June 14, 1976

FINAL REPORT, FIRST YEAR

COORDINATED DESIGN OF CODING AND MODULATION SYSTEMS

(NASA Grant NSG 5025)

(NASA-CR-148163) COORDINATED DESIGN OF CODING AND MODULATION SYSTEMS FINAL REPORT, 1 SEP. 1974 - 30 NOV. 1975 (NOTRE DAME UNIV.) 114 P HC CSCL 17B

N76-25314

UNCLAS

G3/17 42213

Department of

ELECTRICAL ENGINEERING

UNIVERSITY OF NOTRE DAME, NOTRE DAME, INDIANA

REPRODUCED BY
NATIONAL TECHNICAL
INFORMATION SERVICE
U.S. DEPARTMENT OF COMMERCE
SPRINGFIELD, VA. 22161

June 14, 1976

FINAL REPORT, FIRST YEAR

COORDINATED DESIGN OF CODING AND MODULATION SYSTEMS

(NASA Grant NSG 5025)

Submitted to: NASA Goddard Space Flight Center
Greenbelt, Md. 20771
ATT: Dr. Thomas J. Lynch (Code 930)
Mr. William D. Poland (Code 730:4)
Mr. John Rende (Code 814.1)

and

NASA Scientific and Technical Information Facility
P.O. Box 8757
Baltimore/Washington International Airport
Baltimore, Md. 21240

Principal Investigator: Dr. James L. Massey
Freimann Professor of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

Research Assistants: T. Ancheta
R. Johannesson
G. Lauer
L. Lee

RESEARCH PERIOD REPORTED: September 1, 1974 to November 30, 1975.
(Includes no-cost extension from
September 1, 1975 to November 30, 1975)

REPRODUCED BY
NATIONAL TECHNICAL
INFORMATION SERVICE
U. S. DEPARTMENT OF COMMERCE
SPRINGFIELD, VA. 22161

I. INTRODUCTION

The general aim of the research under this grant has been the joint optimization of the coding and modulation systems employed in telemetry systems. Particular emphasis has been placed on that research which would assist in the formulation of the inner and outer coding standards by the Goddard Space Flight Center.

Section II provides a brief description of the major research achievements during the first year of this grant, with reference to the technical reports or other publications where a detailed description may be found. Section III lists the personnel who have been involved in this research. An appendix gives a detailed summary of the research done in concatenated coding systems.

II. SUMMARY OF RESEARCH PERFORMED

A. Convolutional Code Construction

Because of the desirability of standardizing as soon as possible upon a few convolutional codes, either long-constraint-length codes for use with sequential decoding or short-constraint-length codes for use with Viterbi decoding, it has become imperative to ensure that presently-known codes are optimum, or sufficiently close to optimum, so that a marked improvement would not later be forthcoming.

Besides the traditional parameter of free distance, d_∞, as a measure of code goodness, research [1] under the predecessor of this grant has shown the importance of the distance profile, $\underline{d}$, particularly for long-constraint-length codes used with sequential decoding. Moreover, as pointed out in our semi-annual status report [2], there are situations in which long constraint length systematic codes might be preferable over the non-systematic codes now routinely employed. Thus, it became important to find optimum long-

constraint-length systematic codes. This task was carried to completion by R. Johannesson. In his report on this research [3], convolutional codes with an optimum distance profile are given at rate 1/2 for all constraint lengths $K \leq 61$. In the same report, Johannesson also lists quick-look-in rate 1/2 codes ($G_2(D) = D + G_1(D)$) with an optimum distance profile for $K \leq 51$. This report also describes simulations which confirmed the principle investigator's conjecture [2] that systematic codes perform equally as well as non-systematic codes under sequential decoding when the dummy information zeroes are suppressed in the tail of the systematic code.

With the ever-increasing demand for greater reliability in decoded data, there has been a resurgence of interest in concatenated coding. A major effort under this grant has been made to find convolutional codes that are optimum, or nearly so, for use with Viterbi decoding in the inner coding portion of a concatenated coding system. The concern for the interaction of the various components in the total coding/modulation system led to the discovery by L. Lee [4] of a new type of convolutional code, the <u>unit-memory</u> code, which is ideal for such inner system usage because of its "byte-oriented" structure as opposed to the "bit-oriented" structure of conventional convolutional codes. Lee found optimum unit-memory codes for all rates and constraint lengths of practical interest. We consider the discovery and development of this new type of byte-oriented convolutional code to be one of the principal achievements of this research and one certain to give increased impetus to the use of convolutional codes in concatenated coding systems.

In conjunction with the search for good long convolutional codes, simulations of sequential decoding on the deep-space channel have been carried out to compare directly various convolutional codes that have been proposed as candidates for use in various deep-space systems. Detailed comparisons

of rate 1/2, K = 32 codes have been conducted and reported by the principal investigator [5]. These simulations support the choice of the so-called "Massey-Costello" K = 32 quick-look-in code which is made in the proposed Goddard coding standard.

B. Soft-Decision Demodulation

In earlier work [6], the principal investigator demonstrated the value of the "cut-off rate" R_o of the discrete channel created by the modulation system as a measure of the quality of the modulation system for use with coding, and he showed how to design an optimum soft-decision demodulator for this criterion when binary signalling is used. An important advance made under this grant by Lee is the extension of all these results to non-binary signalling [7]. Lee's work shows that the optimum soft-decision regions in likelihood space are always bounded by hyperplanes. Lee gave an algorithm for the determination of these optimal regions, as well as some heuristic rules for finding good, but sub-optimum, decision regions directly in signal space. The combination of the results in [6] and [7] provide a sound basis for the design of modulation systems to be used in conjunction with coding.

C. Syndrome Source Coding

Some of the research under this grant has been concerned with the use of error-correcting codes to perform source coding or "data compression." Continuing his earlier work on syndrome source coding [8], Ancheta has during the past year made an important innovation which he calls "noiseless universal syndrome source coding" (NUSSC) and has demonstrated its robustness in compressing a broad range of sources [9]. Like its parent, NUSSC employs a very simple source encoder and hence appears very attractive for use on board spacecraft; the more complex source decoder being at the ground site. Ancheta's innovation consists of using several different parity-check matrices and adaptively choosing the one to use in forming the syndrome according to

A

the particular source output sequence. By cleverly choosing the parity-check matrices to be those of a nested sequence of error-correcting codes, Ancheta's NUSSC scheme is almost as simple to implement as plain syndrome source coding. NUSSC appears to be a very practical method of data compression, and it is currently being applied to real telemetry data supplied by the Goddard Space Flight Center to confirm its effectiveness.

D. Concatenated Coding Systems

During this year of research, L. Lee has completed a major study of concatenated coding systems which employ convolutional codes in the inner coding subsystem. This work forms the subject of Lee's doctoral dissertation [10] which is included as the appendix to this report. Lee describes an almost bewildering array of options that are available to the designer of a concatenated coding system and performs the valuable service of specifying the precise gain (in db) which each such option affords. The most sophisticated systems considered by Lee outperform all previous concatenated coding systems and represent nearly the ultimate in performance. It is expected that Lee's work will be the standard in this field for many years to come.

III. PERSONNEL

The table below lists all personnel who have been involved in the research under the first year of this grant.

We are pleased to report that Mr. Lin-nan Lee completed the requirements for the Ph.D. degree in electrical engineering under this grant in November 1975 and is now a member of the research staff of the Linkabit Corporation in San Diego, California.

