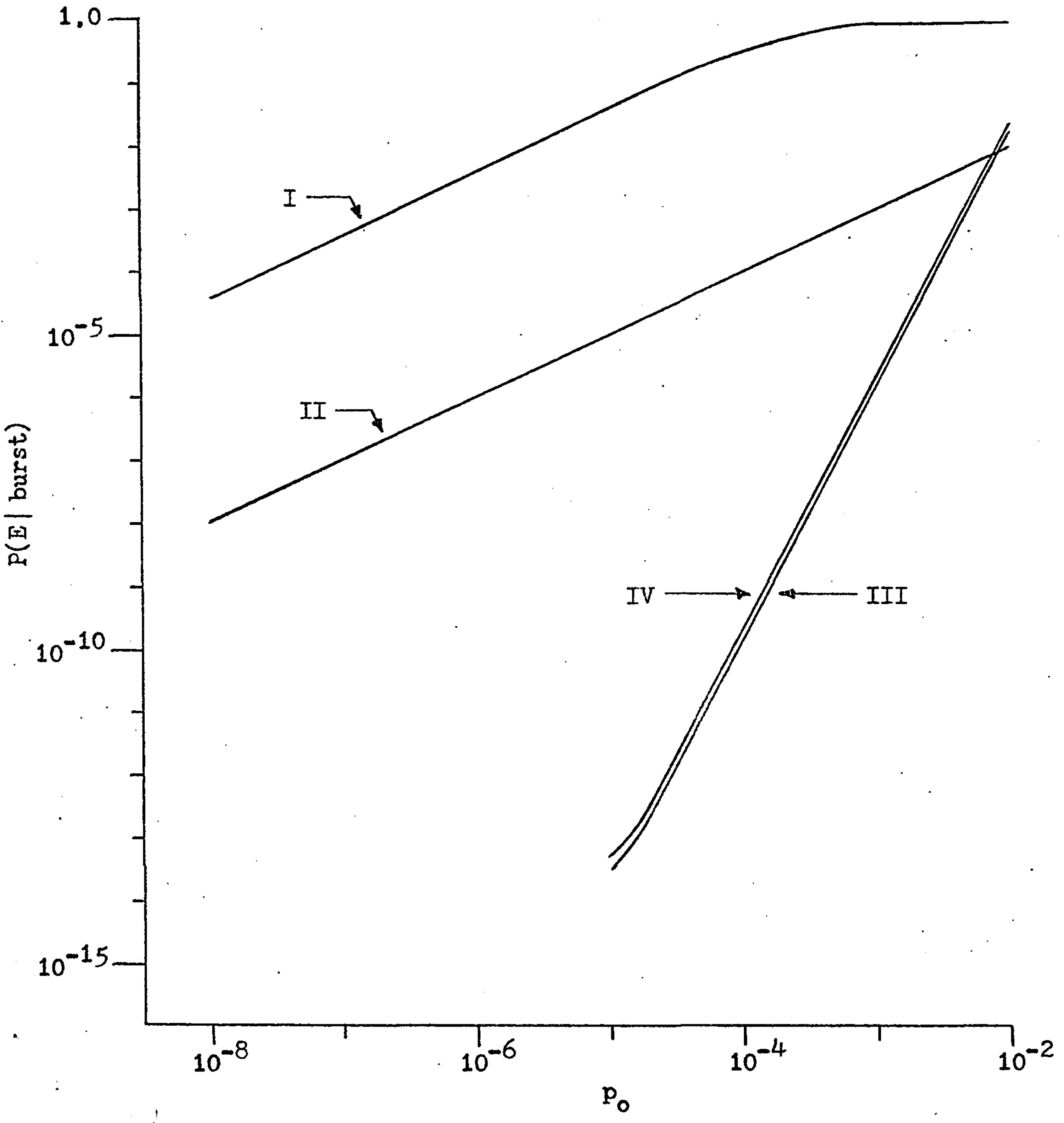
$$N/G_m = 5010/4008 = 1.25000. \qquad (12.2.6)$$

For the range of channel random error rate p_o between 10^{-8} and 10^{-2}, we obtain an upper bound on code performance P(E | burst) by substituting $G_m = 4008$ into (12.2.5). The result is shown in curve I of Fig. 12.2.1.

It follows from (12.2.2) and (12.2.3) that

$$G_m = \frac{n - b}{b} B_m = 4B_m. \qquad (12.2.7)$$

Thus, P(E | burst) is an increasing function of burst correcting capability B_m. P(E | burst) approaches 1 with increasing B_m and approaches p_o with decreasing B_m. The relationship P(E | burst) = p_o therefore is a lower bound on code performance when $B_m \longrightarrow 0$, shown in curve II of Fig. 12.2.1.



I: Interleaved block code alone, $B_m = 1002$.

II: Lower limit of performance for interleaved block code, $B_m \rightarrow 0$.

III: Inner block code and outer interleaved block code, $\lceil hf \rceil = 168$.

IV: Inner convolutional code and outer interleaved block code, $g = 328$.

Figure 12.2.1 : Performance of Interleaved Block Code.

12.2.b. Compound-Concatenated System With Inner Block Code

In Chapter 10 we saw that the performance of a burst correcting code could be improved by concatenating it with an inner random error correcting code; i.e., by forming a compound-concatenated system. Consider an inner (n,k) block code with minimum distance d and error correcting capability t given by (10.3.1),

$$t = \left\lfloor \frac{d-1}{2} \right\rfloor . \tag{12.2.8}$$

The maximum length B_c of channel bursts is expressed as in (10.3.2),

$$B_c = fn. \tag{12.2.9}$$

The outer burst correcting code has burst correcting capability B_m and guard space requirement G_m given by (10.3.3) and (10.3.5),

$$B_m = fk = \frac{k}{n} B_c, \tag{12.2.10}$$

$$G_m = h\, B_m = hfk. \tag{12.2.11}$$

From (10.3.6) and (10.3.7), the channel guard space G_c is

$$G_c = \lceil hf \rceil n \approx h\, B_c \approx \frac{n}{k} G_m. \tag{12.2.12}$$

At the outer decoder, errors in the random mode occur approximately with probability p_1 and errors in the burst mode occur approximately with probability q_1, given by (10.6.2) and (10.6.3) respectively,

$$p_1 \approx 1 - \sum_{j=0}^{t} \binom{n}{j} p_o^{\,j} (1-p_o)^{n-j}, \tag{12.2.13}$$

$$q_1 \approx 1 - \sum_{j=0}^{t} \binom{n}{j} q_o^{\,j} (1-q_o)^{n-j}. \tag{12.2.14}$$

Throughout this chapter, we assume that the inner block code is the (24,12) extended Golay code [1] with error correcting capability $t = 3$. With

$f = 42$, we obtain from (12.2.9) and (12.2.10),

$$B_c = (42)(24) = 1008,$$

$$B_m = (42)(12) = 504 = \tfrac{1}{2}B_c. \qquad (12.2.15)$$

Consider a compound-concatenated system whose outer code belongs to the same class as the interleaved block code of Section 12.2.a.. With interleaving degree $r = 168$, we obtain

$$B_m = (168)(3) = 504,$$

$$G_m = (168)(15 - 3) = 2016,$$

$$h = 2016/504 = 4,$$

$$\lceil hf \rceil = \lceil (4)(42) \rceil = 168,$$

$$G_c = (168)(24) = 4032,$$

$$N = 504 + 2016 \doteq 2520,$$

$$N/G_m = 2520/2016 = 1.25000. \qquad (12.2.16)$$

From (10.6.1) and (10.6.5), the performance of the compound-concatenated system is bounded by

$$P(E \mid \text{burst}) \leq 1 - (1 - p_1)^{\lceil hf \rceil}. \qquad (12.2.17)$$

For every p_o, p_1 is found by substituting the parameters $n = 24$ and $t = 3$ of the inner block code into (12.2.13). Then, with $\lceil hf \rceil = 168$, we obtain the upper bound on system performance $P(E \mid \text{burst})$ shown in curve III of Fig. 12.2.1. Because B_m is linearly related to f by (12.2.10), $P(E \mid \text{burst})$ approaches 1 with increasing B_m and approaches p_1 with decreasing B_m.

12.2.c. Compound-Concatenated System With Inner Convolutional Code

Another compound-concatenated system may be formed by using as the inner code a rate $\frac{1}{2}$ self-orthogonal convolutional code with constraint length n_A, effective length n_E, and error correcting capability t from (10.4.1),

$$t = \left\lfloor {}^{J}/_{2} \right\rfloor . \qquad (12.2.18)$$

J is the number of nonzero coefficients in the code-generating polynomial $G_2(D)$, which has maximum degree u. The maximum length B_c of channel bursts is expressed as in (10.4.2),

$$B_c = sn_A. \qquad (12.2.19)$$

Channel bursts may be propagated at the inner decoder for some length $W \geq wn_A$, where W is given by (10.4.8) and wn_A is given by (10.4.3). The outer burst correcting code then has burst correcting capability B_m and guard space requirement G_m given by (10.4.9) and (10.4.10),

$$B_m = \tfrac{1}{2}(sn_A + W) = \tfrac{1}{2}(B_c + W), \qquad (12.2.20)$$

$$G_m = h\ B_m = \frac{h}{2}(sn_A + W). \qquad (12.2.21)$$

From (10.4.11), the channel guard space G_c is

$$G_c = W + h(B_c + W). \qquad (12.2.22)$$

At the outer decoder, errors in the random mode occur approximately with probability p_2 and errors in the burst mode occur approximately with probability q_2, given by (10.7.1) and (10.7.2) respectively,

$$p_2 \approx 1 - \sum_{j=0}^{t} \binom{n_E}{j} p_o^{\,j} (1 - p_o)^{n_E - j}, \qquad (12.2.23)$$

$$q_2 \approx 1 - \sum_{j=0}^{t} \binom{n_E}{j} q_o^{\,j} (1 - q_o)^{n_E - j}. \qquad (12.2.24)$$

Throughout this chapter, we assume that the inner convolutional code is the rate ½ code taken from Robinson and Bernstein [38] with the code-generating polynomial

$$G_2(D) = 1 + D^2 + D^7 + D^{13} + D^{16} + D^{17}. \qquad (12.2.25)$$

This code has the parameters $n_A = 36$, $n_E = 22$, $J = 6$, $t = 3$, $u = 17$, and $wn_A = 106$. With $s = 28$, we obtain from (12.2.19),

$$B_c = (28)(36) = 1008. \qquad (12.2.26)$$

The parameter W must be chosen so that the probability that a channel burst is propagated beyond the burst correcting capability of the outer code is small compared to p_2, the probability of a decoding error in the channel guard space. Consider $W = 792$. At least eight random errors are necessary to result in no error-free run of length 106 in a sequence of 792 bits, whereas at least four random errors in the effective length $n_E = 22$ are needed to cause a decoding error in the channel guard space. When p_o is small, the probability of multiple random errors is relatively insensitive to the length of the noise sequence, so the probability of eight errors in 792 digits is smaller than the probability of four errors in 22 digits. A choice of $W = 792$ is thus sufficient to make error propagation a negligible factor in the compound-concatenated system. Therefore, from (12.2.20),

$$B_m = \tfrac{1}{2}(1008 + 792) = 900. \qquad (12.2.27)$$

Consider a compound-concatenated system whose outer code belongs to the same class as the interleaved block code of Section 12.2.a.. With interleaving degree $r = 300$, we obtain

$$B_m = (300)(3) = 900,$$

$$G_m = (300)(15 - 3) = 3600,$$

$$h = 3600/900 = 4,$$

$$G_c = 792 + (4)(1008 + 792) = 7992,$$

$$N = 900 + 3600 = 4500,$$

$$N/G_m = 4500/3600 = 1.25000. \qquad (12.2.28)$$

Because error propagation is a negligible factor, we have from (10.6.1) and (10.7.9) that the performance of the compound-concatenated system is approximately

$$P(E \mid \text{burst}) \approx 1 - (1 - p_2)^g,$$

$$g = \left\lceil \frac{h(B_c + W)}{n_E} \right\rceil = \left\lceil \frac{(4)(1008 + 792)}{22} \right\rceil = 328. \qquad (12.2.29)$$

For every p_o, p_2 is found by substituting the parameters $n_E = 22$ and $t = 3$ of the inner convolutional code into (12.2.23). Then, with $g = 328$, we obtain the approximate system performance $P(E \mid \text{burst})$ shown in curve IV of Fig. 12.2.1. Since g is proportional to $B_c + W = 2B_m$, $P(E \mid \text{burst})$ approaches 1 with increasing B_m and approaches p_2 with decreasing B_m.

12.2.d. Comparison

We showed in the preceding sub-sections how code performance $P(E \mid \text{burst})$ is bounded by variations in the burst correcting capability B_m of the burst correcting code. In all cases, $P(E \mid \text{burst})$ approaches 1 as B_m approaches infinity, independent of the channel error rate p_o. This is a very reasonable conclusion and clearly holds for all burst correcting codes. In all cases also, $P(E \mid \text{burst})$ is of the general form

$$P(E \mid \text{burst}) \approx 1 - (1 - p)^x, \qquad (12.2.30)$$

where p is the random error rate in the decoder guard space of the burst correcting code and x is linearly related to B_m. Thus, as B_m approaches zero, x approaches zero, and $P(E \mid \text{burst})$ approaches p as a lower limit. This too is a very reasonable conclusion, since with $B_m = 0$ and B_m proportional to B_c, the channel does not produce bursts, there is no burst correcting code, and errors occur at the "natural" random error rate p. This relation holds for all codes which obey (12.2.30).

Fig. 12.2.1 shows clearly the potential improvement available by concatenating an interleaved block code with a random error correcting code. Since both inner codes have rate $\frac{1}{2}$, we are in effect halving the system rate in exchange for reduced storage requirement and an improvement in performance of roughly nine orders of magnitude at $p_o = 10^{-4}$.

12.3. Diffuse Codes

The diffuse codes originally described in Chapter 7 are rate $\frac{1}{2}$ feedback decodable convolutional codes which treat bursts of length B_m or less as if they contain no more than t errors among the n_E checked bits. From (7.2.1), (7.2.2), and (7.2.3),

$$B_m = B_c = 2B, \tag{12.3.1}$$

$$t = {}^J/_2, \tag{12.3.2}$$

$$n_E \geq 2t^2 + t + 1. \tag{12.3.3}$$

From (7.2.4), (7.2.6), and (7.2.10), the constraint length n_A, the storage requirement N, and the guard space requirement G_m are related by

$$n_A = N = 2N_s = G_m + 2 = G_c + 2, \tag{12.3.4}$$

where N_s is the shift register length in the decoder, given by (7.2.8),

$$N_s \geq 3B + 1. \tag{12.3.5}$$

Besides storage requirement, the decoder complexity is also measured by the number of tapped shift register stages, N_T, and the number of mod-2 adders with two inputs, N_A. From (7.4.3) and (7.4.4),

$$N_T \geq 4t + 1, \tag{12.3.6}$$

$$N_A \geq 2t + 2. \tag{12.3.7}$$

The performance of diffuse codes in correcting bursts is given by (7.6.2),

$$P(E \mid \text{burst}) \approx 1 - (1 - p_o)^{G_m}. \tag{12.3.8}$$

12.3.a. Numerical Example

Consider the diffuse code which was derived in Section 7.3.. This code has the parameters $J = 4$, $t = 2$, $n_E = 11$, and $N_s = 3B + 2$. The effective length is optimal and the shift register length is asymptotically optimal. With $B = 500$, we obtain

$$B_m = B_c = (2)(500) = 1000,$$

$$N = n_A = (6)(500) + 4 = 3004,$$

$$G_m = G_c = 3004 - 2 = 3002,$$

$$N/G_m = 3004/3002 = 1.00067,$$

$$N_T = (4)(2) + 1 = 9,$$

$$N_A = (2)(2) + 2 = 6. \tag{12.3.9}$$

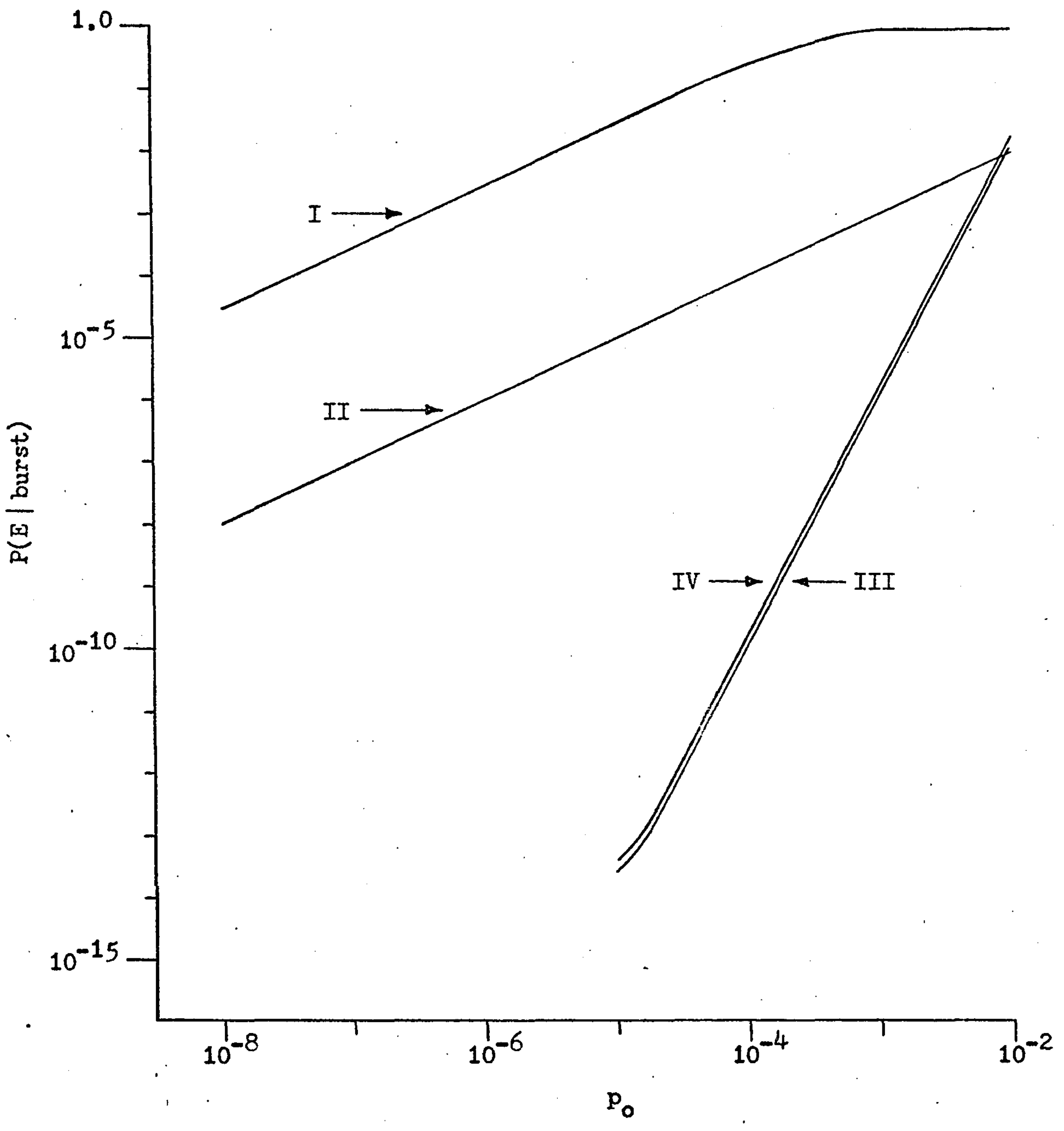
For p_o between 10^{-8} and 10^{-2}, we obtain the approximate code performance $P(E \mid \text{burst})$ by substituting $G_m = 3002$ into (12.3.8). The result is shown in curve I of Fig. 12.3.1. The lower bound on code performance when $B_m \longrightarrow 0$ is shown in curve II.