Name	Category	Dates of Affiliation	Source of Support if not this Grant
J. L. Massey	Principal Investigator	9/1/74 - 11/30/75	
T. C. Ancheta, Jr.	Research Assistant	9/1/74 - 11/30/75	
R. Johannesson	Research Assistant	9/1/74 - 11/3/74	Swedish Government Grant
G. S. Lauer	Research Assistant	6/1/75 - 7/31/75	
L. Lee	Research Assistant	9/1/74 - 7/31/75	

REFERENCES

[1] R. Johannesson, "Robustly-optimal, rate one-half, binary convolutional codes," IEEE Trans. Inform. Theory, Vol. IT-21, pp. 464-468, July 1975.

[2] J. L. Massey, Semi-Annual Status Report NASA Grant NSG 5025, "Coordinated Design of Coding and Modulation Systems," Dept. of Elec. Engr., Univ. of Notre Dame, Notre Dame, Indiana, June 25, 1975.

[3] R. Johannesson, "Some long, rate one-half, binary convolutional codes with an optimum distance profile and the systematic versus nonsystematic code question," Tech. Rpt. No. EE-756, Dept. of Elec. Engr., Univ. of Notre Dame, Notre Dame, Indiana, August 19, 1975.

[4] L. Lee, "Short, unit-memory, byte-oriented, binary convolutional codes with maximal free distance," Tech. Rpt. No. EE754, Dept. of Elec. Engr., Univ. of Notre Dame, Notre Dame, Indiana, July 11, 1975.

[5] J. L. Massey, "A recommended R = 1/2, K = 32, quick-look-in convolutional code for NASA use," Tech. Rpt. No. EE-751, Dept. of Elec. Engr., Univ. of Notre Dame, Notre Dame, Indiana, April 28, 1975.

[6] J. L. Massey, "Coding and modulation in digital communications," in Proc. Int. Zurich Seminar on Digital Comm., Zurich, Switzerland, pp. E2(1)-E2(4), March 12-15, 1975.

[7] L. Lee, "On optimal soft-decision demodulation," Tech. Rpt. No. EE-753, Dept. of Elec. Engr., Univ. of Notre Dame, Notre Dame, Indiana, July 11, 1975.

[8] T. C. Ancheta, Jr., "Syndrome source coding for data compression," Tech. Rpt. No. EE-7313, Dept. of Elec. Engr., Univ. of Notre Dame, Notre Dame, Indiana, July 25, 1973 (Revised July 1974).

[9] T. C. Ancheta, Jr., "Syndrome-source coding and its universal generalization," Tech. Rpt. No. EE-755, Dept. of Elec. Engr., Univ. of Notre Dame, Notre Dame, Indiana, July 17, 1975.

[10] L. Lee, "Concatenated coding systems employing unit-memory convolutional codes and symbol-oriented optimal decoding algorithms," Ph.D. Thesis, Dept. of Elec. Engr., Univ. of Notre Dame, Notre Dame, Indiana, May 1976. [Attached as the APPENDIX to this report.]

APPENDIX

CONCATENATED CODING SYSTEMS EMPLOYING UNIT-MEMORY CONVOLUTIONAL CODES AND SYMBOL-ORIENTED OPTIMAL DECODING ALGORITHMS

A Dissertation

Submitted to the Graduate School of the

University of Notre Dame in Partial Fulfillment

of the Requirements for the Degree of

Doctor of Philosophy in Electrical Engineering

by

Lin-nan Lee, B.S.E.E., M.S.E.E.

James L. Massey
Director

Department of Electrical Engineering

Notre Dame, Indiana

May, 1976

ABSTRACT

of

"CONCATENATED CODING SYSTMES EMPLOYING UNIT-MEMORY CONVOLUTIONAL CODES AND SYMBOL-ORIENTED OPTIMAL DECODING ALGORITHMS"

by

Lin-nan Lee

To achieve very reliable communications in a very noisy channel with relatively small coding complexity, concatenated coding systems utilizing convolutional codes as the inner code and the Reed-Solomon (RS) codes as the outer code have been proposed by previous investigators. However, there has always been a "matching" problem between the bit-oriented convolutional inner codes and the byte-oriented RS outer codes. To use efficiently the potential of concatenation, we propose, in this dissertation, to concatenate a byte-oriented unit-memory convolutional code which has greater free distance than previously known convolutional codes of the same rate and the same state-complexity with RS-outer codes of the same symbol size. We also propose to utilize a Real-Time Minimal-Byte-Error Probability (RTMBEP) decoding algorithm in conjunction with the feedback from the outer decoder as the decoder for the inner convolutional code. The performance of this concatenated coding system is studied, and the improvement due to each additional feature is calculted. It is shown by simulation that this concatenated coding system out-performs all previously known concatenated coding schemes.

ACKNOWLEDGEMENT

I owe a very great debt of gratitude to Professor James L. Massey for his patient and generous guidence of this research as well as his numerous suggestions and discussions which have eventually resulted in the completion of this dissertation. He has also provided me with the necessary background in the area of information theory and coding theory through his skillful teaching in his courses during my graduate study. I would also like to thank Professors Micheal K. Sain, John J. Uhran, Jr., David L. Cohn, and R. Jeffery Leake who have not only agreed to serve as my readers, but have also provided me with useful knowledge in the areas of system theory and communication theory. I would also like to thank Miss Anh, Nguyen and Mrs. Rosemary Reiter who have carefully typed either the final copy of this dissertation or the technical reports included in this dissertation.

I would also like to express my appreciation for the research assistantships provided to me under NASA Grants NGL 15-004-026 and NSG 5025 during the past three years, and the assistance provided by the LINKABIT Corporation in preparing the final copy of this dissertation.

Finally, I wish to thank my dear parents, who, even in the most difficult situation, have furnished me the indispensible encouragement and moral support.

TABLE OF CONTENTS

	Page
ACKNOWLEDGEMENT	ii
CHAPTER I. INTRODUCTION	1
CHAPTER II. BYTE-ORIENTED CONVOLUTIONAL CODES	10
2.1. Binary Convolutional Codes	10
2.2. Unit-memory Codes with Maximal d_{free}	12
2.3. Byte-oriented Nature of Unit-memory Codes	17
Chapter III. BYTE-ORIENTED DECODING ALGORITHMS FOR CONVOLUTIONAL CODES	20
3.1. Derivation of A Real-Time Minimal-Bit-Error Probability Decoding Procedure for $(n_o,1)$ Convolutional Codes	23
3.2. Implementation of the Decoding Algorithm	27
3.3. Real-Time Minimal-Byte-Error Probability Decoding Algorithm for Unit-memory Convolutional Codes	33
3.4. Real-Time Viterbi Decoding	40
3.5. Simulation Results for the Algorithms	44
3.6. Remarks	48
CHAPTER IV. OPTIMAL SOFT-DECISION DEMODULATION	49
4.1. The Symmetric Cut-off Rate	54
4.2. A Necessary Condition for Optimal Demodulation	54
4.3. Decision Regions in Likelihood Space	57
4.4. Examples and an Iterative Optimization Technique	61
4.5. A Counterexample to the Sufficiency of the Optimality Condition	77
4.6. Summary	80
CHAPTER V. PERFORMANCE OF CONCATENATED CODING SYSTEMS ON A SIMULATED AWGN CHANNEL	81
5.1. Odenwalder's Concatenated Coding System and Soft-Decision Modification with the RTMBEP Decoding Algorithm	83
5.2. Feedback from the Outer Decoder to the Inner Decoder	91
5.3. Zeoli's Concatenated Coding System and Zeoli's Modification Applied	

Page

to Unit-memory Convolutional Codes 95

5.4. Degradation of Performance for Employing Higher Rate Inner Codes . . 100

5.5. Confidence Intervals for the Simulation Results 104

CHAPTER VI. SUMARY AND CONCLUSIONS 106

BIBLIOGRAPHY 111

CHAPTER I

INTRODUCTION

Shannon's celebrated theory of information states that information can be transmitted with an arbitrarily small error probability provided that the rate of transmission is below channel capacity. Communications engineers now recognize that this promise of highly reliable communications can be achieve only by means of coding.

Because the complexity of coded communication systems grows exponentially with the block length for block codes (or with the constraint length for convolutional codes), instead of directly using very long codes, the idea of cascading two or more codes of less complexity to achieve higly reliable communications was first considered by Elias [1], and later by Forney [2]. Forney's technique of using two or more block codes over different alphabets to obtain very low error rate over noisy channels is known as concatenated coding. Guided by the premise that a convolutional code generally performs better than a block code of the same complexity, Falconer [3], and later Jelinek and Cocke [4] tried to cascade block codes and convolutional codes. Figure 1.1 shows a general representation of a block-convolutional concatenated coding system.

In their schemes, sequential decoding is used for the inner decoder. However, the performance of sequential decoding is such that the probability of error can be reduced very sharply with slight increment of signal energy if the rate of transmission is

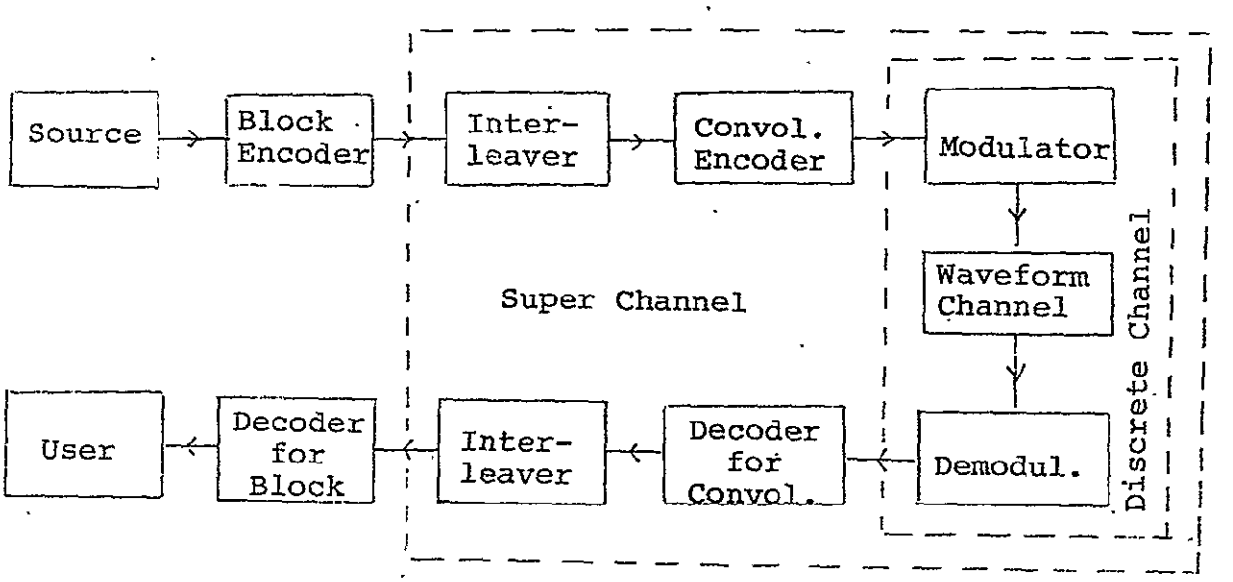

Figure 1.1: A general representation of the block-convolutional concatenated coding systems.

below the computational cut-off rate, R_{comp}. The primary function of the outer block code is only to resolve decisions when the sequential decoder experiences computational overflow. Therefore, the over-all coding system can be regarded, more or less, as a sequential decoded convolutional coding system.

The implementation and operation of Viterbi decoding are later found simpler than sequential decoding in a number of respects. In particular, with a moderate constraint length convolutional code, Viterbi decoders can be operated with an error rate less than 10^{-2} at a rate slightly higher than R_{comp}. From Forney's result [2], it is conceivable that a concatenated coding system with a powerful outer code can perform reasonably well when its inner decoder is operated with a probability of error in the range between 10^{-2} and 10^{-3}. Odenwalder [5], then, chose a Viterbi decoder for the inner coding system. We recall that it is always possible to achieve very low error probability by means of sequential decoding of long constraint length convolutional codes at rates slightly below the "cut-off rate", $R_o(R_{comp})$, of the channel if one can tolderate a certain amount of erasures. Therefore, in order to be an attractive alternative to sequential decoding, a concatenated coding system has to operate at an overall rate slightly above the cut-off rate of the channel. On the other hand, we can always regard the convolutional encoder-discrete channel-decoder combination in Figure 1.1 as a "super-channel" for the outer coding system; the function of the inner convolutional coding system is then to create a better channel for the outer coding system.

Hence it is necessary that the inner convolutional coding system be operated at a rate slightly below the channel capacity so that the super-channel is at least slightly "cleaner" than the original discrete channel. This requirement has thus limited the choices of outer codes to those of very high rate. And because it is usually difficult to implement Viterbi decoders for high rate convolutional codes with large enough free distance, high rate block codes are the only option left for the outer coding system.

Generally, we can reduce the total signal energy per information bit required to achieve a certain error probability for a given channel by employing a lower-rate code and thus spending less energy in each baud. For example, simulation result in section 5.4 shows a 0.5 dB improvement in the signal-energy-per-information-bit to one-sided noise-power-spectral-density ratio when one elects to use a rate 1/3 convolutional coding system of the same decoder complexity in place of a rate 1/2 convolutional coding system. But for a given data rate, the lower rate coding system implies a higher bandwidth, which in turn implies the need for a much more sophisticated signal detection device. The difficulty in operating the phase locked loop in the detection device increases with the bandwidth and negates the advantage of using very low rate codes. Experience has shown that convolutional codes of rate 1/2 and 1/3 are the most attactive candidates for the inner coding system. Furthermore, because the complexity of Viterbi-like decoders for convolutional codes is exponentially dependent on the constraint length of the convolutional code, the task of finding "good" inner codes for the concatenated coding system is then focused to the search for rate

1/2 and rate 1/3 convolutional codes of short constraint length which are capable of good performance on a very noise channel.

Since the output error patterns of Viterbi-type decoders for convolutional codes are usually bursty, block codes over a large alphabet, such that many bits of the inner code form one symbol of the outer code, appear very attractive for the outer coding system. In particular, it appears that the Reed-Solomon (RS) block codes are the most satisfactory because there are relatively simple decoding procedures (such as the Berlekamp-Massey [6], [7], algorithm) for RS codes and because of the "maximum-distance-separable" property of the RS codes. But the lengths of the bursts of errors made by Viterbi-like decoders are widely distributed so that it is generally necessary to interleave the decoder output for the inner convolutional code such that errors in the individual RS-symbols of one block are independent. Otherwise, we would have to use a very long block code to operate the system efficiently.