12.3.b. Compound-Concatenated System With Inner Block Code

Consider a compound-concatenated system whose outer code belongs to the same class as the diffuse code of Section 12.3.a. and whose inner code is the extended Golay code of Section 12.2.b.. With $B = 252$, we obtain

$$B_m = (2)(252) = (42)(12) = 504,$$

$$N = (6)(252) + 4 = 1516,$$



I: Diffuse code alone, $B_m = 1000$.

II: Lower limit of performance for diffuse code, $B_m \to 0$.

III: Inner block code and outer diffuse code, $\lceil hf \rceil = 127$.

IV: Inner convolutional code and outer diffuse code, $g = 246$.

Figure 12.3.1 : Performance of Diffuse Code.

$$G_m = 1516 - 2 = 1514,$$

$$h = 1514/504 = 3.00397,$$

$$\lceil hf \rceil = \lceil (3.00397)(42) \rceil = 127,$$

$$G_c = (127)(24) = 3048,$$

$$N/G_m = 1516/1514 = 1.00132. \qquad (12.3.10)$$

Analogous to (12.2.17), if robustness and error propagation are approximately equal and opposite effects in the feedback decoder, system performance is given by

$$P(E \mid \text{burst}) \approx 1 - (1 - p_1)^{\lceil hf \rceil}. \qquad (12.3.11)$$

p_1 is again found from (12.2.13), and with $\lceil hf \rceil = 127$, $P(E \mid \text{burst})$ is shown in curve III of Fig. 12.3.1.

12.3.c. Compound-Concatenated System With Inner Convolutional Code

Consider a compound-concatenated system whose outer code belongs to the same class as the diffuse code of Section 12.3.a. and whose inner code is the convolutional code of Section 12.2.c.. With $B = 450$, we obtain

$$B_m = (2)(450) = \tfrac{1}{2}(1008 + 792) = 900,$$

$$N = (6)(450) + 4 = 2704,$$

$$G_m = 2704 - 2 = 2702,$$

$$h = 2702/900 = 3.00222,$$

$$g = \left\lceil \frac{(3.00222)(1008 + 792)}{22} \right\rceil = 246,$$

$$G_c = 792 + (3.00222)(1008 + 792) = 6196,$$

$$N/G_m = 2704/2702 = 1.00074. \qquad (12.3.12)$$

Analogous to (12.2.29), the performance of the compound-concatenated system is approximately

$$P(E \mid \text{burst}) \approx 1 - (1 - p_2)^g. \qquad (12.3.13)$$

p_2 is found from (12.2.23), and with $g = 246$, $P(E \mid \text{burst})$ is shown in curve IV of Fig. 12.3.1.

12.3.d. Comparison

The improvement in performance available with a compound-concatenated system is evident from Fig. 12.3.1. Another obvious point is the great similarity between this figure and Fig. 12.2.1, the equivalent for an interleaved block code. This was to be expected since both codes are non-adaptive and meet the Reiger or Gallager bounds with (near) equality. It should be noted, however, that the interleaved block code has rate 0.6, while the diffuse code has rate 0.5. Thus, the interleaved block code achieves nearly the same performance with a 20 percent increase in efficiency, while in return, the diffuse code is much less expensive to implement.

12.4. Gallager Codes

Gallager codes, originally described in Chapter 8, are an adaptive burst correcting technique. Their random mode is obtained from a rate $\frac{1}{2}$ feedback decodable convolutional code whose code-generating polynomial $G_2(D)$ has maximum degree u with J nonzero coefficients. The convolutional code has constraint length n_A^*, effective length n_E, and error correcting capability t given by (8.2.2), (8.2.3), and (8.2.4),

$$n_A^* = 2(u + 1) = 2N_s^*, \qquad (12.4.1)$$

$$n_E \geq \tfrac{1}{2}J^2 + \tfrac{1}{2}J + 1, \qquad (12.4.2)$$

$$1 \leq t \leq \left\lfloor {}^{J}/_{2} \right\rfloor. \qquad (12.4.3)$$

From (8.2.1), the Gallager code has burst correcting capability B_m

such that

$$B_m = B_c = 2B. \tag{12.4.4}$$

The shift register length N_s is given by (8.2.5),

$$N_s = B + N_s^* = B + u + 1, \tag{12.4.5}$$

and the constraint length n_A, the storage requirement N, and the guard space requirement are related by (8.2.6), (8.3.7), (8.3.8), and (8.4.1),

$$N = n_A = B_m + 2(u + 1) = G_m - 2(y - 1), \tag{12.4.6}$$

$$G_m = G_c = B_m + 2(u + y) \ , \ B - y \leq b \leq B,$$

$$G_{db} = G_{cb} = 2(b + u + 2y) \ , \ b < B - y, \tag{12.4.7}$$

where y is the number of consecutive zero-valued syndrome bits required to initiate a transition from the decoder burst mode to the decoder random mode, and b is the number of information bits contained in the burst. The decoder complexity is also defined by (8.4.3) and (8.4.4),

$$N_T \geq 2J + 2, \tag{12.4.8}$$

$$N_A \geq J + 2. \tag{12.4.9}$$

In the channel random mode, code performance is given by (8.5.8), (8.5.9), (8.5.10), (8.5.11), and (8.5.12),

$$P(E \mid \text{random}) = P(E \mid \text{no } A)[1 - P(A)] + P(E \mid A)\, P(A), \tag{12.4.10}$$

$$P(A) \leq \sum_{j=t}^{J-t} \binom{n_E}{j} p_o^{\,j} (1 - p_o)^{n_E - j}, \tag{12.4.11}$$

$$P(E \mid A) \approx 1 - (1 - p_o)^{G_{db}}, \tag{12.4.12}$$

$$P(E \mid \text{no } A) \leq 1 - \sum_{j=0}^{J-t} \binom{n_E}{j} p_o^{\,j} (1 - p_o)^{n_E - j}. \tag{12.4.13}$$

In the channel burst mode, performance is given by (8.5.4), (8.5.5), and (8.5.7),

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)[1 - P(F)] + P(F), \qquad (12.4.14)$$

$$P(F) \approx 1 - \sum_{j=0}^{J-t} \binom{n_E}{j} q_o^{\,j} (1 - q_o)^{n_E - j}, \qquad (12.4.15)$$

$$P(E \mid \text{no } F) \approx 1 - (1 - p_o)^{G_{db}}. \qquad (12.4.16)$$

12.4.a. Numerical Example

Consider the Gallager code whose random mode is obtained from the self-orthogonal convolutional code in Robinson and Bernstein [38] with the code-generating polynomial

$$G_2(D) = 1 + D^2 + D^7 + D^{15} + D^{21} + D^{24} + D^{25}, \qquad (12.4.17)$$

and with the parameters $n_A^* = 52$, $n_E = 29$, $J = 7$, and $u = 25$. With $B = 500$ and $y = 20$, we obtain

$$B_m = B_c = (2)(500) = 1000,$$

$$G_m = G_c = 1000 + (2)(25 + 20) = 1090 \;, \; 480 \leq b \leq 500,$$

$$G_{db} = G_{cb} = 2b + (2)(25 + 40) = 2b + 130 \;, \; b < 480,$$

$$N = n_A = 1000 + (2)(25 + 1) = 1052,$$

$$N/G_m = 1052/1090 = 0.96514,$$

$$N_T = (2)(7) + 2 = 16,$$

$$N_A = 7 + 2 = 9. \qquad (12.4.18)$$

For this code, $J = 7$ and $n_E = 29$ are constant parameters, while random error correcting capability may span the range $t = 1, 2, 3$ and decoder guard space requirement is bounded by $G_{db} = 130, 1090$. For the range of channel random error rate p_o between 10^{-8} and 10^{-2}, we obtain the random mode performance curves $P(E \mid \text{random})$ for this code, Fig. 12.4.1, by substituting the various parameters into (12.4.10), (12.4.11), (12.4.12), and (12.4.13). These curves

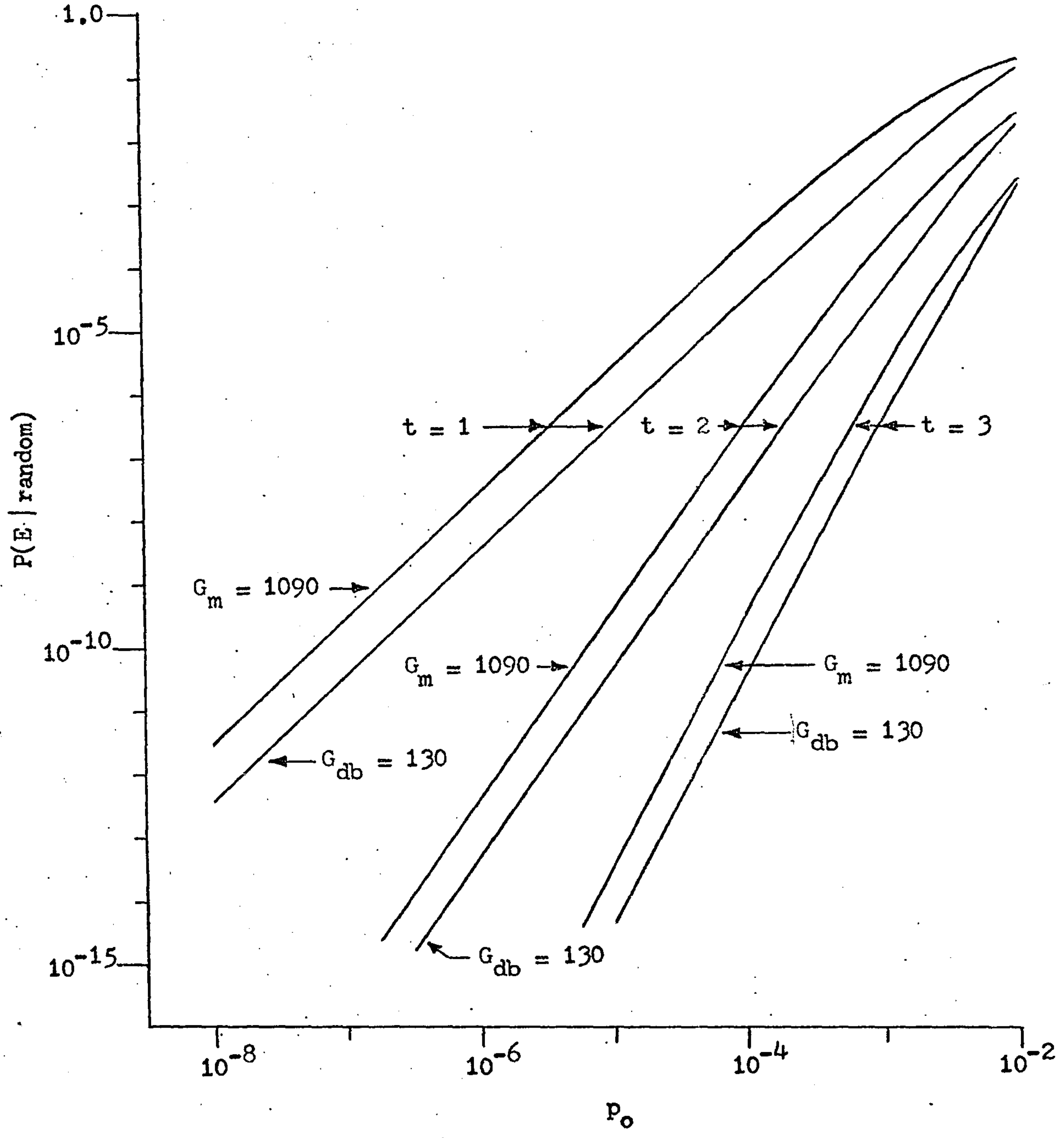


Figure 12.4.1 : Performance of Gallager Code in Channel Random Mode.

are actually pairs of bounds for different values of t. The bounds are given because any particular false alarm can require any guard space between 130 and 1090.

For the range of channel burst error rate q_o between 0.01 and 0.50, we obtain the probability of failure P(F) of this code, Fig. 12.4.2, by substituting the code parameters into (12.4.15). This figure shows clearly that by decreasing the error correcting capability t, we increase the detecting capability of the code, thereby improving P(F). Note that for this particular code, P(F), which lower-bounds $P(E \mid \text{burst})$, becomes prohibitively large for bursts with error densities in excess of 5 percent. This situation can be improved by choosing a convolutional code for which J is greater. However, increasing J leads to increasing complexity N, N_T, and N_A, as well as increasing maximum guard space G_m and increasing effective length n_E. For large values of q_o, the effect of the factor $(1 - q_o)^{n_E - j}$ in (12.4.15) may be such that increasing n_E by increasing J will result in a deterioration of performance. Careful consideration must be given to all these trade-offs when selecting a code for a particular channel.

By substituting the code parameters into (12.4.16), we obtain the limits on $P(E \mid \text{no } F)$, the probability of a decoding error given that the burst is detected, shown in curves I and II of Fig. 12.4.3. Combining the results of Figs. 12.4.2 and 12.4.3 as in (12.4.14), we obtain the performance $P(E \mid \text{burst})$ of the Gallager code as shown in curves I and II of Figs. 12.4.4, 12.4.5, and 12.4.6. The fact that $P(E \mid \text{burst})$ is lower-bounded by P(F) is evident in these curves, and becomes increasingly evident for larger values of q_o, when P(F) itself becomes large.

Two particularly interesting points are brought out by Figs. 12.4.4, 12.4.5, and 12.4.6. First, the performance of the adaptive Gallager code is