From the above general discussion, we conclude that Odenwalder's configuration of a block-convolutional concatenated coding system is generally a sound choise. However, we have observed that we may further improve the performance of this type of concatenated coding system in several different directions. These possible directions of improvement will be studied in the following chapters of this dissertation.

Because the most-likely burst length of the decoding error patterens made by the inner decoder are on the order of the constraint length of the inner convolutional code, Odenwalder chose

the RS symbol to be of the same size as the constraint length of the convolutional code. Although this is certainly a reasonable choice, it can be improved upon. For example, it is very unlikely that the beginning of a burst is always aligned with the boundary between two RS symbols; therefore, it is possible that a burst only two bits long may affect two RS symbols. This fact leads us to the idea of constructing good convolutional codes which are symbol-oriented rather than bit-oriented. In Chapter II, we discuss a systematic approach to constructing such code, and we shall see that the codes thus constructed generally have free distance better than Odenwalder's convolutional codes of roughly the same complexity in terms of the Viterbi decoder implementation. In fact, because of this improved free distance and the symbol oriented nature of these codes, we obtain an approximately 0.3 dB improvement in the over-all performance of the concatenated coding system when these codes replaces Odenwalder's codes.

Another possible improvement is to modify the decoder for the convolutional code such that the decoder emits not only the most-likely estimated symbol, but also reliability information about the estimated symbol. Based on this reliability information, the outer decoder is then able to perform either "erasures-and-errors" decoding or "generalized-minimum-distance" (GMD) decoding as suggested by Forney [2]. Zeoli [8] and Jelinek [9] have proposed methods of extracting reliability information from a Viterbi decoder. Conceptually, their approach is to annex a long tail to the original

convolutional code and to use this added tail to provde an error detection capability for the estimate made by the Viterbi decoder for the original shorter convolutional code. This approach requires feedback from previously decoded symbols in the Viterbi decoder and, more importantly, uses the symbol as corrected by the outer decoder to restart the inner Viterbi decoder whenever an error is corrected by the outer decoder. We find that this feedback from the outer decoder improves the performance by 0.3 dB, while the error detecting capability and the "erasures-and-errors" decoder provide an additional improvement of 0.2 dB.

We have also studied algorithms which compute the a posteriori probability of each decoded symbol for the short constraint length convolutional code and which use this a posteriori probability as the reliability information provided to the outer coding system. Although this technique proved to be less powerful than Zeoli's tail annnexation scheme (this erasure scheme improves the performance by only 0.05 dB to 0.1 dB over hard-decision decoding), its performance is undoubtly optimal among all the possible schemes employing only a short constraint length convolutional code without an annexed tail, because decoding decisions are based on the a posteriori probability calculated. Moreover, in conjection with the use of the feedback, the a posteriori probability decoding algorithms seem to perform much better than the Viterbi decoder does when aided by feedback from the outer decoder (approximately 0.2 dB

difference). The algorithm used together with feedback from the outer decoder, even without the extra tail, offers a slight improvement over Zeoli's scheme. Moreover, in this scheme, the inner encoder and the inner decoder have the same constraint length and the decoder can return to normal operation a few branches after an error event occurs. It is possible to interleave the output of the decoder for the convolutional code to create a memoryless superchannel for the RS decoder. This results a simpler implementation than that for Zeoli's type of system. These algorithms, which we call the Real-Time Minimal-Byte-Error (RTMBEP) decoding algorithm, and the Real-Time Minimal-Bit-Error Probability (RTMbEP) decoding algorithm are described in detail in Chapter III.

Another area of possible improvement which one can visualize for Odenwalder's system is the area of soft-decision demodulation. Using the criterion proposed by Wozencraft and Kennedy [10] and by Massey [11] we can "optimize" the demodulator by maximizing the cut-off rate, R_o, of the resultant discrete channel. In chapter IV, we shall show that the decision boundary of the demodulator maximizing R_o are hyperplanes in likelihood space. Although we are not able to improve the performance of coded communication system very much in the case of binary signaling with this optimal demodulator because Viterbi decoding is relatively insensitive to the demodulator design, we believe that the decision rules for optimal demodulators thus obtained may be useful for future soft-decision decoding, because, the symbol-oriented decoding algorithms described in Chapter III compute the reliability in-

formation not only for the most-likely symbol but also for all symbols in the coding alphabet and, therefore, make available the likelihood-ratio vector. Although this aspect of the study has not been carried very far and deviates somewhat from the main line of our research, we summarize in Chapter IV some scattered results which are related to the subject of this dissertation.

In Chapter V, we give the performance of various types of concatenated coding systems as obtained from simulations and compare the improvement in performance due to each feature employed in the system. Finally, the results and conclusions are summarized in Chapter VI.

CHAPTER II

BYTE-ORIENTED CONVOLUTIONAL CODE

In this chapter, we introduce "unit-memory" convolutional codes which are "byte-oriented" in such a way as to be attractive for use in concatenated systems. We shall show that (n_o, k_o) convolutional codes with unit memory always achieve the largest free distance among all codes of the same rate k_o/n_o and the same number 2^{Mk_o} of encoder states, where M is the encoder memory.

2.1 BINARY CONVOLUTIONAL CODES

Let the binary k_o -tuple $\underline{a}_t$ denote the subblock of information digits at time t (t = 0, 1, 2, ...), and let the binary n_o - tuple $\underline{b}_t$ denote the encoded subblock at time t in an (n_o, k_o) convolutional code. Then, the encoding equations may be written

$$\underline{b}_t = \underline{a}_t G_0 + \underline{a}_{t-1} G_1 + \ldots + \underline{a}_{t-M} G_M \quad (t = 0,1,2,\ldots) \quad (2.1)$$

where each G_i is a $k_o \times n_o$ binary matrix, where M is the code memory, where the operations are in GF(2), and where, by way of convention, $\underline{a}_t = \underline{0}$ for t>0. An encoding circuit is shown in Figure 2.1. Note that the encoder has 2^{MK_o} distinct states where the state is taken as the contents of the delay cells in the encoder. We shall refer to the number Mk_o of binary state variables in the encoder as the <u>state-complexity</u> of the convolutional code.

The constraint length K (measured in information digits) of the convolutional code is defined by

$$K = (M + 1) \quad k_o.$$

---

* This chapter of the dissertation is taken from [28].

The rate of the code is defined by

$$R = k_o/n_o.$$

In virtually all past applications of convolutional codes, k_o and n_o have been taken as relatively prime, i.e., gcd $(k_o, n_o) = 1$ where "gcd" denoted "greatest common divisor". In fact, the condition gcd $(k_o, n_o) = 1$ is generally tacitly assumed so that speaking, for instance, of a convolutional code as being of rate $R = 1/2$ would imply $k_o = 1$ and $n_o = 2$ unless the contrary were explicitly states. As will be seen, however, there can be advantages in taking gcd $(k_o, n_o) > 1$.

For convenience, let $\underline{b}_{[t,t']}$ denote the encoded sequence $[\underline{b}_t : \underline{b}_{t+1} : \ldots : \underline{b}_{t'}]$ over time units t through t' and let $\underline{b}_{[0,\infty)}$ denote the entire semi-infinite encoded sequence. Let $\underline{a}_{[t,t']}$ and $\underline{a}_{[0,\infty)}$ be similarly defined. The <u>free distance</u>, d_{free}, of the convolutional code is the minimum Hamming distance between all pairs of encoded sequences $\underline{b}_{[0,\infty)}$ resulting from pairs of information sequences $\underline{a}_{[0,\infty)}$ that differ in their time 0 subblock. By

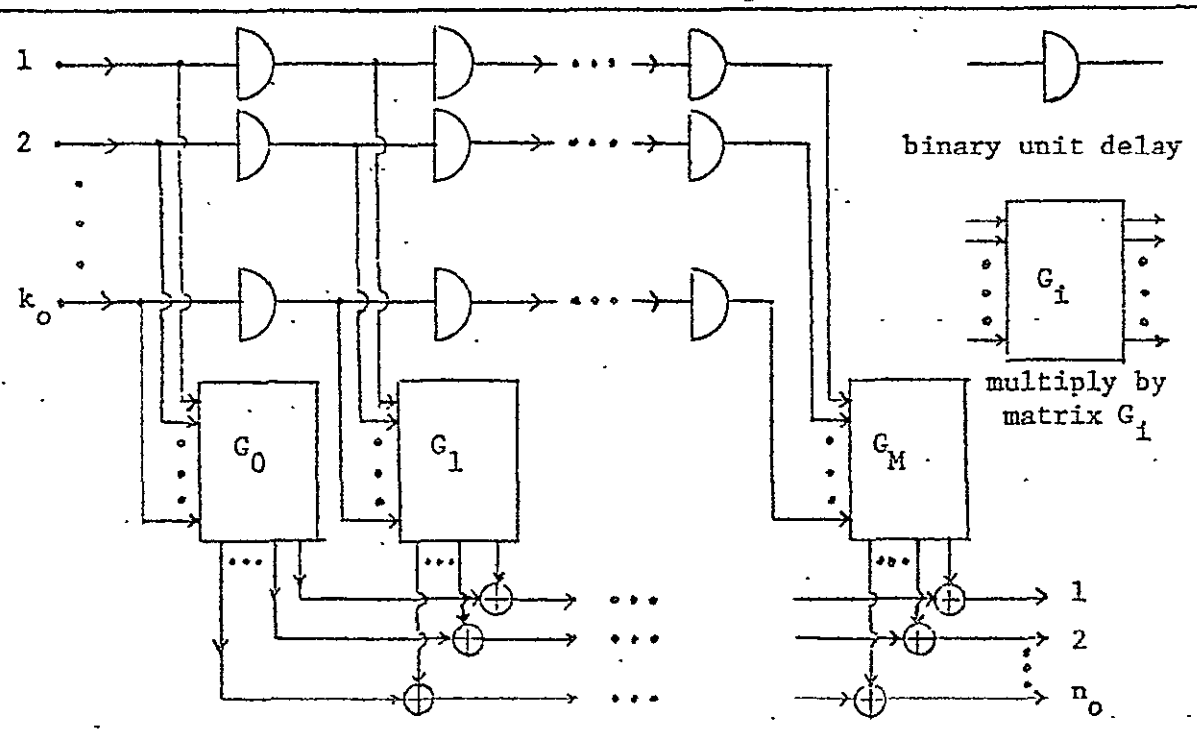

Figure 2.1. An encoding circuit for an (n_o, k_o) convolutional code with memory M.

the linearity of (2.1), it follows that

$$d_{free} = \min_{\underline{a} \neq 0} W(\underline{b}_{[0,\infty)}) \tag{2.2}$$

where $W(\cdot)$ denotes the Hamming weight of the enclosed binary sequence and where the minimization is over all $\underline{a}_{[0,\infty)}$ such that $\underline{a}_0 \neq \underline{0}$. The code is <u>non-catastrophic</u> [12, 13] when $W(\underline{a}_{[0,\infty)}) = \infty$ implies that $W(\underline{b}_{[0,\infty)}) = \infty$, in which case the minimization in (2.2) reduces to the minimum over all $\underline{a}_{[0,\infty)}$ such that $\underline{a}_0 \neq 0$ and $W(\underline{a}_{[0,\infty)}) = \infty$. The restriction to non-catastrophic codes entails no loss in the achievable value of d_{free} for given n_o, k_o and M, a fortunate situation because the non-catastrophic property is essential in applications (cf. [12]).

Because d_{free} is the primary determiner of decoding error probability when Viterbi (i.e., maximum likelihood) decoding is used with a non-catastrophic code, d_{free} is the usual criterion of goodness for codes to be used with Viterbi decoders. Because the number of states of the Viterbi decoder [14] coincides with the number of encoder states, viz. 2^{Mk_o}, practically dictates a small state-complexity. The region $Mk_o \leq 6$ appears to be about the range where Viterbi decoding is attractive in applications. Thus, for Viterbi decoding applications, we are motivated to find, for a given code rate and a given state-complexity in the above range, a convolutional code with maximum d_{free}. In the next section, we report the results of our search for such codes and we also derive a useful upper bound on the attainable d_{free}.

2.2. UNIT-MEMORY CODES WITH MAXIMAL d_{free}

Any (n_o, k_o) convolutional code with memory M can be considered as an $(n_o' = Mn_o,\ k_o' = Mk_o)$ code with $M'=1$ simply by taking

$$G_0' = \begin{bmatrix} G_0 & G_1 & \cdots & G_{M-1} \\ 0 & G_0 & \cdots & G_{M-2} \\ \vdots & \vdots & & \vdots \\ 0 & 0 & & G_0 \end{bmatrix}$$

and

$$G_1' = \begin{bmatrix} G_M & 0 & \cdots & 0 \\ G_{M-1} & G_M & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ G_1 & G_2 & & G_M \end{bmatrix} .$$

These two codes are entirely equivalent in the sense that the same semi-infinite binary information sequence produces the same semi-infinite binary encoded sequence, although the division into time subblock would be different. Since the state-complexity is Mk_o for both codes, it follows that <u>for a given state-complexity and given rate, the maximum value of d_{free} is achieved within the subset of convolutional codes with</u> $M = 1$. Hence, we can restrict our search for optimal codes, for a given rate and state-complexity, to codes with <u>unit-memory</u>.

For a unit-memory code, equation (2.1) reduces to

$$\underline{b}_t = \underline{a}_t G_0 + \underline{a}_{t-1} G_1 \qquad (t = 0,1,2,\ldots). \qquad (2.3)$$

When $\underline{a}_0 \neq \underline{0}$ is the only non-zero information subblock, then

$$\underline{b}_{[0,1)} = \underline{a}_0 \; [G_0 : G_1] \qquad (2.4)$$

is the only possibly-non-zero portion of $\underline{b}_{[0,\infty)}$. From (2.2), it then follows that <u>the attainable</u> d_{free} <u>of a unit-memory</u> (n_o, k_o) <u>convolutional code is upper-bounded by the largest minimum distance of an</u> $(n = 2n_o,\ k = k_o)$ <u>block code</u>. We shall call this upper bound the block code upper bound on d_{free}, and we note that McEliece and Rumsey [15] have used similar arguments to derive more elaborate upper bounds on d_{free} for codes where $M \neq 1$.

We see that the argument used above to establish the block code upper bound on d_{free} suggests the following search procedure for finding a non-catastrophic unit-memory (n_o, k_o) code with maximal d_{free}:

(i) Set d equal to the largest d_{min} achievable by any $(n=2n_o, k = k_o)$ block code.