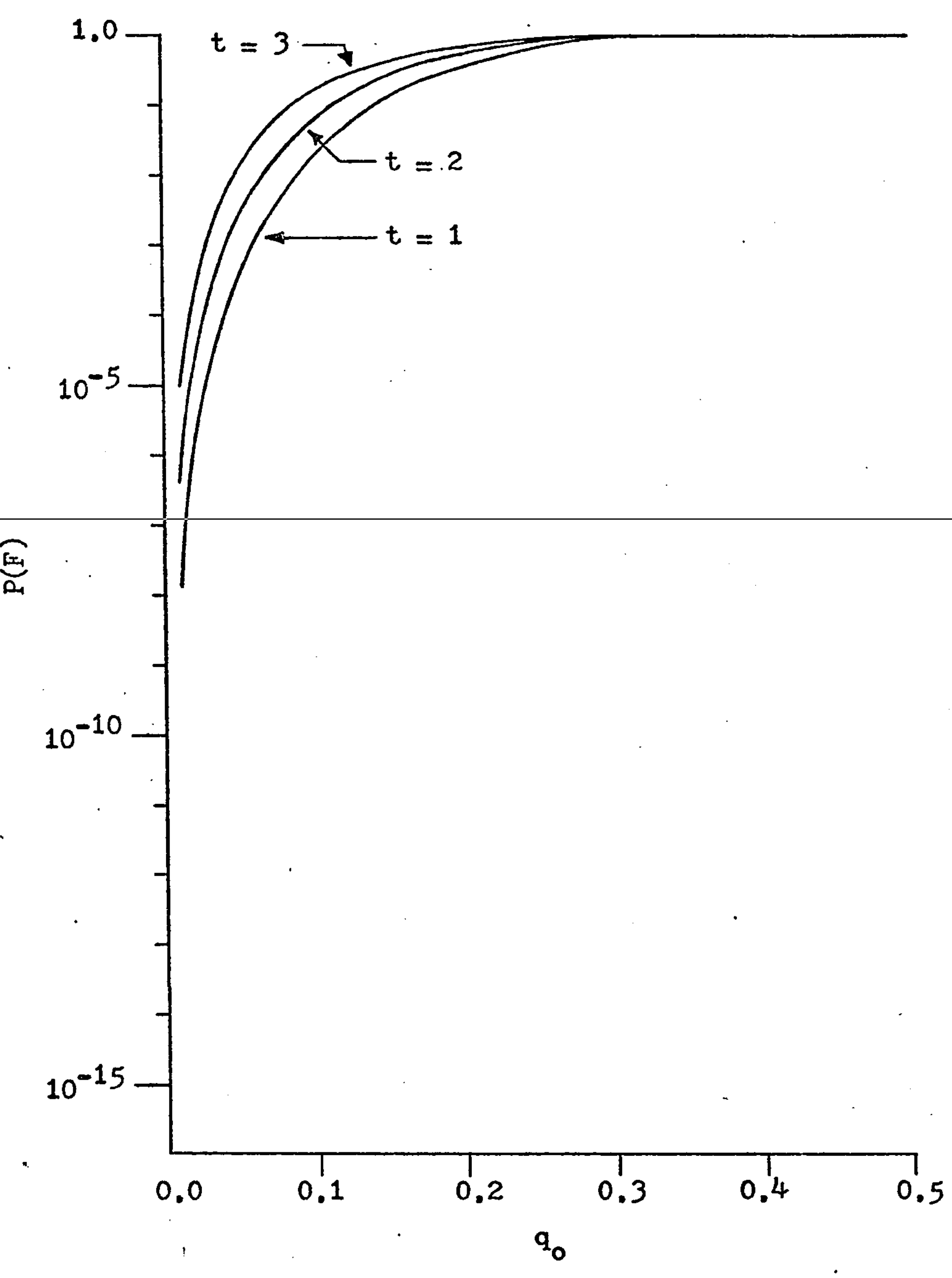
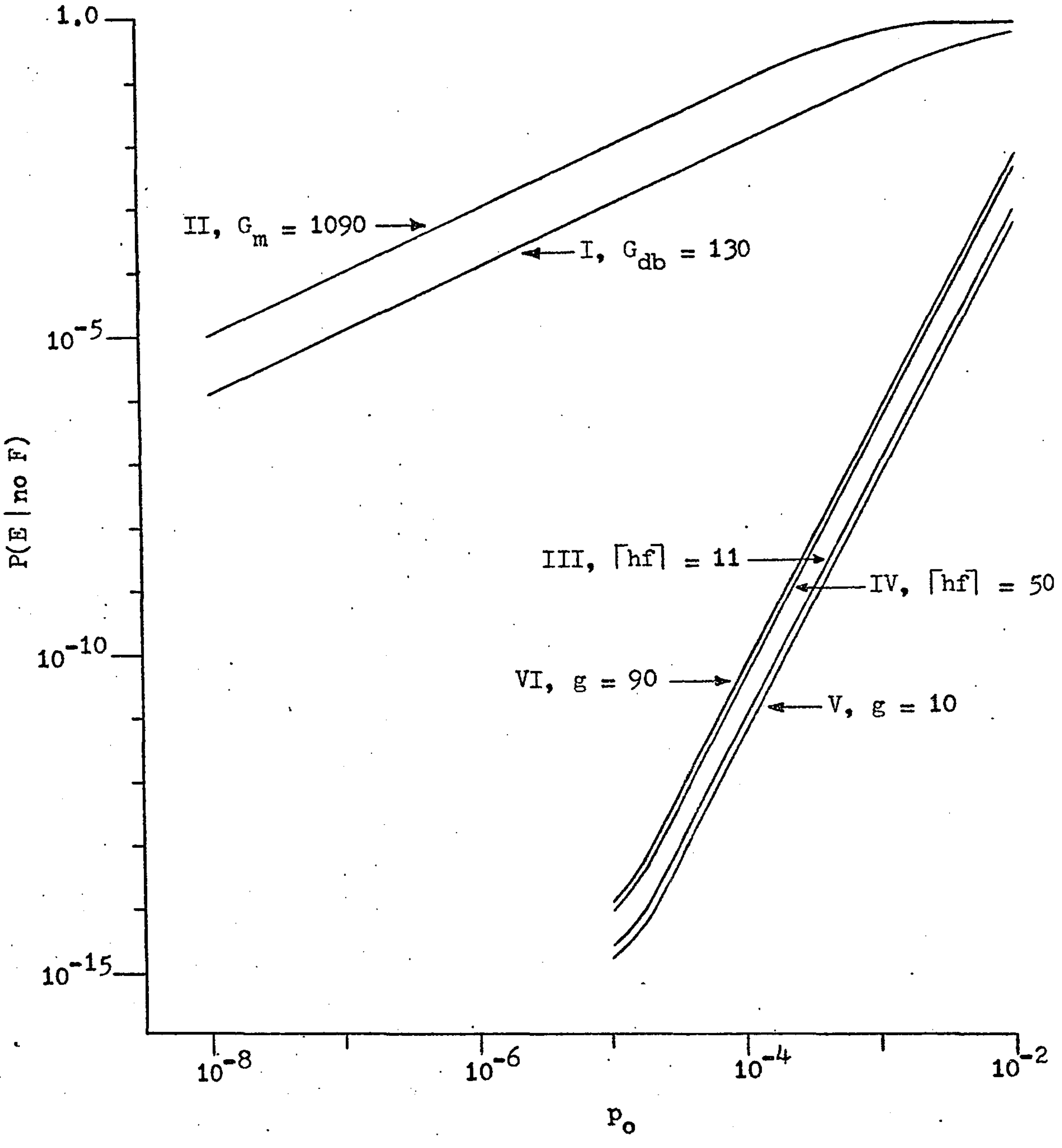
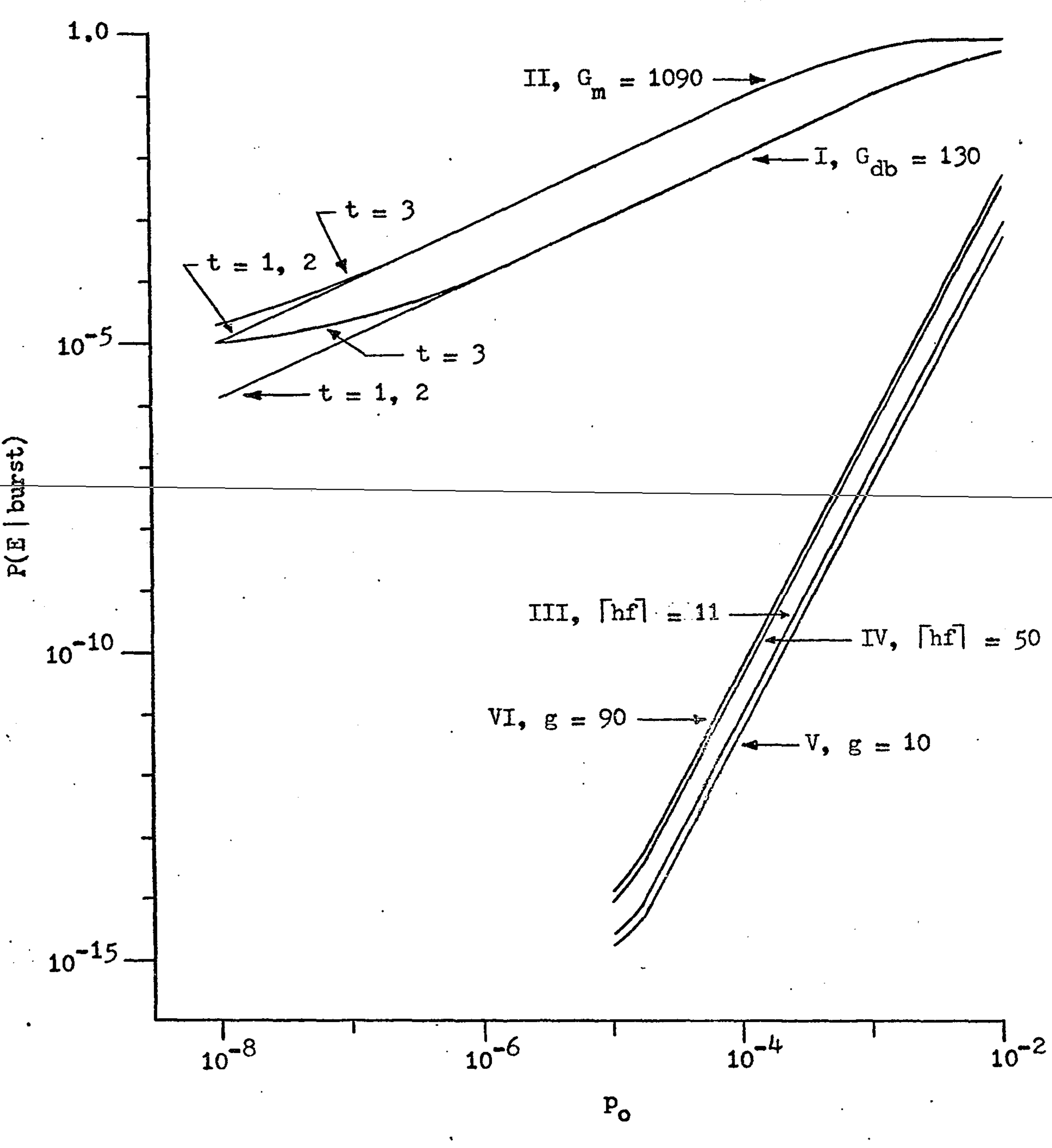


Figure 12.4.2 : Probability of Failure of Gallager Code.



I, II:	Limits for Gallager code alone.
III, IV:	Limits for inner block code and outer Gallager code.
V, VI:	Limits for inner convolutional code and outer Gallager code.

Figure 12.4.3 : Performance of Gallager Code Given a Detected Burst.

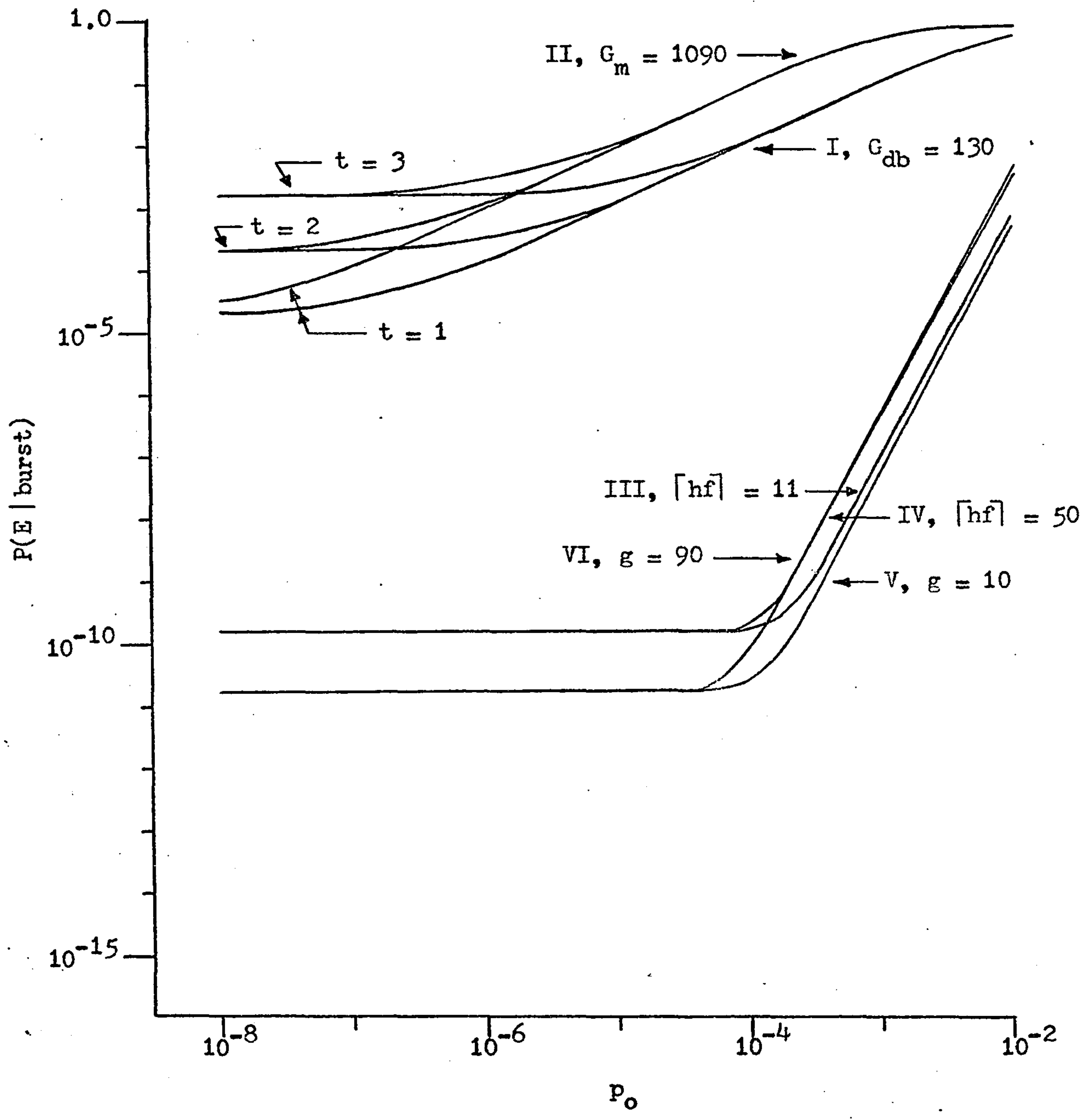


I, II: Limits for Gallager code alone.

III, IV: Limits for inner block code and outer Gallager code.

V, VI: Limits for inner convolutional code and outer Gallager code.

Figure 12.4.4 : Performance of Gallager Code Given That $q_o = 0.010$.

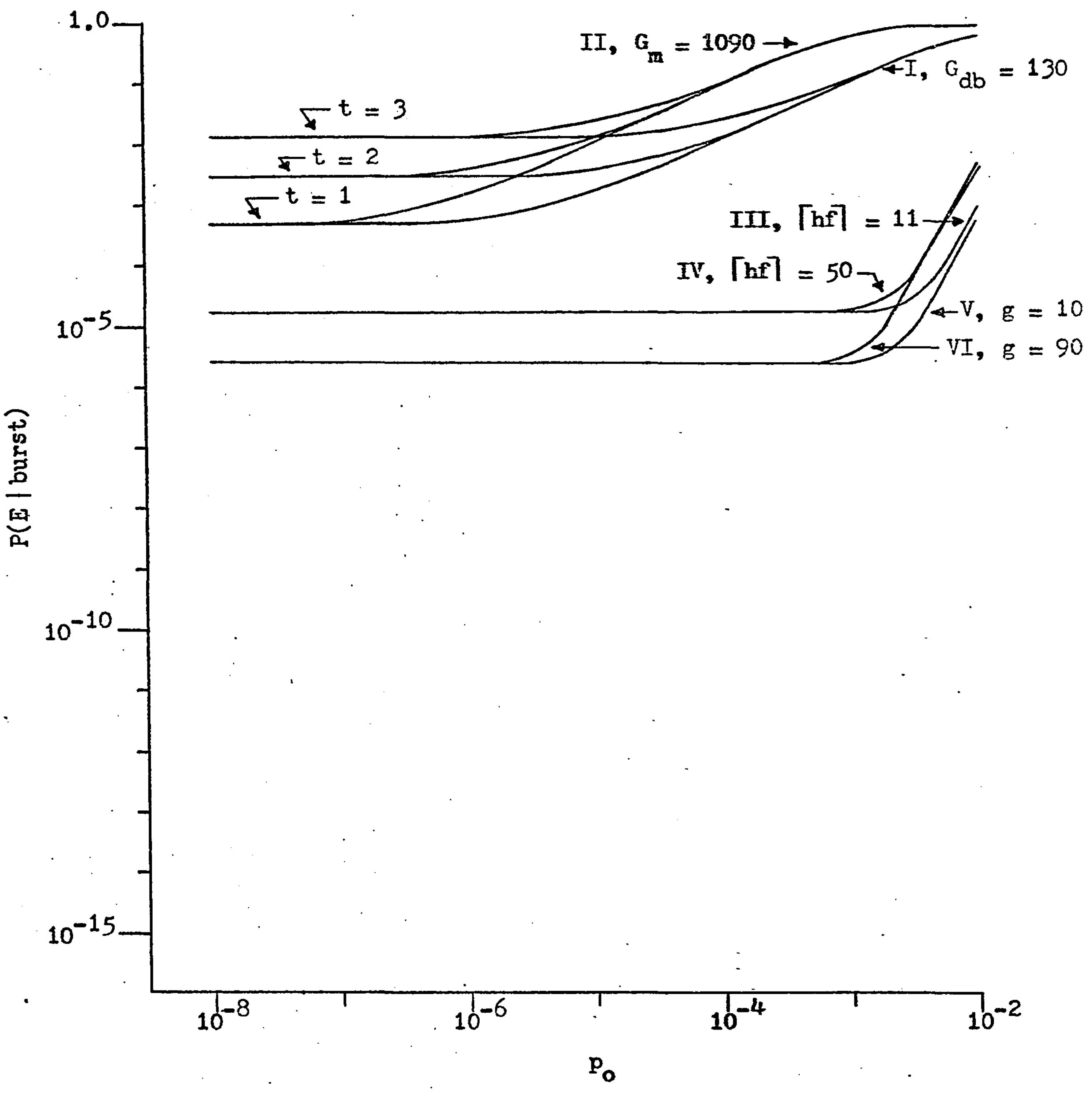


I, II: Limits for Gallager code alone.

III, IV: Limits for inner block code and outer Gallager code.

V, VI: Limits for inner convolutional code and outer Gallager code.

Figure 12.4.5 : Performance of Gallager Code Given That $q_o = 0.030$.



I, II: Limits for Gallager code alone.

III, IV: Limits for inner block code and outer Gallager code.

V, VI: Limits for inner convolutional code and outer Gallager code.

Figure 12.4.6 : Performance of Gallager Code Given That $q_o = 0.050$.

extremely sensitive to the error rate q_o in the channel burst mode. This particular code is essentially useless for $q_o > 0.05$. Second, for mid-range values of p_o, around 10^{-4}, before the effect of P(F) becomes important, the performance is nearly independent of the choice of error correcting capability t of the convolutional code. Thus, by choosing t large, we can realize an improvement of six or more orders of magnitude in $P(E \mid \text{random})$, Fig. 12.4.1, without suffering any important penalty in $P(E \mid \text{burst})$.

12.4.b. Compound-Concatenated System With Inner Block Code

Consider a compound-concatenated system whose outer code belongs to the same class as the Gallager code of Section 12.4.a. and whose inner code is the extended Golay code of Section 12.2.b.. With B = 252 and y = 20, we obtain

$$\begin{aligned}
B_m &= (2)(252) = (42)(12) = 504,\\
G_m &= 504 + (2)(25 + 20) = 594\ ,\ 232 \le b \le 252,\\
G_{db} &= 2b + (2)(25 + 20) = 2b + 130\ ,\ b < 232,\\
N &= 504 + (2)(25 + 1) = 556,\\
h &= 594/504 = 1.17857,\\
N/G_m &= 556/594 = 0.92923. \qquad (12.4.19)
\end{aligned}$$

From (12.2.12), we know that for any decoder guard space G_{db}, the corresponding channel guard space G_{cb} is

$$G_{cb} \approx \frac{n}{k} G_{db} = \frac{24}{12} G_{db} = 2G_{db}$$

such that $$G_{cb} = \lceil hf \rceil n = 24\lceil hf \rceil\ ,\ \lceil hf \rceil \text{ an integer.} \qquad (12.4.20)$$

For the range of G_{db} in (12.4.19), we have

$$G_{cb} = 264,\ 288,\ \ldots,\ 1200,$$

$$G_c = G_{cb}^{\max} = 1200,$$

$$\lceil hf \rceil = 11,\ 12,\ \ldots,\ 50,$$

$$\lceil hf \rceil^{\max} = \lceil (1.17857)(42) \rceil = 50. \qquad (12.4.21)$$

From (10.6.1), (12.4.15), and (10.6.5), the performance of the compound-concatenated system in correcting bursts is given by

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)[1 - P(F)] + P(F), \qquad (12.4.22)$$

$$P(F) \approx 1 - \sum_{j=0}^{J-t} \binom{n_E}{j} q_1^{\,j} (1 - q_1)^{n_E - j}, \qquad (12.4.23)$$

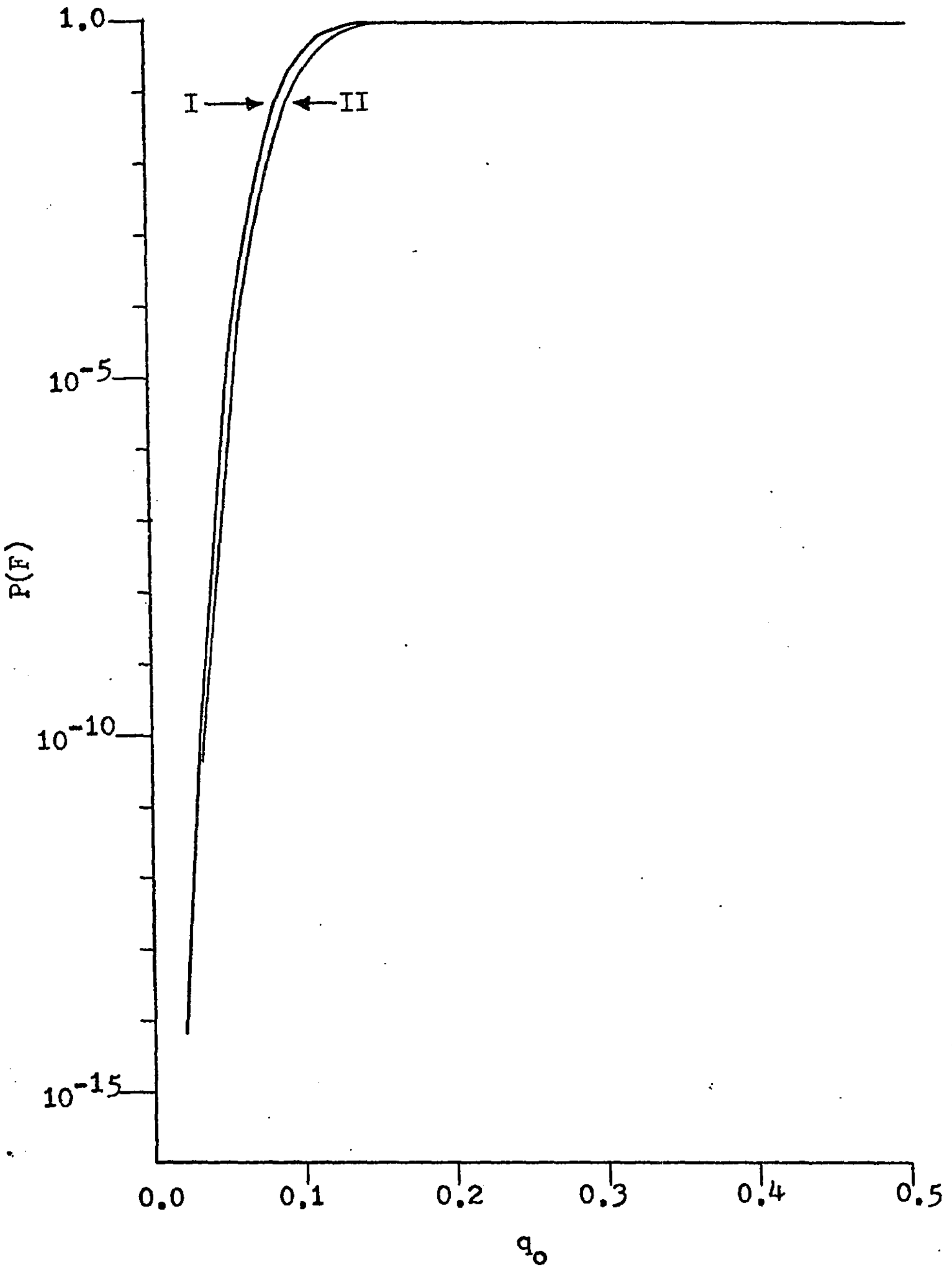
$$P(E \mid \text{no } F) \approx 1 - (1 - p_1)^{\lceil hf \rceil}. \qquad (12.4.24)$$

For every p_o and q_o, the corresponding values of p_1 and q_1 are found by substituting the parameters $n = 24$ and $t = 3$ of the inner block code into (12.2.13) and (12.2.14). We obtain the probability of failure P(F) of the compound-concatenated system by substituting the parameters of the Gallager code, $J = 7$, $n_E = 29$, and $t = 1$, into (12.4.23). The result is shown in curve I of Fig. 12.4.7. We choose $t = 1$ because it is known from Fig. 12.4.2 to yield the best curve for P(F).

By substituting the extreme values $\lceil hf \rceil = 11,\ 50$ into (12.4.24), we obtain the limits of $P(E \mid \text{no } F)$, the probability of a decoding error given a detected burst, shown in curves III and IV of Fig. 12.4.3. Combining P(F) and $P(E \mid \text{no } F)$ as in (12.4.22), we obtain the performance $P(E \mid \text{burst})$ of the compound-concatenated system as shown in curves III and IV of Figs. 12.4.4, 12.4.5, and 12.4.6.

12.4.c. Compound-Concatenated System With Inner Convolutional Code

Consider a compound-concatenated system whose outer code belongs to the same class as the Gallager code of Section 12.4.a. and whose inner code is the convolutional code of Section 12.2.c.. With $B = 450$, $y = 20$, and $W = 792$, we obtain



I: With inner block code.

II: With inner convolutional code.

Figure 12.4.7 : Probability of Failure of Compound-Concatenated System With Outer Gallager Code.

$$B_m = (2)(450) = \tfrac{1}{2}(1008 + 792) = 900,$$

$$G_m = 900 + (2)(25 + 20) = 990 \ , \ 430 \le b \le 450,$$

$$G_{db} = 2b + (2)(25 + 40) = 2b + 130 \ , \ b < 430,$$

$$N = 900 + (2)(25 + 1) = 952,$$

$$h = 990/900 = 1.10000,$$

$$N/G_m = 952/990 = 0.96162. \qquad (12.4.25)$$

From (12.2.21) and (12.2.22), for any decoder guard space G_{db}, the corresponding channel guard space G_{cb} is

$$G_{cb} = 2G_{db} + W. \qquad (12.4.26)$$

Also, from (12.2.22) and (12.2.29),

$$g = \left\lceil \frac{h(B_c + W)}{n_E} \right\rceil = \left\lceil \frac{G_{cb} - W}{n_E} \right\rceil. \qquad (12.4.27)$$

Thus, for the range of G_{db} in (12.4.25),

$$G_{cb} = 1052,\ 1054,\ \dots,\ 2772,$$

$$G_c = G_{cb}^{\max} = 2772,$$

$$g = 10,\ 11,\ \dots,\ 90,$$

$$g^{\max} = \left\lceil \frac{(1.1)(1008 + 792)}{22} \right\rceil = 90. \qquad (12.4.28)$$

From (10.6.1), (12.4.15), and (10.7.9), the performance of the compound-concatenated system in correcting bursts is given by

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)[1 - P(F)] + P(F), \qquad (12.4.29)$$

$$P(F) \approx 1 - \sum_{j=0}^{J-t} \binom{n_E}{j} q_2^{\,j} (1 - q_2)^{n_E - j}, \qquad (12.4.30)$$

$$P(E \mid \text{no } F) \approx 1 - (1 - p_2)^g. \qquad (12.4.31)$$

For every p_o and q_o, the corresponding values of p_2 and q_2 are found by substituting the parameters $n_E = 22$ and $t = 3$ of the inner convolutional code into (12.2.23) and (12.2.24). We obtain the probability of failure P(F) of the compound-concatenated system by substituting the parameters of the Gallager code, $J = 7$, $n_E = 29$, and $t = 1$, into (12.4.30). The result is shown in curve II of Fig. 12.4.7.

By substituting the extreme values $g = 10, 90$ into (12.4.31), we obtain the limits of $P(E \mid \text{no } F)$ as shown in curves V and VI of Fig. 12.4.3. Combining P(F) and $P(E \mid \text{no } F)$ as in (12.4.29), we obtain the performance $P(E \mid \text{burst})$ of the compound-concatenated system as shown in curves V and VI of Figs. 12.4.4, 12.4.5, 12.4.6.

12.4.d. Comparison

There are many obvious differences between Gallager codes, which are adaptive, and interleaved block codes and diffuse codes, which are non-adaptive. First, the performance of Gallager codes is described by bounds corresponding to the shortest and longest possible guard space requirements. Second, the performance of Gallager codes is highly dependent upon the probability of failure, P(F), which in turn is a function of the channel burst mode error rate q_o. Non-adaptive codes, on the other hand, perform equally well for all burst densities. Third, the optimum choice of random error correcting capability t for the adaptive code is found by correlating the curves for $P(E \mid \text{burst})$ and $P(E \mid \text{random})$, and this optimum choice can be any allowable value of t.

By using a Gallager code as the outer code of a compound-concatenated system, a great improvement in performance can be realized. However, this is only true when P(F) is small. As q_o increases, P(F) increases until a compound-concatenated system actually has inferior performance. For the choice of codes in this section, this transition occurs for $q_o > 0.08$, and the approaching transition is apparent in Figs. 12.4.4, 12.4.5, and 12.4.6. The reason for this is that, from Figs. 12.4.2 and 12.4.7, with $q_o > 0.08$, P(F) is larger for a compound-concatenated system than for a Gallager code alone.

12.5. Burst-Trapping Codes

Burst-trapping codes, originally described in Chapter 9, are another adaptive burst correcting technique. They are derived from a systematic (n,k) parity-check code with rate R_o as in (9.2.1),

$$R_o = \frac{k}{n} = \frac{x-1}{x}. \tag{12.5.1}$$

From (9.2.2), these codes have burst correcting capability B_m such that

$$B_m = B_c = vn. \tag{12.5.2}$$

For bursts spanning y blocks, y = 1, 2, ..., v, their guard space requirement G_{dy} is adaptive and is given by (9.3.1),

$$G_{dy} = (x-1)yn. \tag{12.5.3}$$

From (9.3.2), the corresponding adaptive channel guard space is

$$G_{cy} = G_{dy}, \tag{12.5.4}$$

where $G_{cy} \leq G_c$ and $G_{dy} \leq G_m$. The complexity of the decoder of a burst-trapping code is defined by (9.4.1), (9.4.5), and (9.4.6),

$$N = (x-1)\,v\,(k+1) + n, \tag{12.5.5}$$

$$N_T = n, \tag{12.5.6}$$

$$N_A = k. \tag{12.5.7}$$

In the channel random mode, code performance is given by (9.5.4), (9.5.5), (9.5.6), and (9.5.7),

$$P(E \mid \text{random}) = P(E \mid \text{no } A)[1 - P(A)] + P(E \mid A)\,P(A), \tag{12.5.8}$$

$$P(A) \approx \sum_{j=t+1}^{d-t-1} \binom{n}{j} p_o^{\,j} (1-p_o)^{n-j}, \tag{12.5.9}$$

$$P(E \mid A) \approx 1 - (1 - p_o)^{G_{dy}}, \qquad (12.5.10)$$

$$P(E \mid \text{no } A) = 1 - \sum_{j=0}^{d-t-1} \binom{n}{j} p_o^{\,j} (1 - p_o)^{n-j}. \qquad (12.5.11)$$

In the channel burst mode, performance is given by (9.5.1), (9.5.2), and (9.5.3),

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)[1 - P(F)] + P(F), \qquad (12.5.12)$$

$$P(F) = P_d\, P_N,$$

$$P_N = 2^{k-n} \sum_{j=0}^{t} \binom{n}{j},$$

$$P_d = 1 - \sum_{j=0}^{d-t-1} \binom{n}{j} q_o^{\,j} (1 - q_o)^{n-j}, \qquad (12.5.13)$$

$$P(E \mid \text{no } F) \approx 1 - (1 - p_o)^{G_{dy}}. \qquad (12.5.14)$$

12.5.a. Numerical Example

Consider the burst-trapping code derived from a systematic (30,15) shortened parity-check code [4] with minimum distance $d = 7$. Since this is a rate $\frac{1}{2}$ code, $x = 2$. With $v = 34$, we obtain

$$B_m = B_c = (30)(34) = 1020,$$

$$G_{dy} = G_{cy} = 30y = 30,\ 60,\ \ldots,\ 1020\ ,\ y \leq 34,$$

$$N = (1)(34)(16) + 30 = 574,$$

$$N/G_m = 574/1020 = 0.56275,$$

$$N_T = 30,$$

$$N_A = 15. \qquad (12.5.15)$$

For this code, $n = 30$, $k = 15$, and $d = 7$ are constant parameters, while random error correcting capability may span the range $t = 1, 2, 3$ and decoder guard space requirement is bounded by $G_{dy} = 30, 1020$. By substituting these parameters into (12.5.8), (12.5.9), (12.5.10), and (12.5.11), we obtain the random mode performance curves $P(E \mid \text{random})$ for this code, Fig. 12.5.1. Note that the largest choice of t, $t = 3$, does not permit a false alarm, so the burst mode of the decoder cannot be used to correct random error patterns with large weight. Thus, the overall performance of the code in correcting random errors is actually reduced when it has maximum error correcting capability t.

Substituting the code parameters into (12.5.13) gives the curves in Fig. 12.5.2 for the probability of failure, P(F), and into (12.5.14) gives curves I and II in Fig. 12.5.3 for the probability of decoding error given a detected burst, $P(E \mid \text{no } F)$. Combining the results of Figs. 12.5.2 and 12.5.3 as in (12.5.12), we obtain the performance $P(E \mid \text{burst})$ of the burst-trapping code as shown in curves I and II of Figs. 12.5.4, 12.5.5, and 12.5.6.

12.5.b. Compound-Concatenated System With Inner Block Code

Consider a compound-concatenated system whose outer code belongs to the same class as the burst-trapping code of Section 12.5.a. and whose inner code is the extended Golay code of Section 12.2.b.. With $v = 17$, we obtain

$$B_m = (30)(17) = 510,$$

$$G_{dy} = 30y = 30, 60, \ldots, 510 \ , \ y \leq 17,$$

$$N = (1)(17)(16) + 30 = 302,$$

$$h = 510/510 = 1.00000,$$

$$N/G_m = 302/510 = 0.59216. \qquad (12.5.16)$$

Analogous to (12.4.20), the adaptive channel guard space requirement is $G_{cy} \approx 2G_{dy}$ such that

$$G_{cy} = \lceil hf \rceil n = 24\lceil hf \rceil \ , \ \lceil hf \rceil \text{ an integer.} \qquad (12.5.17)$$

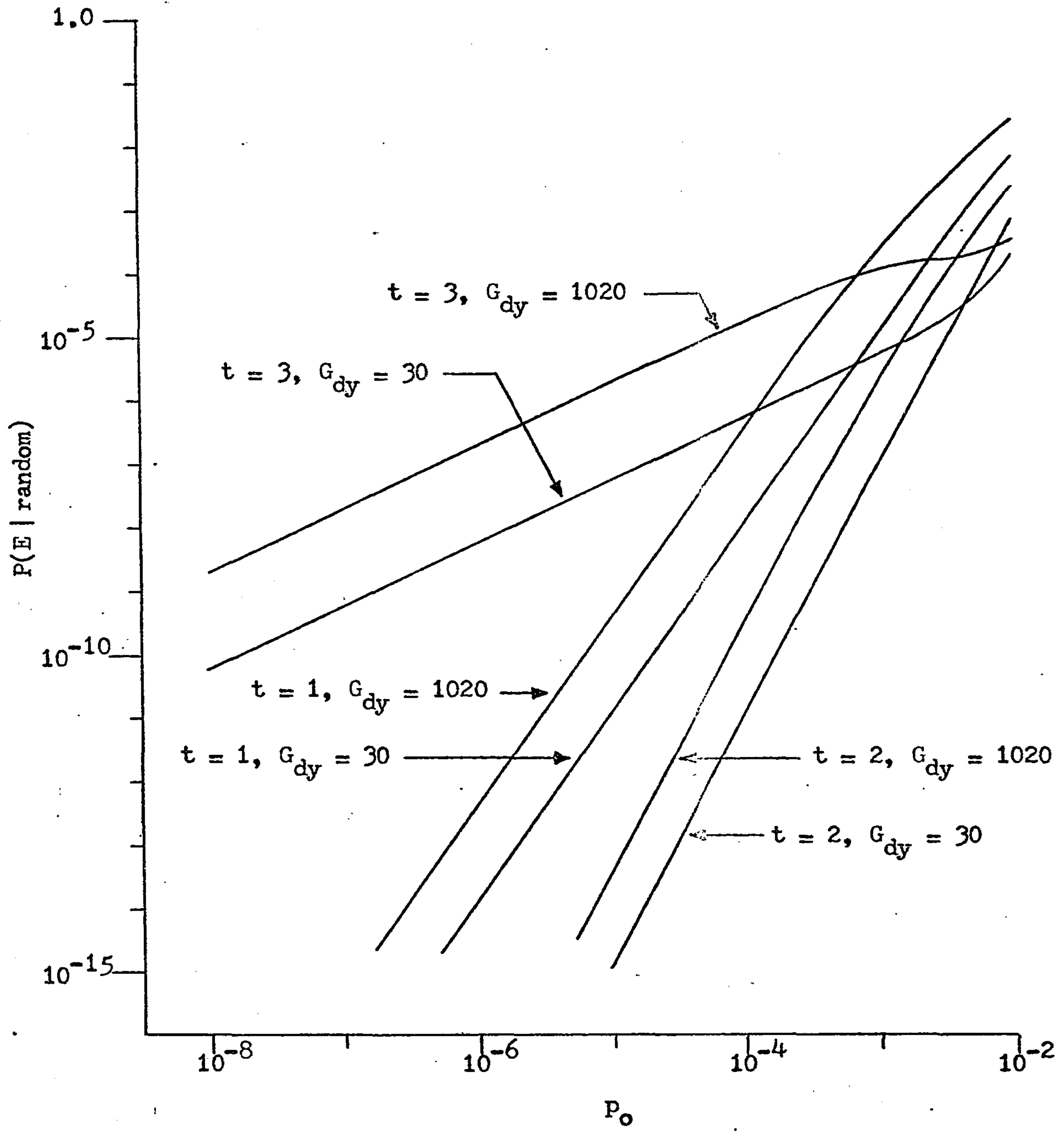


Figure 12.5.1 : Performance of Burst-Trapping Code in Channel Random Mode.