(ii) Choose $[G_0:G_1]$ as the generator matrix of a $(n=2n_o, k = k_o)$ block code with $d_{min} = d$. If $d_{free} = d$ and the code is non-catastrophic, stop. Otherwise continue with step (ii) until all block codes with $d_{min} = d$ have been exhausted.

(iii) Reduce d by 1 and return to step (ii).

The above search procedure was carried out to obtain, for rates 1/4, 1/3, 1/2 and 2/3 (which are the usual rates of interest in applications), a non-catastrophic unit-memory convolutional code with maximal d_{free} for all state-complexities of 6 or less. The values of d_{free} obtained are given in Table I where we also list, for comparison, the largest d_{free} attainable by a code of the same state-complexity having gcd $(n_o, k_o) = 1$. The block code upper bound on d_{free} for each case is also listed.

The codes with gcd $(n_o, k_o) = 1$ that achieve the values of d_{free} given in Table 2.1 may be found in Larsen [16] and Paaske [17] The values of the block code upper bound on d_{free}, given in Table I, were taken from Calabi and Myrvaagnes [18] and from Helgert and Stinaff [19]. In Table 2.2 we give the

matrices G_0 and G_1 of a non-catastrophic unit-memory code with maximal d_{free} at each place where Table 2.1 shows that value of exceed the d_{free} available from the best gcd $(n_o, k_o)=1$ code having the same state-complexity.

Rate (k_o/n_o)	State Complexity (No. of State Variables)	Block Code Upper Bound on d_{free}	Maximal d_{free} of Unit-Memory Codes	Maximal d_{free} of gcd $(n_o, k_o) = 1$ Codes
1/4	1	8	7	7
	2	10	10	10
	3	13	13	13
	4	16	16	16
	5	20	20	18
	6	24	24	20
1/3	1	6	5	5
	2	8	8	8
	3	10	10	10
	4	12	12	12
	5	15	15	13
	6	16	16	15
1/2	1	4	3	3
	2	5	5	5
	3	6	6	6
	4	8	8	7
	5	9	9	8
	6	10	10	10
2 2/3	2	4	3	3
	4	6	6	5
	6	8	7	7

Table 2.1: Maximal d_{free} for a given state-complexity Mk_o of unit memory convolutional codes and of convolutional codes with gcd $(n_o, k_o) = 1$.

Rate	n_o	k_o	G_0	G_1	d_{free}
1/2	8	4	10000111 01001011 00101101 00011110	10001011 11100010 10111000 11010001	8
1/2	10	5	1000011111 0100001111 0010011110 0001011001 0000110101	1111111111 1111000000 0010110100 1010011010 0110101001	9
1/3	15	5	100001111011010 010000111101101 001001011110110 000101101101011 000011110110101	000111110010100 001101100101010 011001001100101 110000011110010 100010111001001	15
1/3	18	6	111000110100110000 011100011010011000 001110001101001100 000111100110000110 100011010011000011 110001101001100001	000011000111001011 000110001110010110 001100011100101100 011000111000011001 110000110001110010 100001100011100101	16
1/4	20	5	10000011111110011000 01000101110111001100 00100011011001100110 00010111011001100011 00001111101100110001	00011110100001100111 00110011010110001110 01100101101000111100 11000010110011011001 10001101011100010011	20
1/4	24	6	100000111110111010110000 010000011111011101011000 001000101111101110001100 000100110111010111000110 000010111011101011000011 000001111101110101100001	001111000101001011010011 011110001010010110100110 111100010100101100001101 111001101000011001011010 110011010001110010110100 100111100010100101101001	24
2/3	6	4	100001 010010 001011 000111	111100 110110 010011 101001	6

Table 2.2. The encoding matrices of some non-catastrophic unit-memory convolutional codes with free distance greater than the maximal free distance of gcd $(n_o, k_o) = 1$ codes of the same rate and state complexity.

2.3. BYTE-ORIENTED NATURE OF UNIT-MEMORY CODES

We now show that short, unit-memory convolutional codes are "byte-oriented" in such a way as to be attractive for use, with Viterbi decoding, as the inner coding component of a concatenated coding system.

In general, the state at time t of the convolutional encoder is the information sequence $\underline{a}_{[t-M,\ t-1]} = [\underline{a}_{t-M}:\underline{a}_{t-M+1}: \ldots :\underline{a}_{t-1}]$ over the preceding M time units. Note that the successor of this state, namely, $[\underline{a}_{t-M+1}: \ldots :\underline{a}_{t-1}:\underline{a}_t]$, is already determined up to the 2^{k_o} choices of $\underline{a}_t$, i.e., each state will have 2^{k_o} successors in the "trellis" defined by the convolutional code [12]. In the corresponding Viterbi decoder, the "metric" for the best path to each of the 2^{Mk_o} possible states at time t must be relayed to each of its 2^{k_o} successors. Hence, the value of k_o influences the overall complexity of the Viterbi decoder, although much less strongly then does the state-complexity. Nonetheless, to determine, for instance, whether the Viterbi decoder for an $R = 1/3$, $M=1$, $k_o = 6$ code (state-complexity 6) is simpler than the Viterbi decoder for an $R = 1/3$, $M = 7$, $k_o = 1$ code (state-complexity 7) would require a detailed analysis of the specific decoder design.

With Viterbi decoding of convolutional codes, there is a natural segmentation of the decoded information digits into bytes of k_o bits, because the information byte $\underline{a}_t$ acts as a unit in determining the correct state. The typical decoding "error events " [13] give rise to the incorrect decoding of a small

number of bytes, viz., the average number of non-zero information bytes in an information sequence $\underline{a}_{[0,\infty)}$ that generates an encoded $\underline{b}_{[0,\infty)}$ of Hamming weight d_{free}. In an M = 1 code, this average is near 1, although there will generally be about $k_o/2$ bit errors per byte. In a code with $k_o = 1$, there will generally be a small (say, about 3) of byte errors but, since a byte is a bit in this case, the same number of bit errors. However, if the information bits with the latter code are gathered into "bytes" of $Mk_o = M$ bits, there will generally be about $1\frac{1}{2}$ "byte" errors per error event since the bit decoding errors are not synchronized to begin at the start of these "bytes." Thus, even if both codes have the same state-complexity and same d_{free}, one would expect the unit-memory code to have a lower byte-error probability for bytes of Mk_o bits.

To test the validity of these observations, we simulated Viterbi decoding on an additive white Gaussian noise (AWGN) channel with several values of the energy per information bit to one-sided noise power spectral density ratio, E_b/N_o, for several convolutional codes of rate R = 1/3. The codes tested were (i) the $k_o = 6$ unit-memory code of Table 2.2, (ii) the $k_o = 1$, M = 6 code given by Larsen [16] and (iii) the $k_o = 1$, M = 7 code given by Larsen [16]. The byte size was 6 bits. The results of the simulation are given in Table 2.3. The unit-memory code (i) had a decoding byte-error probability about one-half that of the gcd $(n_o, k_o) = 1$ code (ii) with the same state-complexity $(Mk_o = 6)$. This super-

iority of code (i) over code (ii) is due mainly to its greater free distance, 16 as opposed to 15. But we also see from Table 2.3 that the decoding byte-error probability of code (i) is about two-thirds that of code (iii) which has the same free distance 16 and greater state-complexity, 7 versus 6. This superiority of code (i) over code (iii) is due entirely to its byte-oriented nature.