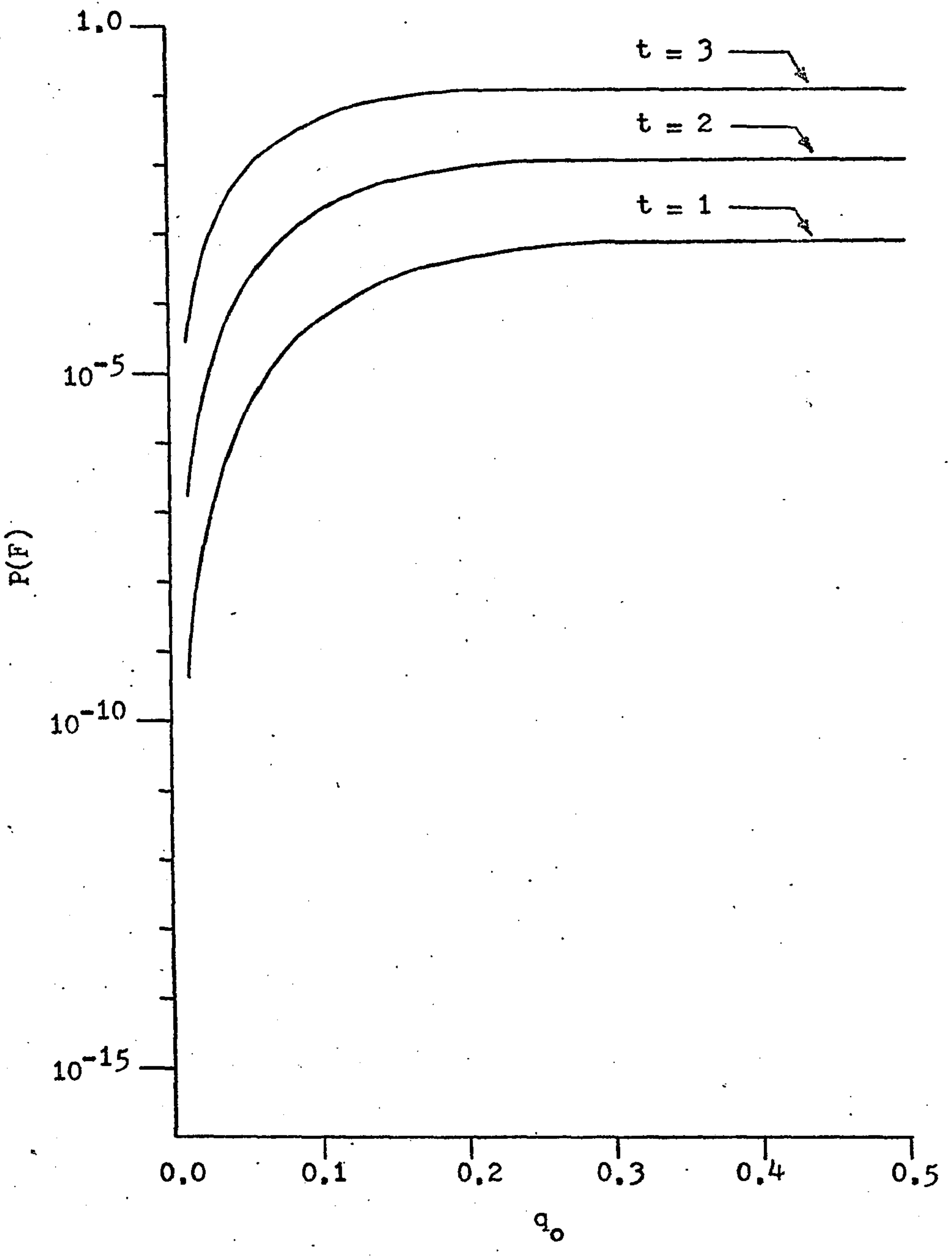
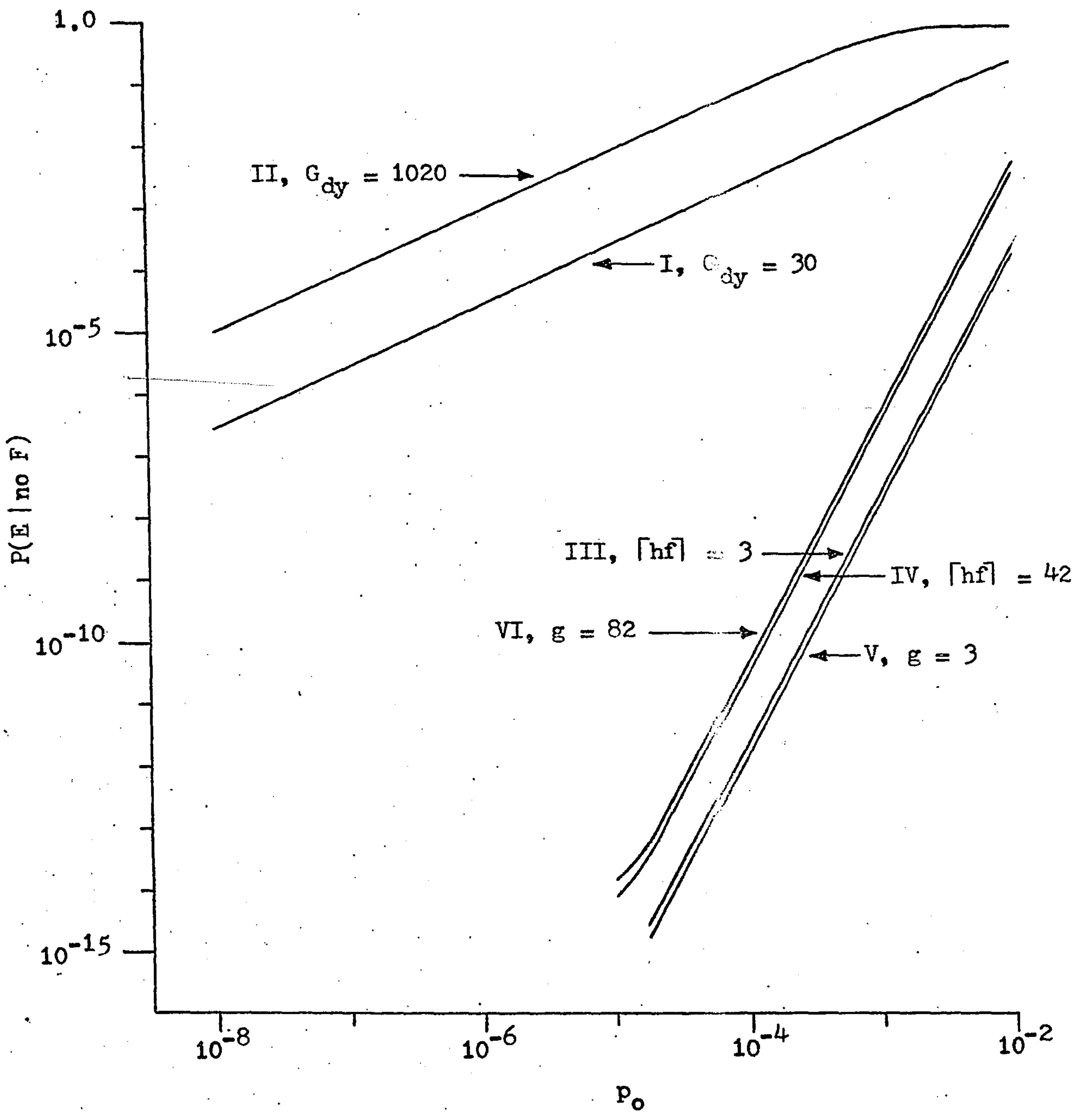


Figure 12.5.2 : Probability of Failure of Burst-Trapping Code.

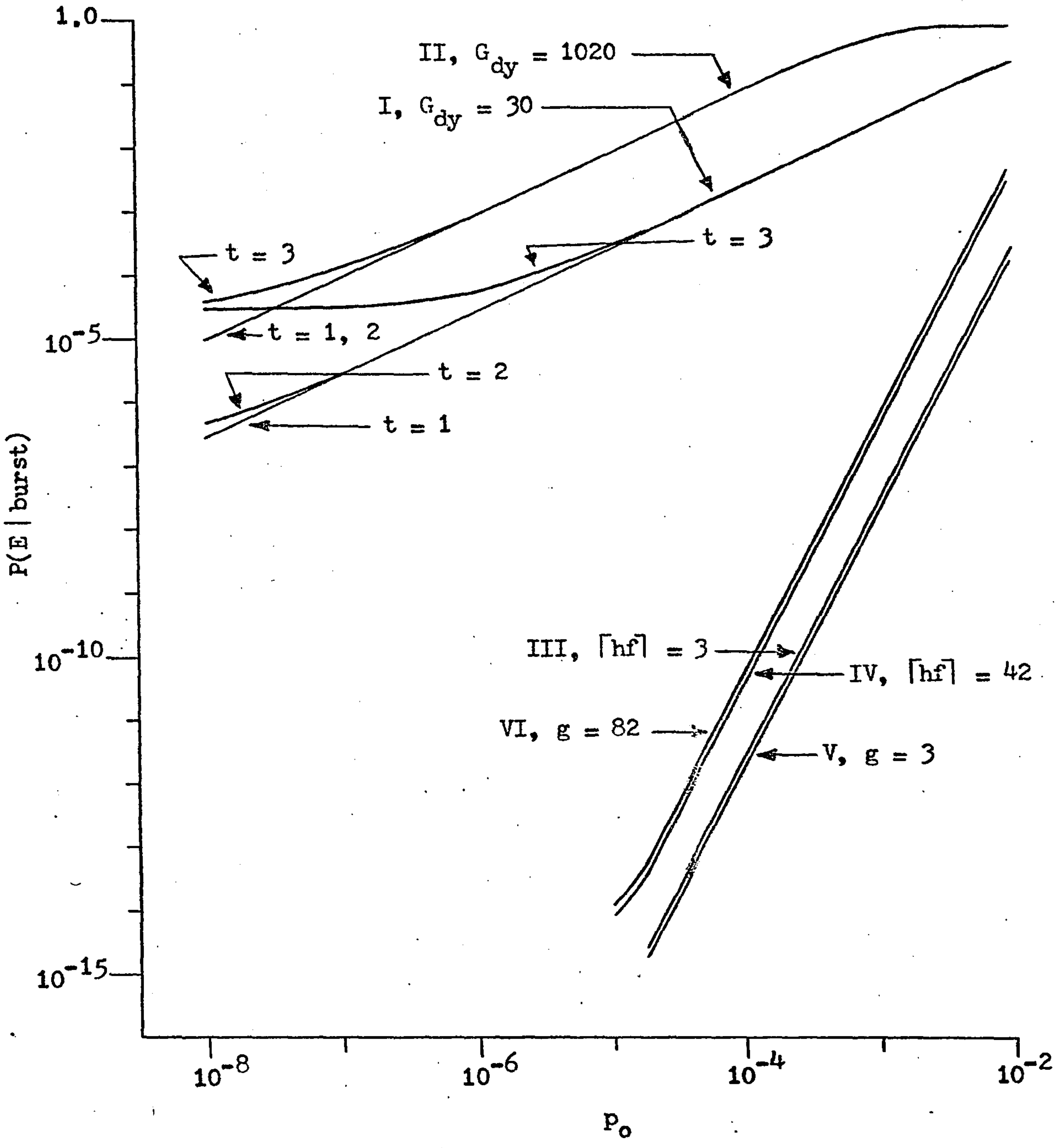


I, II: Limits for burst-trapping code alone.

III, IV: Limits for inner block code and outer burst-trapping code.

V, VI: Limits for inner convolutional code and outer burst-trapping code.

Figure 12.5.3 : Performance of Burst-Trapping Code Given a Detected Burst.

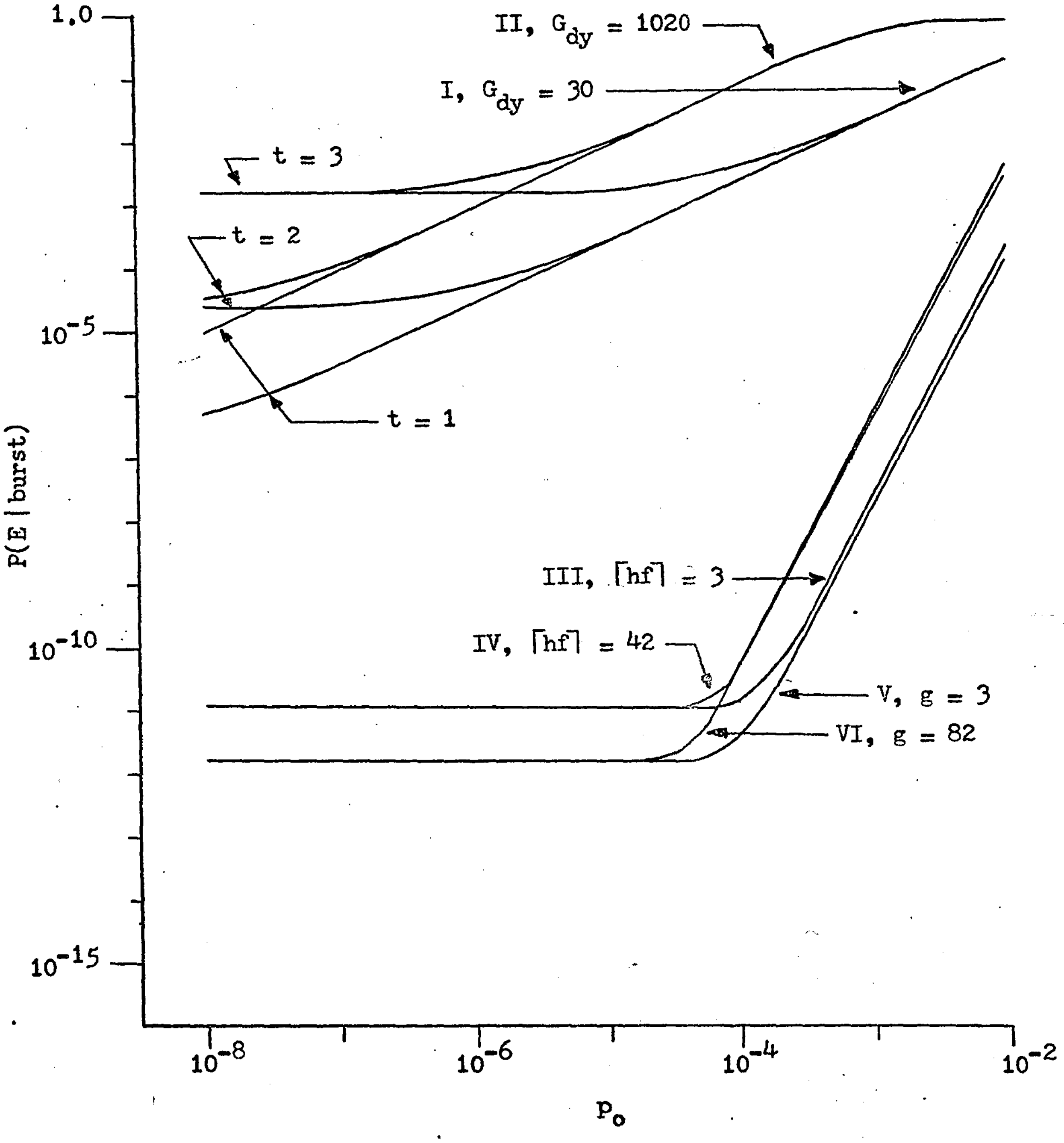


I, II: Limits for burst-trapping code alone.

III, IV: Limits for inner block code and outer burst-trapping code.

V, VI: Limits for inner convolutional code and outer burst-trapping code.

Figure 12.5.4 : Performance of Burst-Trapping Code Given That $q_o = 0.010$.

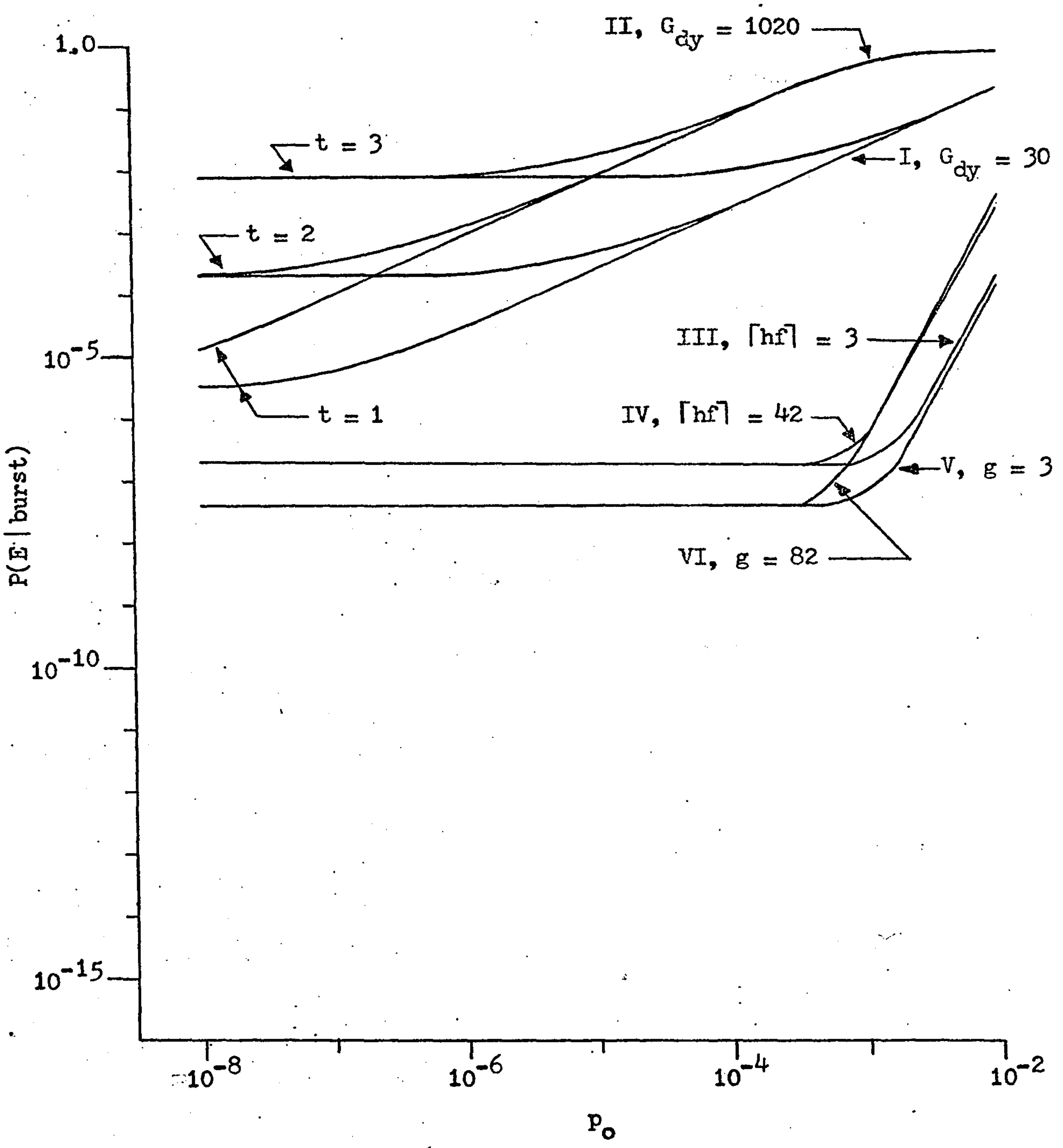


I, II: Limits for burst-trapping code alone.

III, IV: Limits for inner block code and outer burst-trapping code.

V, VI: Limits for inner convolutional code and outer burst-trapping code.

Figure 12.5.5 : Performance of Burst-Trapping Code Given That $q_o = 0.030$.



I, II: Limits for burst-trapping code alone.

III, IV: Limits for inner block code and outer burst-trapping code.

V, VI: Limits for inner convolutional code and outer burst-trapping code.

Figure 12.5.6 : Performance of Burst-Trapping Code Given That $q_o = 0.050$.

For the range of G_{dy} in (12.5.16), we have

$$G_{cy} = 72, 96, \ldots, 1008,$$

$$\lceil hf \rceil = 3, 4, \ldots, 42. \qquad (12.5.18)$$

From (10.6.1), (12.5.13), and (10.6.5), the performance of the compound-concatenated system in correcting bursts is given by

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)[1 - P(F)] + P(F), \qquad (12.5.19)$$

$$P(F) = P_d \, P_N,$$

$$P_d = 1 - \sum_{j=0}^{d-t-1} \binom{n}{j} q_1^{\,j} (1 - q_1)^{n-j}, \qquad (12.5.20)$$

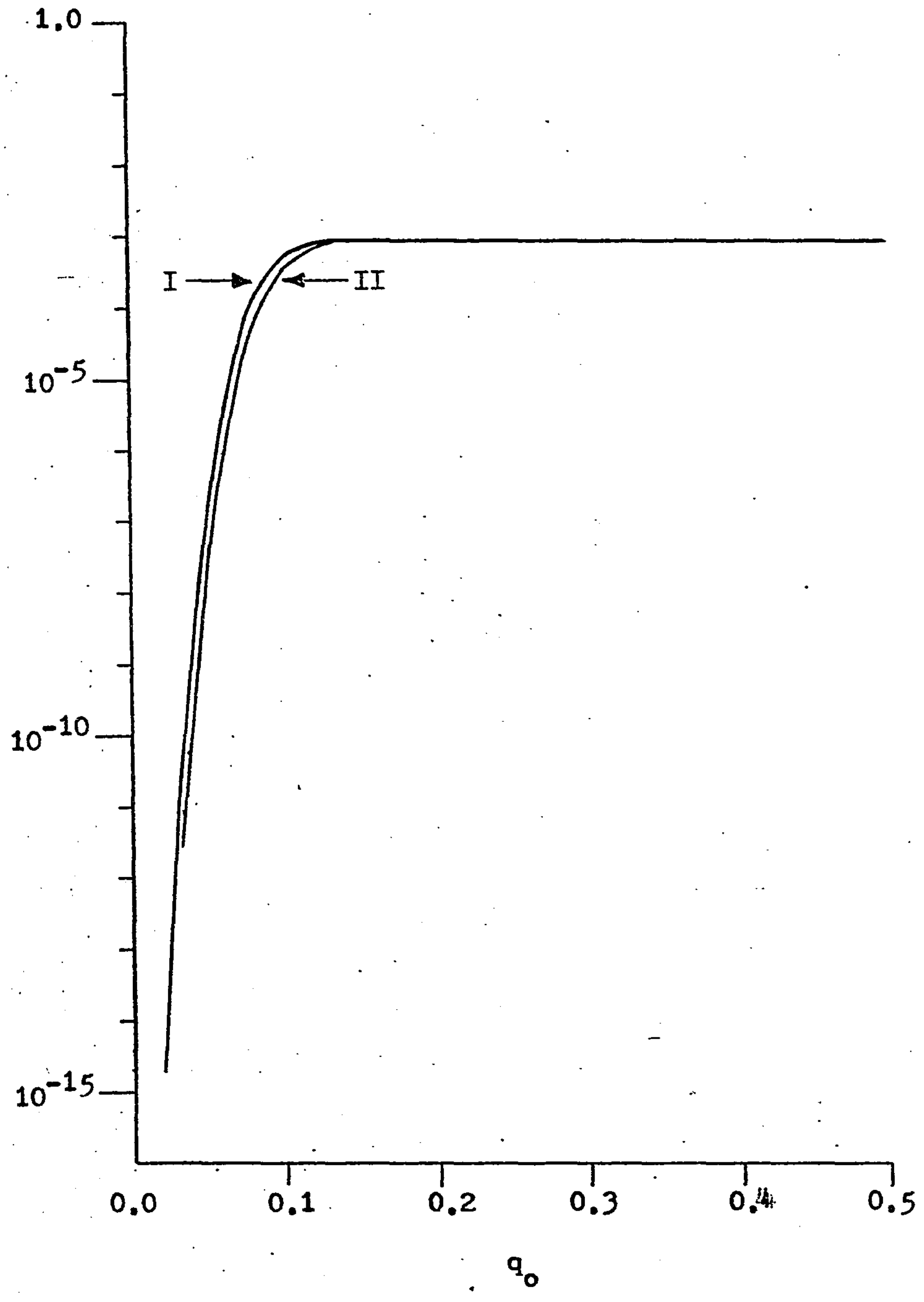
$$P(E \mid \text{no } F) \approx 1 - (1 - p_1)^{\lceil hf \rceil}. \qquad (12.5.21)$$

For every p_o and q_o, the corresponding values of p_1 and q_1 are found by substituting the parameters $n = 24$ and $t = 3$ of the inner block code into (12.2.13) and (12.2.14). The probability of failure P(F) of the compound-concatenated system is found by substituting the parameters of the burst-trapping code, $n = 30$, $k = 15$, $d = 7$, and $t = 1$, into (12.5.20) and (12.5.13). The result is shown in curve I of Fig. 12.5.7. We choose $t = 1$ because it is known from Fig. 12.5.2 to yield the best curve for P(F).

By substituting the extreme values $\lceil hf \rceil$ 3,42 into (12.5.21), we obtain the limits of $P(E \mid \text{no } F)$, the probability of a decoding error given a detected burst, shown in curves III and IV of Fig. 12.5.3. Combining P(F) and $P(E \mid \text{no } F)$ as in (12.5.19), we obtain the performance $P(E \mid \text{burst})$ of the compound-concatenated system as shown in curves III and IV of Figs. 12.5.4, 12.5.5, and 12.5.6.

12.5.c. Compound-Concatenated System With Inner Convolutional Code

Consider a compound-concatenated system whose outer code belongs to



I: With inner block code.

II: With inner convolutional code.

Figure 12.5.7 : Probability of Failure of Compound-Concatenated System With Outer Burst-Trapping Code.

the same class as the burst-trapping code of Section 12.5.a. and whose inner code is the convolutional code of Section 12.2.c.. With $v = 30$ and $W = 792$, we obtain

$$\begin{aligned} B_m &= (30)(30) = \tfrac{1}{2}(1008 + 792) = 900, \\ G_{dy} &= 30y = 30,\ 60,\ \ldots,\ 900\ ,\ y \le 30, \\ N &= (1)(30)(16) + 30 = 510, \\ h &= 900/900 = 1.00000, \\ N/G_m &= 510/900 = 0.56667. \end{aligned} \tag{12.5.22}$$

Analogous to (12.4.26), for any decoder guard space G_{dy}, the corresponding channel guard space G_{cy} is

$$G_{cy} = 2G_{dy} + W. \tag{12.5.23}$$

Also, from (12.4.27),

$$g = \left\lceil \frac{h(B_c + W)}{n_E} \right\rceil = \left\lceil \frac{G_{cy} - W}{n_E} \right\rceil. \tag{12.5.24}$$

Thus, for the range of G_{dy} in (12.5.22),

$$\begin{aligned} G_{cy} &= 852,\ 912,\ \ldots,\ 2592, \\ G_c &= G_{cy}^{\max} = 2592, \\ g &= 3,\ 4,\ \ldots,\ 82, \\ g^{\max} &= \left\lceil \frac{(1)(1008 + 792)}{22} \right\rceil = 82. \end{aligned} \tag{12.5.25}$$

From (10.6.1), (12.5.13), and (10.7.9), the performance of the compound-concatenated system in correcting bursts is given by

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)[1 - P(F)] + P(F), \tag{12.5.26}$$

$$P(F) = P_d\, P_N,$$

$$P_d = 1 - \sum_{j=0}^{d-t-1} \binom{n}{j} q_2^{\,j} (1 - q_2)^{n-j}, \tag{12.5.27}$$

$$P(E \mid \text{no } F) \approx 1 - (1 - p_2)^g. \qquad (12.5.28)$$

For every p_o and q_o, the corresponding values of p_2 and q_2 are found by substituting the parameters $n_E = 22$ and $t = 3$ of the inner convolutional code into (12.2.23) and (12.2.24). We obtain the probability of failure P(F) of the compound-concatenated system by substituting the parameters of the burst-trapping code, $n = 30$, $k = 15$, $d = 7$, and $t = 1$, into (12.5.27) and (12.5.13). The result is shown in curve II of Fig. 12.5.7.

By substituting the extreme values $g = 3$, 82 into (12.5.28), we obtain the limits of $P(E \mid \text{no } F)$ as shown in curves V and VI of Fig. 12.5.3. Combining P(F) and $P(E \mid \text{no } F)$ as in (12.5.26), we obtain the performance $P(E \mid \text{burst})$ of the compound-concatenated system as shown in curves V and VI of Figs. 12.5.4, 12.5.5, and 12.5.6.

12.5.d. Comparison

Adaptive burst-trapping codes are seen to be a block-code-analog of adaptive Gallager codes. They suffer the same disadvantage of high dependency on the channel burst mode error rate and have the same advantages of reduced guard space requirement and reduced complexity. Since burst-trapping codes have a lower probability of failure than equivalent Gallager codes, they tend to have superior performance. In addition, they have a much lower storage requirement than any other burst correcting code described in this thesis.

12.6. GSA Codes

GSA codes, originally described in Chapter 11, are an adaptive burst correcting technique with the structure of a compound-concatenated system. The random mode of the system is provided by an inner (n_i,k_i) parity-check code with minimum distance d and error correcting capability t. The burst modes of the system are obtained from a modified burst-trapping code derived from a trivial (n_o,k_o) systematic parity-check code for which the parity bits are always zero. From (11.2.2),

$$\frac{k_o}{n_o} = \frac{x-1}{x}. \tag{12.6.1}$$

For some integer f, (11.2.9) gives the channel burst length B_c as

$$B_c = fn_i. \tag{12.6.2}$$

From (11.4.1), the outer code has burst correcting capability B_m such that

$$B_m = fk_i = rbn_o, \tag{12.6.3}$$

where r is known as the interleaving degree and b is the number of algorithms by which bursts can be decoded, $b \leq 4$. The adaptive guard space requirement G_{oy} is, from (11.4.2),

$$G_{oy} = (x-1)ryn_o = (x-1)y\frac{f}{b}k_i \;,\; y = 1, 2, \ldots, b, \tag{12.6.4}$$

and from (11.2.13), the corresponding adaptive channel guard space G_{iy} is

$$G_{iy} = (x-1)y\frac{f}{b}n_i, \tag{12.6.5}$$

where $G_{oy} \leq G_m$ and $G_{iy} \leq G_c$. The complexity of the outer decoder is defined by (11.6.1), (11.6.9), and (11.6.12),

$$N = r(k_o+1)(bx-1) + rbk_o + n_o + 1, \tag{12.6.6}$$

$$N_T = [l_E - b + 1 + \sum_{j=1}^{b}(x-1)rj](n_o - k_o) + rbk_o, \tag{12.6.7}$$

$$N_A \geq (l_E - 1)(n_o - k_o), \tag{12.6.8}$$

where l_E is called the effective length of the GSA code.

In the channel random mode, the performance of the code is given by (11.7.4), (11.7.6), (11.7.7), (11.7.8), and (11.7.9),

$$P(E \mid \text{random}) = P(E \mid \text{no } A)[1 - P(A)] + P(E \mid A)\, P(A), \tag{12.6.9}$$

$$P(A) \approx 1 - [1 - \sum_{j=t+1}^{d-t-1} \binom{n_i}{j} p_o^{\,j} (1 - p_o)^{n_i - j}]^{f/rb}, \tag{12.6.10}$$

$$P(E \mid A) \approx 1 - (1 - P_b)^{(x-1)yf/b}, \tag{12.6.11}$$

$$P_b = 1 - \sum_{j=0}^{t} \binom{n_i}{j} p_o^{\,j} (1 - p_o)^{n_i - j}, \tag{12.6.12}$$

$$P(E \mid \text{no } A) = 1 - [\sum_{j=0}^{d-t-1} \binom{n_i}{j} p_o^{\,j} (1 - p_o)^{n_i - j}]^{f/rb}. \tag{12.6.13}$$

In the channel burst mode, performance is given by (11.7.1), (11.7.2), (11.7.3), (11.7.4), and (11.7.5),

$$P(E \mid \text{burst}) = P(E \mid \text{no } F)[1 - P(F)] + P(F), \tag{12.6.14}$$

$$P_f = P_d\, P_N,$$

$$P_N = 2^{k_i - n_i} \sum_{j=0}^{t} \binom{n_i}{j},$$

$$P_d = 1 - \sum_{j=0}^{d-t-1} \binom{n_i}{j} q_o^{\,j} (1 - q_o)^{n_i - j}, \tag{12.6.15}$$

$$P(F) \approx (P_f)^{f/rb}, \tag{12.6.16}$$

$$P(E \mid \text{no } F) \approx 1 - (1 - P_b)^{(x-1)yf/b}. \tag{12.6.17}$$

12.6.a. Numerical Example

Consider a GSA code whose inner code is the same (30,15) shortened parity-check code used in Section 12.5.a.. This code has the parameters $n_i = 30$, $k_i = 15$, and $d = 7$. With $f = 36$, we obtain

$$B_c = (36)(30) = 1080,$$

$$B_m = (36)(15) = 540. \tag{12.6.18}$$

The outer code has the parameters $x = 3$, $n_o = 45$, $k_o = 30$, $b = 4$, and $r = 3$. From Table 11.3.1 an optimal code with $l_E = 5$ is given by

$$f(\underline{I}) = \underline{I}_1^{j-3} + \underline{I}_2^{j-6} + \underline{I}_1^{j-12} + \underline{I}_2^{j-18} + \underline{I}_2^{j-24}. \qquad (12.6.19)$$

This gives $f/b = 9$ and $f/rb = 3$, and we obtain

$$B_m = (3)(4)(45) = 540,$$

$$G_{oy} = (2)(3)(45)y = 270y = 270,\ 540,\ 810,\ 1080,$$

$$N = (3)(31)(11) + (3)(4)(30) + 45 + 1 = 1429,$$

$$N/G_m = 1429/1080 = 1.32315,$$

$$N_T = [5 - 4 + 1 + (2)(3) \sum_{j=1}^{4} j](45 - 30) + (3)(4)(30) = 990,$$

$$N_A \geq (5 - 1)(45 - 30) = 60. \qquad (12.6.20)$$

The random error correcting capability of the inner code may span the range $t = 1, 2, 3$ and the guard space requirement of the outer code is $G_{oy} = 270y$, $y = 1, 2, 3, 4$. By substituting these and the other parameters of the GSA code into (12.6.9), (12.6.10), (12.6.11), (12.6.12), and (12.6.13), we obtain the random mode performance curves $P(E \mid \text{random})$ for this code, Fig. 12.6.1. Again, because a choice of $t = 3$ does not permit a false alarm, the decoder burst mode cannot be used to correct random error patterns with large weight, resulting in reduced overall performance. This effect is not as pronounced as with a burst-trapping code, however, because the GSA code allows the false alarm to occur in any of $f/rb = 3$ blocks at the inner decoder.