We conclude that the unit-memory codes, because of their byte-oriented nature, appear attractive for use as the inner code in concatenated coding systems [2] where the outer code is a Reed-Solomon (RS) code over the alphabet $GF(2^{k_o})$, i.e. the bytes of the convolutional code are single digits for the RS code. For instance, code (i) above would be used with an RS code over $GF(2^6)$. We shall report in detail on the effectiveness of unit-memory codes in concatenated coding systems applications in Chapter V.

E_b/N_o	M=1 (k8,6) code		M-6,(3,1) code		M=7 (3,1) code	
(dB)	Byte-Error Prob.	95% Confidence	Byte-Error Prob.	95% Confidence	Byte-Error Prob.	95% Confidence
1.00	0.03050	±0.00533	0.04875	±0.00681	0.04000	±0.00619
1.25	0.02000	±0.00435	0.03250	±0.00561	0.02250	±0.00469
1.50	0.01175	±0.00329	0.02325	±0.00477	0.01400	±0.00372
1.75	0.00650	±0.00250	0.01275	±0.00350	0.01025	±0.00319

Table 2.3: Byte-error probability for Viterbi decoding of three R=1/3 convolutional codes on a simulated AWGN channel.

CHAPTER III

BYTE-ORIENTED DECODING ALGORITHMS FOR CONVOLUTIONAL CODE*

In the previous chapters, the terminology "byte", or "symbol", has appeared in several places to denote the basic grouping of digits such that the distance measure between any two sequences is the number of different "bytes" or "symbols" between them. In other words, the byte is defined to be the single "super-symbol" for which the Hamming distance measurement or the error rate, is concerned. The byte size is determined by the nature of the application. For example, 128 characters are included in the standard teletype alphabet, the byte size of tyletype signals is, then, 7 bits. In the case of concatenated coding, only the symbol error rate of the outer coding system can be controlled, the byte size is therefore the symbol size of the outer code.

For convenience of discussion, we shall assume that the byte size is integer multiples of k_o, and a byte covers m time instances. We shall denote an information byte encoded between time t and t+m-1 as

$$\underline{A}_T = [\underline{a}_t, \underline{a}_{t+1}, \cdots, \underline{a}_{t+m-1}],$$

where $T = (t+m-1)/m$ is implicitly implied. Further, if the encoding shift-register is initially loaded with zeros, after which $\underline{a}_1, \underline{a}_2, \cdots, \underline{a}_L$ are encoded, the followed by Mk_o zeros,

---

* Parts of this chapter are extracted from [23] and [25].

(i. e $\underline{a}_{L+1} = \underline{a}_{L+M} = 0$) to clear the encoder, L is called the frame length. In a frame, we have L' = L/m bytes. The byte error probability of a frame

$$P_{BE} = \frac{1}{L'} \sum_{T=1}^{L'} P(\hat{\underline{A}}_T \neq \underline{A}_T)$$

is the quantity used to measure the quality of information transmission. The Viterbi decoding algorithm [20], which is the maximum likelihood decoding algorithm for a convolutionally coded frame sent over a discrete memoryless channel, forms as its estimate $\hat{\underline{A}}_{[1,L']}$ the information sequence that maximizes the conditional probability.

$$P(\hat{A}_{[1,L']} \mid \underline{r}_{[1,L+M]}),$$

based on the sequence $\underline{r}_{[1,L+M]}$ received at the output of the discrete memoryless channel. Hence, this algorithm minimizes the <u>frame error probability</u>,

$$P_{FE} = P_r\ (\hat{\underline{A}}_{[1,L']} \neq \underline{A}_{[1,L']})$$

for any interesting channel, it must be true that

$$\lim_{L' \to \infty} P_{FE} = 1,$$

so that P_{FE} is not a meaningful optimality criterion for large frames. However, since we can write the byte-error-probability as

$$P_{BE} = \frac{1}{L'} \sum_{T=1}^{L'} P_r\ (\hat{\underline{A}}_T \neq \underline{A}_T),$$

it follows that

$$P_{BE} \leq P_{FE} \leq L'P_{BE}.$$

Hence, when L' is fairly small, it makes little difference whether P_{BE} or P_{FE} is minimized (which explains the appropriateness of Viterbi decoding when L' is small). The byte-probability, P_{BE}, is minimized by the decoding rule which, for each T, $1 \leq T \leq L'$, chooses its estimate $\hat{\underline{A}}_T$ as the byte which maximizes the conditional probability

$$P(\underline{A}_T, \underline{r}_{[1,L+M]}).$$

When the byte size is one bit, algorithms, similar to Viterbi's to accomplish this maximization have been proposed independently by Bahl _et al_ [21], and McAdam _et al_ [22]. If the byte size is smaller than or equal to the size of the memory M, but larger than a bit, the natural extension of these algorithms making use of the state property of the convolutional codes is self-evident. However, these algorithms require receipt of the entire frame $\underline{r}_{[1,L+M]}$ before decoding begins and so cannot be used without resynchronization (i.e. cannot be used when $L=\infty$). Moreover, their implementation requires storage which grows linearly in L and, hence are practical alternatives to Viterbi decoding only for moderately small L.

In the following section we derived a recursive procedure (incorporating the observation of Fritchman and Mixsell [24]) for "real-time minimum-bit-error probability(RTMbEP) decoding " for $(n_o,1)$ convolutional codes to minimize P_{bE} under the constraint that the decoding delay be limited to Δ branches. (Here we use the lower-case letter "b" to remind the reader that the byte size discussed is one bit.) In section 3.2 we formulate

the corresponding decoding algorithm and show that its storage requirements are independent of L. We also show the necessary modification needed to minimize the byte-error probability instead of the bit-error probability. This algorithm can be generalized to decode any (n_o,k_o) convolutional code, but, because of the particular importance of unit-memory convolutional codes, we show a modified version of the RTMBEP decoding algorithm for (n_o,k_o) unit-memory convolutional codes in section 3.3 which is much more efficient than that for general codes. For comparison purposes, we also formulate a "real-time", modified Viterbi decoding algorithm in section 3.4. Section 3.5 reports the results of using these decoding procedures on a simulated additive White Gaussian noise (AWGN) channel. It is concluded that the improvement in P_{BE} for the real-time minimal-byte-error probability decoding (RTMBEP) algorithm is not enough to justify the added complexity compared to Viterbi decoding in hard-decision applications but, as shown in the later chapters, the new algorithm offers advanges in soft-decision applications such as concatenated coding.

3.1 DERIVATION OF A REAL-TIME MINIMUM-BIT-ERROR PROBABILITY DECODING PROCEDURE FOR $(n_o,1)$ CONVOLUTIONAL CODES

As in all previous optimal (in some sense) decoding procedures for convolutional codes, we shall make important use of the encoder state which at time t is defined as the contents of the shift-register in Fig. 2.1, i.e, the M-tuple of past information bit

$$s_t = [a_{t-1}, a_{t-2}, \ldots a_{t-M}] \tag{3.1}$$

and where, by our convention, $a_i=0$ for $i\leq 0$ and $i>L$. As the term "state" implies, s_t completely accounts for the past history of the encoder input in the sense that s_t and the input segment $\underline{a}_{[t,L]}$ uniquely determine the output segment $\underline{b}_{[t,L+M]}$. By conditioning of the encoder state, the calculation of the probabilities required for the decision rule can be simplified.