Substituting the code parameters into (12.6.15) and (12.6.16) gives the curves in Fig. 12.6.2 for the probability of failure P(F). Because a failure must occur simultaneously in f/rb blocks at the inner decoder, P(F) is much less for a GSA code than for other adaptive codes or compound-concatenated systems.

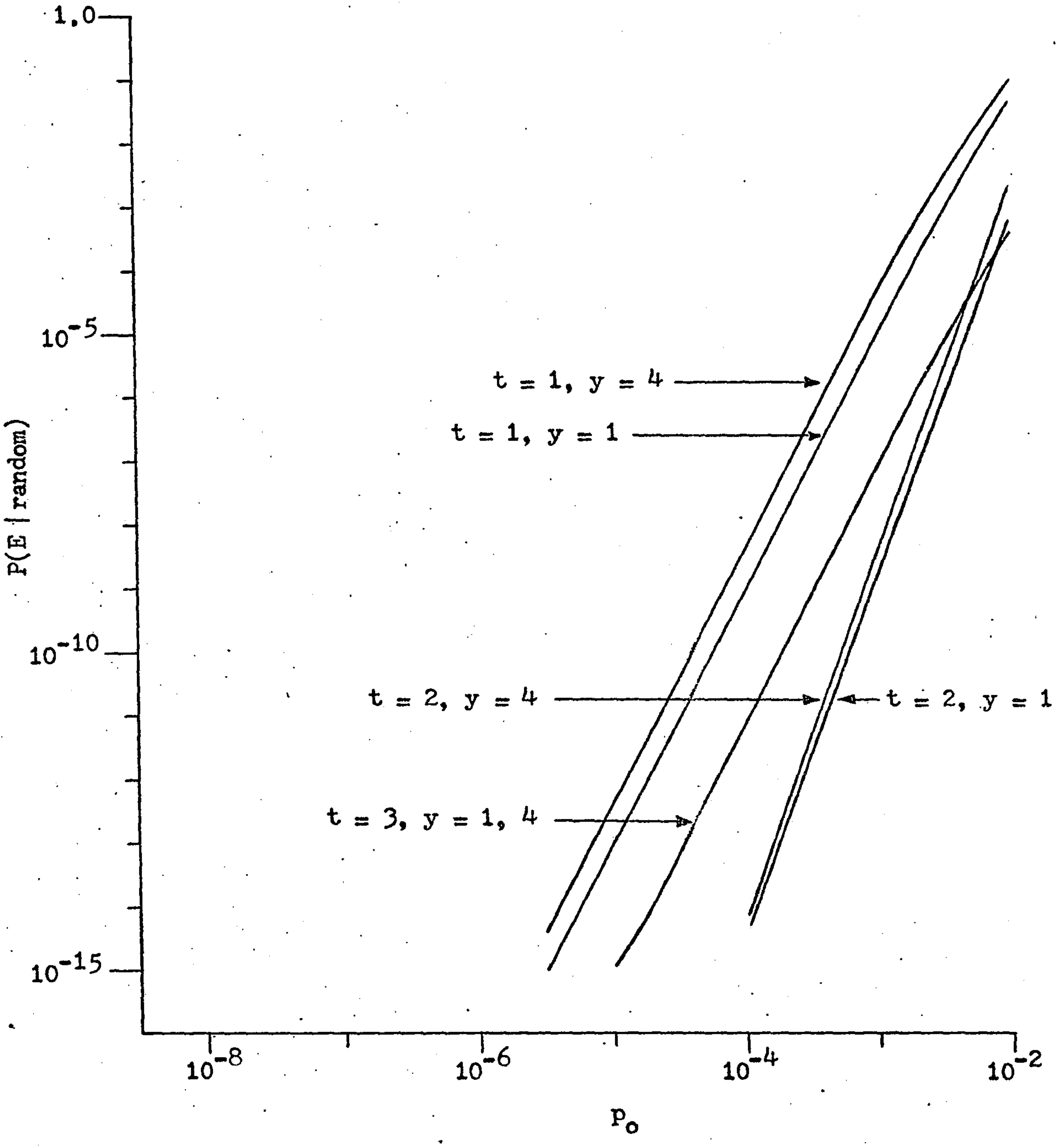


Figure 12.6.1 : Performance of GSA Code in Channel Random Mode.

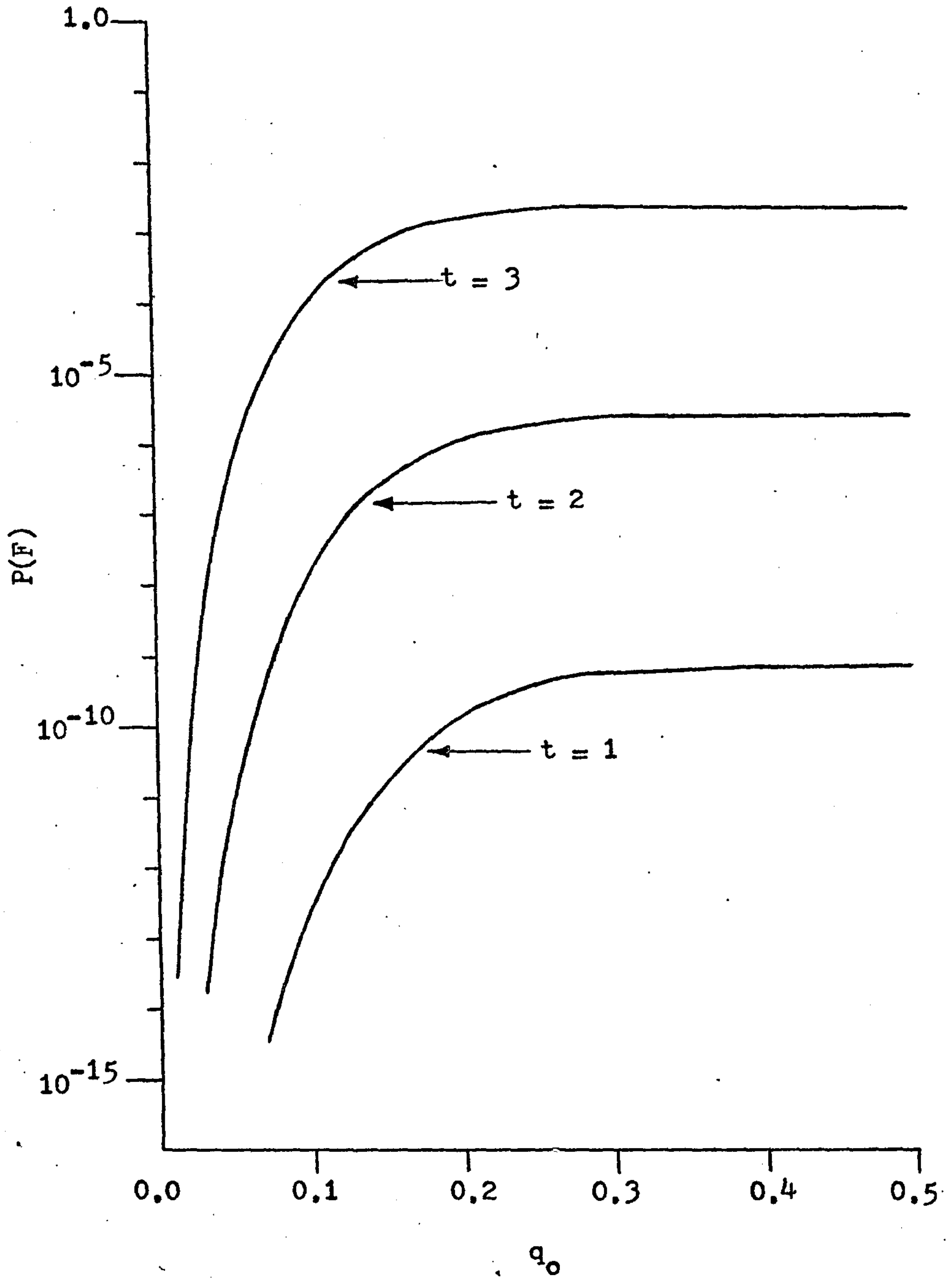


Figure 12.6.2 : Probability of Failure of GSA Code.

(12.6.12) and (12.6.17) yield the curves for P(E | no F), the probability of a decoding error given a detected burst, in Fig. 12.6.3. Combining P(F) and P(E | no F) as in (12.6.14), we obtain the code performance P(E | burst) in Figs. 12.6.4, 12.6.5, and 12.6.6.

12.6.b. Comparison

With a choice of error correcting capability $t = 1$ at the inner decoder, this GSA code is not as effective as the other compound-concatenated systems when the channel burst mode error rate q_o is low. However, since P(F) for the GSA code is comparatively low, its performance does not deteriorate nearly as badly as that of the other adaptive compound-concatenated systems as q_o increases. It should also be noted that this GSA code has higher rate than the other systems; i.e., 1/3 compared to 1/4. The main disadvantage of GSA codes is their relatively large complexity, particularly in terms of the number of tapped shift register stages.

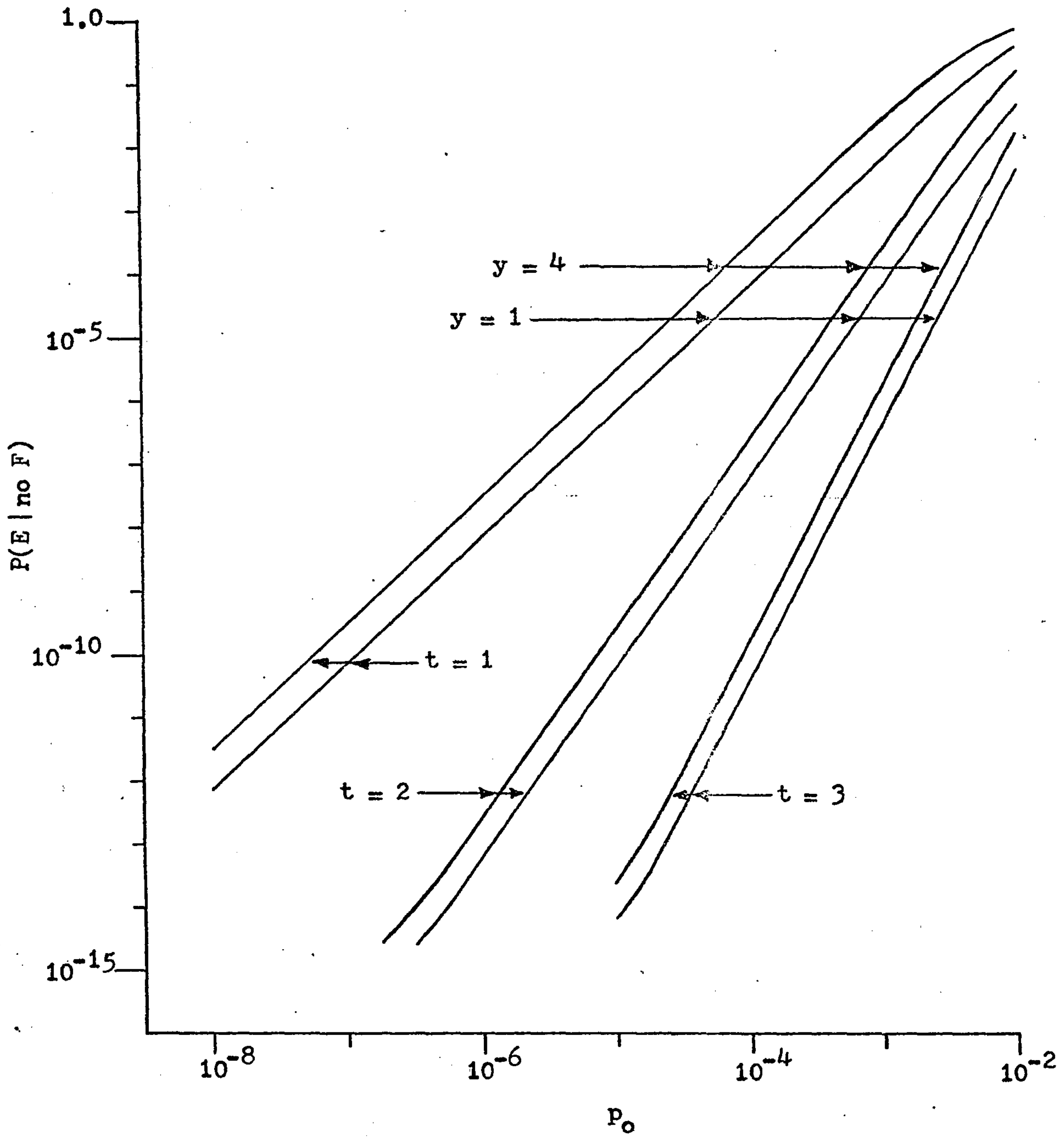


Figure 12.6.3 : Performance of GSA Code Given a Detected Burst.

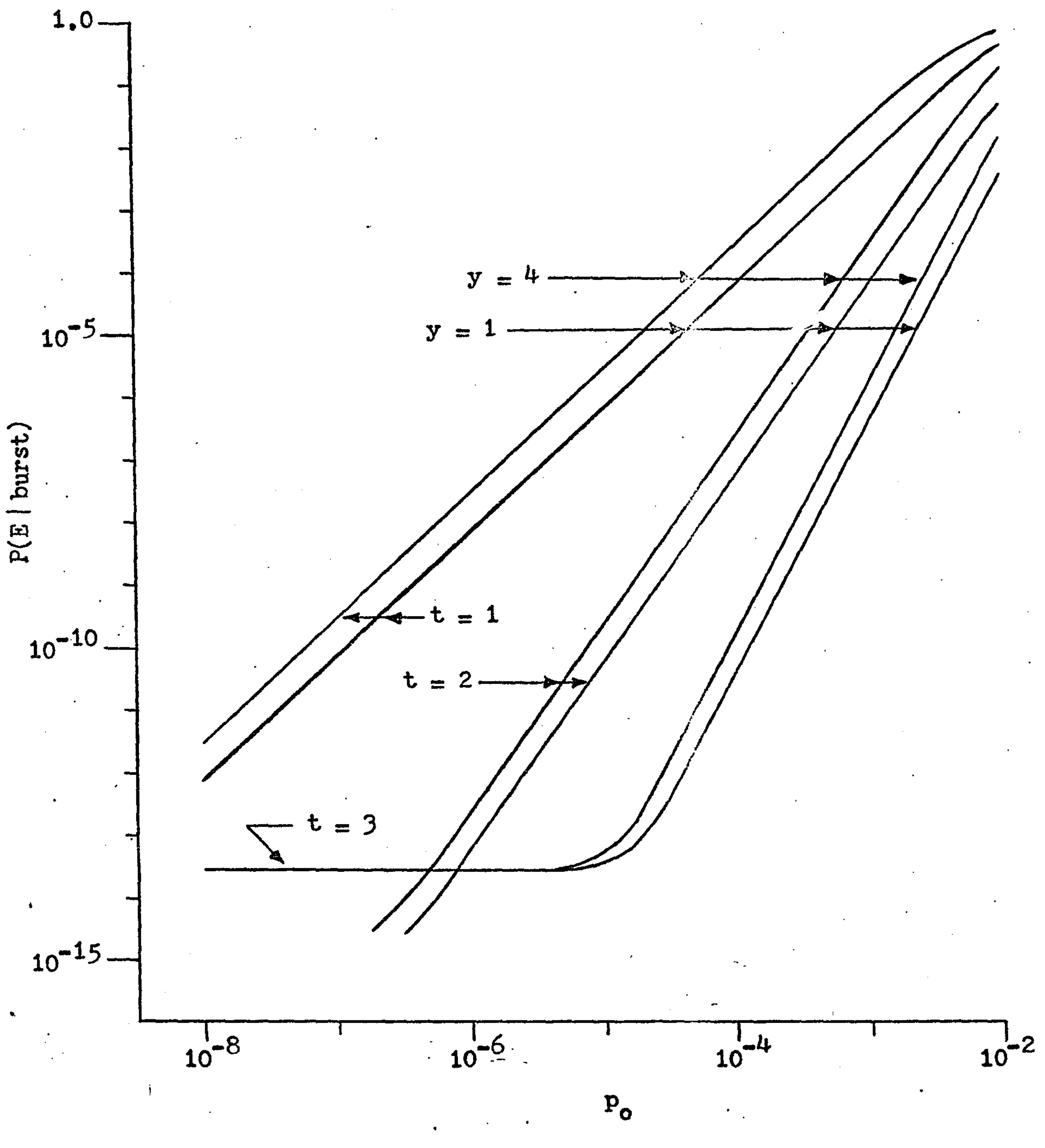


Figure 12.6.4 : Performance of GSA Code Given That $q_o = 0.010$.

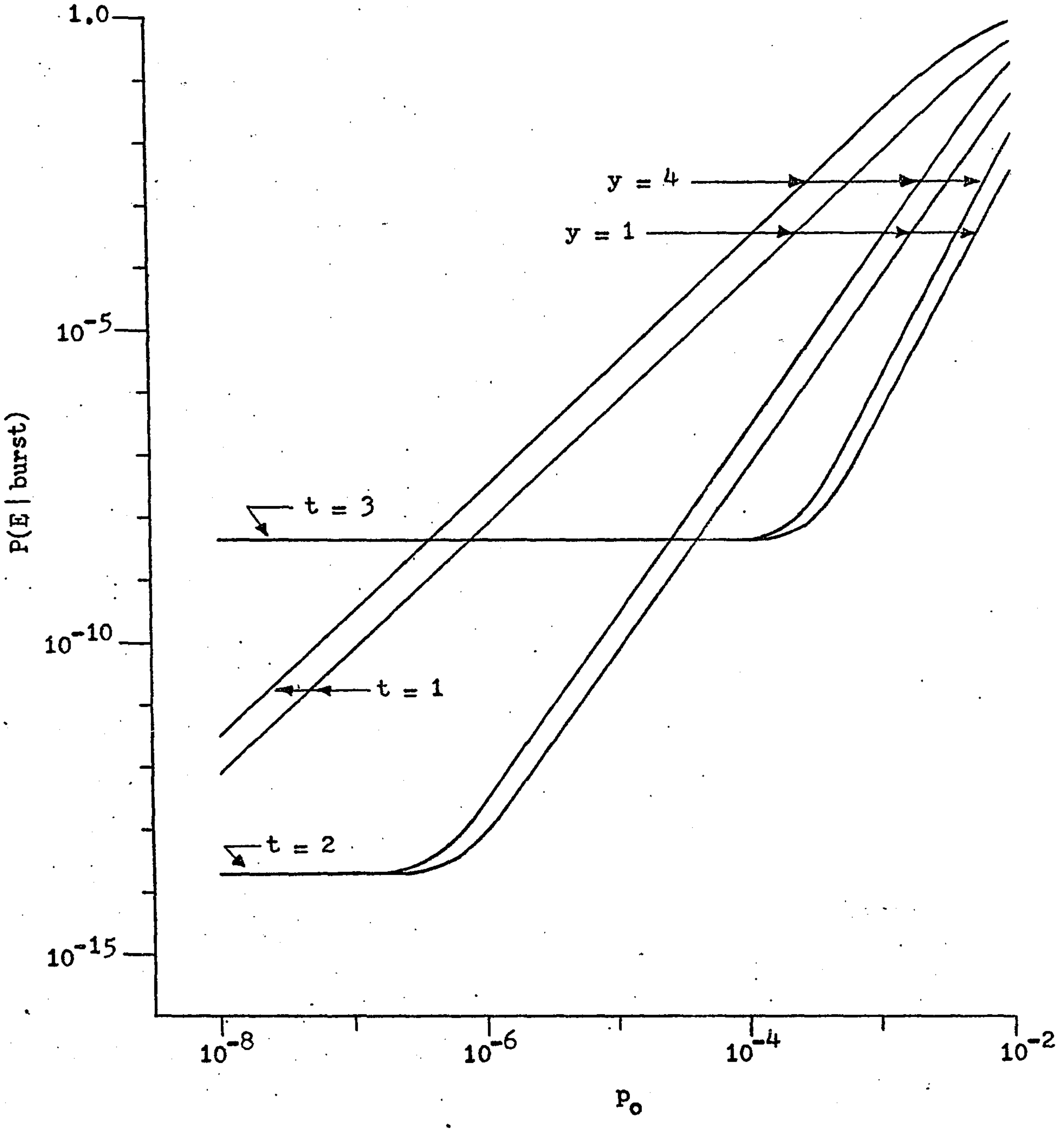


Figure 12.6.5 : Performance of GSA Code Given That $q_o = 0.030$.

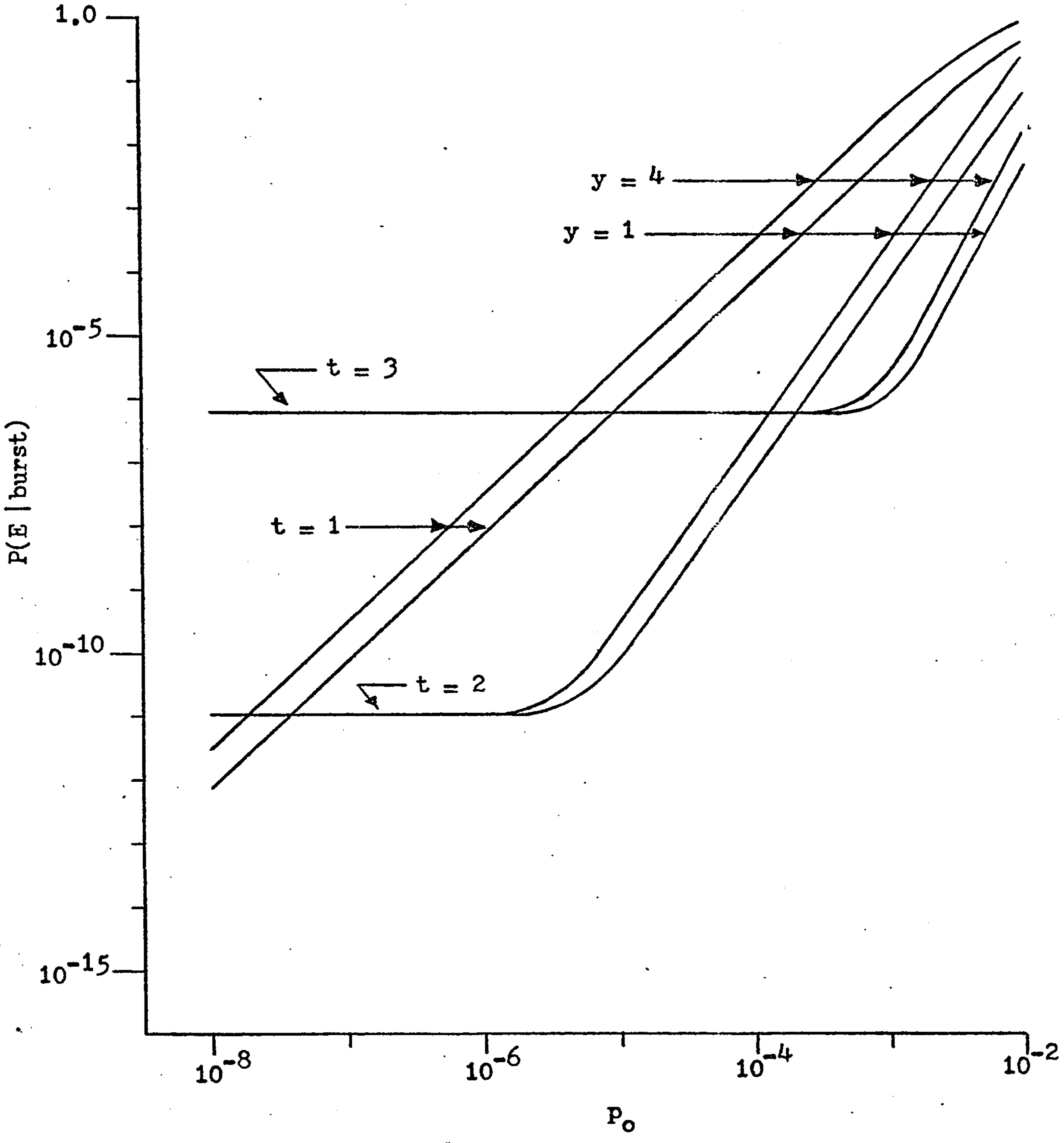


Figure 12.6.6 : Performance of GSA Code Given That $q_o = 0.050$.

13. CONCLUSION

The purpose of this thesis was to describe various error control techniques for the compound channel, particularly the two new techniques called compound-concatenated systems and GSA codes. As a basis for comparison between these codes, we used the criteria of decoder complexity and probability of decoding error, important measures of system cost and reliability. To simplify the comparison, we attempted to unify the descriptions of the principles of operation of interleaved block codes, diffuse codes, Gallager codes, and burst-trapping codes, as well as of the two new techniques, and to obtain relatively simple expressions for their complexity and performance in a standardized format. In the last chapter, a summary and numerical evaluation of these various codes was presented.

Burst correcting codes, though not presently enjoying widespread application due to the greater cost effectiveness of retransmission requests in low speed situations, will likely appear ever more attractive to system designers as hardware costs decrease and transmission rate increases. The most important parameter in the selection of a burst correcting code is its compatibility with the characteristics of the particular transmission channel. For example, if bursts on the channel are known to be of high error density, then we have seen that an adaptive code is probably unsuitable because of its poor performance under such a condition. On the other hand, with low error density, adaptive codes give performance comparable to that of non-adaptive codes but with lower complexity. If a very high degree of reliability is important and a reduction in efficiency is an acceptable trade-off, then a compound-concatenated system is appropriate, particularly if the outer code is non-adaptive. If bursts are very common, then a GSA code may be suitable because of its adjacent adaptive guard space and relatively high tolerance of variations in burst error density.

A great many interesting and important topics in this thesis were treated only lightly and many others were omitted completely. For example, the classes and implementations of block codes has been for two decades and is still an area of intensive research. Then there are the related problems of timing, control, and synchronization of encoders and decoders, as well as channel design and signal design. However, one factor that more directly concerns the characteristics of specific burst correcting codes, but which is very difficult to describe accurately, is robustness, the capability of a code to correct or to detect an error pattern not guaranteed to be corrected or detected by the parameters of the code. In the text of the thesis, we generally ignored the specific effects of robustness, and this is the most important factor affecting the accuracy of the performance curves in Chapter 12.

Despite their inherent inaccuracies, performance curves like those in Chapter 12 are extremely useful tools in determining the suitability of various codes. First, they give rough approximations to the performance which can actually be expected on a real channel. Second, since robustness is the most significant factor neglected in obtaining these curves, they are generally pessimistic; that is, true performance can be expected to be better than shown in the curves. Third, for adaptive codes, the optimum choice of error correcting capability in the random mode can be determined by inspection if the predominant channel random mode error rate is known.

In addition to performance, of course, a major engineering consideration is the complexity or cost of implementing these codes. It is their generally much lower complexity which is the major factor in directing current interest in burst correcting codes towards the adaptive codes, such as Gallager codes and burst-trapping codes. It is the fact that overall system complexity need not be increased, and may even be reduced, that makes compound-concatenated systems appear to be a very attractive method of improving performance on noisy channels. And it is, unfortunately, complexity which makes GSA codes unattractive in cost-sensitive applications.

For over two decades now, the major problem in the field of random error correcting codes has been the development of simple and inexpensive decoders. An important breakthrough in this area was the introduction of threshold decoding. As research intensified over the past few years in the field of burst correcting codes, the cost of decoders again appeared as the primary deterrent to widespread acceptance. Even though threshold decodable diffuse codes and Gallager codes exist, a block coding technique, burst-trapping codes, is less complex. Thus, threshold decoding in itself does not provide as significant a solution to the complexity problem for burst correcting codes as for random error correcting codes. The search, then, must continue for ever more inexpensive codes and decoding methods.

BIBLIOGRAPHY

[1] R.G. Gallager, Information Theory and Reliable Communication, New York: Wiley; 1968.

[2] E.N. Gilbert, "Capacity of a Burst-Noise Channel," Bell System Technical Journal (BSTJ), Vol. 39, pp. 1253 -- 1265; 1960.

[3] E.R. Berlekamp, Algebraic Coding Theory, New York: McGraw-Hill; 1968.

[4] W.W. Peterson, Error Correcting Codes, Cambridge, Mass.: MIT Press; 1961.

[5] C.E. Shannon, "A Mathematical Theory of Communication," BSTJ, Vol. 27, pp. 379 - 423 and 623 - 656; 1948.

[6] R.W. Hamming, "Error Detecting and Error Correcting Codes," BSTJ, Vol. 29, pp. 147 - 160; 1950.

[7] I.S. Reed, "A Class of Multiple-Error-Correcting Codes and the Decoding Scheme," IRE Transactions on Information Theory (IRE Trans. IT), Vol. IT-4, pp. 38 - 49; 1954.

[8] D.E. Muller, "Application of Boolean Algebra to Switching Circuit Design and to Error Detecting," IRE Transactions on Electronic Circuits (IRE Trans. EC), Vol. EC-3, pp. 6 - 12; 1954.

[9] P. Elias, "Error Free Coding," IRE Trans. IT, Vol. IT-4, pp. 29 - 37; 1954.

[10] D. Slepian, "A Class of Binary Signalling Alphabets," BSTJ, Vol. 35, pp. 203 - 234; 1956.

[11] A. Hocquenghem, "Codes Correcteurs d'Erreurs," Chiffres, Vol. 2, pp. 147 - 156; 1959.

[12] R.C. Bose and D.K. Ray-Chaudhuri, "On a Class of Error Correcting Binary Group Codes," Information and Control, Vol. 3, pp. 68 - 79; 1960.

[13] R.C. Bose and D.K. Ray-Chaudhuri, "Further Results on Error Correcting Binary Group Codes," Information and Control, Vol. 3, pp. 279 - 290; 1960.

[14] W.W. Peterson, "Encoding and Error-Correction Procedures for the Bose-Chaudhuri Codes," IRE Trans. IT, Vol. IT-6, pp. 459 - 470; 1960.

[15] J.L. Massey, Threshold Decoding, Cambridge, Mass.: MIT Press; 1963.

[16] P. Elias, "Coding for Noisy Channels," IRE Convention Record, Part 4, pp. 37 - 46; 1955.

[17] J.M. Wozencraft and B. Reiffen, Sequential Decoding, Cambridge, Mass.: MIT Press; 1961.

[18] A.D. Wyner and R.B. Ash, "Analysis of Recurrent Codes," IEEE Transactions on Information Theory (IEEE Trans. IT), Vol. IT-9, pp. 143 - 156; 1963.

[19] S.H. Reiger, "Codes for the Correction of 'Clustered' Errors," IRE Trans. IT, Vol. IT-6, pp. 16 - 21; 1960.

[20] D.W. Hagelbarger, "Recurrent Codes: Easily Mechanized, Burst-Correcting, Binary Codes," BSTJ, Vol. 38, pp. 969 - 984; 1959.

[21] J.L. Massey, "Implementation of Burst-Correcting Convolutional Codes," IEEE Trans. IT, Vol. IT-11, pp. 416 - 422; 1965.

[22] A. Kohlenberg and G.D. Forney, Jr., "Convolutional Coding for Channels with Memory," IEEE Trans. IT, Vol. IT-14, pp. 618 - 626; 1968.

[23] S.Y. Tong, "Systematic Construction of Self-Orthogonal Diffuse Codes," IEEE Trans. IT, Vol. IT-16, pp. 594 - 604; 1970.

[24] M.J. Ferguson, "'Diffuse' Threshold Decodable Rate $\frac{1}{2}$ Convolutional Codes," IEEE Trans. IT, Vol. IT-17, pp. 171 - 180; 1971.

[25] D.D. Sullivan, "A Generalization of Gallager's Adaptive Error Control Scheme," IEEE Trans. IT, Vol. IT-17, pp. 727 - 735; 1971.

[26] S.Y. Tong, "Burst-Trapping Techniques for a Compound Channel," IEEE Trans. IT, Vol. IT-15, pp. 710 - 715; 1969.

[27] H.O. Burton, D.D. Sullivan, and S.Y. Tong, "Generalized Burst-Trapping Codes," IEEE Trans. IT, Vol. IT-17, pp. 736 - 742; 1971.

[28] G.D. Forney, Jr., Concatenated Codes, Cambridge, Mass.: MIT Press; 1967.

[29] A. Papoulis, Probability, Random Variables, and Stochastic Processes, New York: McGraw-Hill; 1965.

[30] J.P. Robinson, "Error Propagation and Definite Decoding of Convolutional Codes," IEEE Trans. IT, Vol. IT-14, pp. 121 - 128; 1968.

[31] D.D. Sullivan, "Control of Error Propagation in Convolutional Codes," Ph.D. Thesis, Department of Electrical Engineering, University of Notre Dame, Notre Dame, Indiana; 1966.

[32] R.L. Townsend and E.J. Weldon, Jr., "Self-Orthogonal Quasi-Cyclic Codes," IEEE Trans. IT, Vol. IT-13, pp. 183 - 195; 1967.

[33] L.D. Rudolph, "A .ass of Majority Logic Decodable Codes," IEEE Trans. IT, Vol. IT-13, pp. 305 - 307; 1967.

[34] E.J. Weldon, Jr., "Difference-Set Cyclic Codes," BSTJ, Vol. 45, pp. 1045 - 1055; 1966.

[35] L.D. Rudolph, "Threshold Decoding of Cyclic Codes," IEEE Trans. IT, Vol. IT-15, pp. 414 - 418; 1969.

[36] W.C. Gore, "Generalized Threshold Decoding and the Reed-Solomon Codes," IEEE Trans. IT, Vol. IT-15, pp. 78 - 81; 1969.

[37] W.C. Gore, "The Equivalence of L-S [illegible] Orthogonalization and a Reed Decoding Procedure," IEEE Trans. IT, Vol. IT-15, pp. 184 - 186; 1969.

[38] J.P. Robinson and A.J. Bernstein, "A Class of Binary Recurrent Codes with Limited Error Propagation," IEEE Trans. IT, Vol. IT-13, pp. 106 - 113; 1967.

[39] J. Singer, "A Theorem in Finite Projective Geometry and Some Applications to Number Theory," Transactions of the American Math. Society, Vol. 43, pp. 377 - 385; 1938.

[40] J.L. Massey, "Shift Register Synthesis and BCH Decoding," IEEE Trans. IT, Vol. IT-15, pp. 122 - 127; 1969.

[41] J.E. Savage, "Complexity of Decoders: I - Classes of Decoding Rules," IEEE Trans. IT, Vol. IT-15, pp. 689 - 695; 1969.

[42] H.O. Burton, "A Survey of Error Correcting Techniques for Data on Telephone Facilities," Bell Telephone Laboratories, Holmdel, New Jersey; 1970.

[43] H.O. Burton and W.K. Pehlert, Jr., "On the Use of Error Statistics from Data Transmission on Telephone Facilities to Estimate Performance of Forward-Error-Correction," Bell Telephone Laboratories, Holmdel, New Jersey; 1970.

[44] M.J. Ferguson, Unpublished memorandum on rate $\frac{1}{2}$ diffuse codes.

[45] B. Tabak, Unpublished memorandum on rate $\frac{1}{2}$ diffuse codes.

[46] W. Feller, An Introduction to Probability Theory and Its Applications, Vol. 1, New York: Wiley; 1968.

[47] E.J. Weldon, Jr., "Performance of a Forward-Acting Error-Control System on the Switched Telephone Network," BSTJ, Vol. 45, pp. 759 - 761; 1966.