The decoding rule which minimizes P_{BE} under the "real-time" constraint that $\hat{a}_t$ be decided from $\underline{r}_{[1,t+\Delta]}$ is that which chooses

$$\hat{a}_t = 0 \text{ if}$$

$$P(a_t=0|\underline{r}_{[1,t+\Delta]}) \geq \frac{1}{2} \tag{3.2}$$

and chooses $\hat{a}_t=1$ otherwise (where we have arbitrarily resolved ties in favor of the decision $\hat{a}_t=0$.) Since

$$P(a_t=0|\underline{r}_{[1,t+\Delta]}) = \frac{P(a_t=0,\underline{r}_{[1,t+\Delta]})}{P(\underline{r}_{[1,t+\Delta]})}$$

and since the probabilities on the righthand side of this latter equation can be expressed as the summation over all states of the joint probabilities including the state, we have

$$P(a_t=0\ \underline{r}_{[1,t+\Delta]} = \frac{\sum_s P(a_t=0,\underline{r}_{[1,t+\Delta]},s_{t+\Delta+1}=s)}{\sum_s P(\underline{r}_{[1,t+\Delta]},s_{t+\Delta+1}=s)}. \tag{3.3}$$

We now proceed to develop recusive formulas for the two probabilities appearing on the righthand side of (3.3).

For any t, t>1, we may write

$$P(\underline{r}_{[1,t]},s_{t+1}) = \sum_{s_t} P(\underline{r}_{[1,t]},s_t,s_{t+1}), \tag{3.4}$$

but also

$$P(\underline{r}_{[1,t]}, s_{t-1}, s_t) = P(\underline{r}_{[1,t-1]}, s_t) P(\underline{r}_t, s_{t+1} | \underline{r}_{[1,t-1]}, s_t)$$

$$= P(\underline{r}_{[1,t-1]}, s_t) P(\underline{r}_{t+1} | s_t), \qquad (3.5)$$

where we have made use of the state property and our assumption that the channel is memoryless. Writing

$$P(\underline{r}_t, s_{t+1} | s_t) = P(s_{t+1} | s_t) \; P(\underline{r}_t | s_t, s_{t+1}), \qquad (3.6)$$

we then use the fact seen from (3.1) that the state $s_{t+1} = [a_t, a_{t-1}, \ldots . a_{t-m+1}]$ has only two possible predecessors s_t, namely $[a_{t-1}, \ldots . a_{t-m+1}, 0]\,[a_{t-1}, \ldots, a_{t-m+1}, 1]$, to write

$$P(s_{t+1} | s_t) = \begin{cases} \frac{1}{2} & \text{if } s_t \varepsilon \rho(s_{t+1}) \\ 0 & \text{otherwise} \end{cases}, \qquad (3.7)$$

where here and hereafter we write $\rho(s)$ for the set containing the two possible <u>predecessors</u> of a state s. Finally, we note that, for s_{t+1} and s_t [and we write $\underline{b}(s_t, s_{t+1})$ for this branch] so that

$$P(\underline{r}_t | s_t, s_{t+1}) = P(\underline{r}_t | \underline{b}(s_t, s_{t+1})), \qquad (3.8)$$

and we note that this quantity is determined by the channel transition probabilities. Substituting (3.5), (3.6), (3.7) and (3.8) into (3.4) we have our desired recursion

$$P(\underline{r}_{[1,t]}, s_{t+1}) = \sum_{s_t \varepsilon \rho(s_{t+1})} \frac{1}{2} P(\underline{r}_t | \underline{b}(s_t, s_{t+1})) \; P(\underline{r}_{[1,t-1]}, s_t). \tag{3.9}$$

The starting value $P(\underline{r}_{[1,1]}, s_2) = P(\underline{r}, s_2)$ needed to apply the recursion is simply

$$P(\underline{r}_1, s_2) = \begin{cases} \frac{1}{2} P(\underline{r}_1 | \underline{b}(\underline{0}, s_2)) & \text{if } \underline{0} \varepsilon P(s_2) \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

where we have used the fact that $s_1 = \underline{0}$.

By an entirely analogous argument whose details we omit, the following recursion for the other probability on the righthand side of (3.3) may be obtained:

$$P(a_t, \underline{r}_{[1,t+i]}, s_{t+i+1}) = \sum_{s_{t+i} \varepsilon \rho(s_{t+i+1})} \frac{1}{2} P(\underline{r}_{t+i} | \underline{b}(s_{t+i}, s_{t+i+1})) \cdot$$

$$\cdot P(a_t, \underline{r}_{[1,t+i-1]}, s_{t+i}) \tag{3.11}$$

which we shall use for $M \leqq i \leqq \Lambda$. The starting value needed for this recursion is

$$P(a_t, \underline{r}_{[1,t]}, s_{t+M}) = P(\underline{r}_{[1,t]}, s_{t+M}) P(a | \underline{r}_{[1,t]}, s_{t+M})$$

$$= \begin{cases} P(\underline{r}_{[1,t]}, s_{t+M}) & \text{if } a_t \text{ is the last digit of } s_{t+M} \\ 0 & \text{otherwise} \end{cases},$$

so that the quanities obtained from the recursion (3.9) directly

provide the necessary initial conditions (3.12) to be used with the recursion (3.11). Hence we have now obtained a complete recursive procedure for performing real-time minimum-bit-error probability decoding of convolutional codes.

3.2 IMPLEMENTATION THE DECODING ALGORITHM

We now describe an algorithm for implementing the decoding procedure whose recursive basis was developed in the previous section. Our algorithm requires the storage of two real numbers for each of the 2^m encoder states. We denote these stored quantities for state s as f(s) and g(s). At time t+M-1 (in the algorithm) the first of these quantities will store the value

$$f(s) = P(\underline{r}_{[1,t+M-1]}, s_{t+M} = s), \tag{3.14}$$

and the second will store the value

$$h(s) = P(a_t=0, \underline{r}_{[1,t+j]}, s_{t+j+1} = s) \tag{3.15}$$

when j is indicated from M through Δ within the time interval [t+m-1, t+m] in the algorithm. When $j=\Delta$, we store the value of h(s) summed over all states which we denote as H_1. We keep the previous Δ-m values of this quantity and denote these stored values as $H_1, H_2, \ldots, H_{\Delta-m}$, that is,

$$H_i = \sum_s P(a_{t-1}=0, \underline{r}_{[1,t+\Delta-M-i]}, s_{t+\Delta-M-i+1}=s) \tag{3.16}$$

for $1 \le i \le \Delta-M$ will be the available value at time t+m-1 (in the algorithm). It follows that $H_{\Delta-M}$ and

$$\sum_s f(s) = \sum_s P(\underline{r}_{[1,t]}, s_{t+1} = s) \tag{3.17}$$

are the desired numerator and denominator on the right-hand side of (3.4) for the decoding of $\hat{a}_{t-\Delta}$ which is accomplished at time t+M-1 (in the algorithm). The only additional storage required is that for the received branches $\underline{r}_{t+M}, \underline{r}_{t+M+1}, \ldots, \underline{r}_{t+\Delta}$.

The recursions (3.10) and (3.14) directly correspond to the following:

<u>The RTMbEP Decoding Algorithm</u>

<u>Step 0</u>: Set t=1, set $f(s) = \frac{1}{2} P(\underline{r}_1 | \underline{b}(\underline{0},s))$ for the stored states s having $\underline{0}$ as a predecessor, and set f(s)= 0 for all other states. Set $G_i = 0$ for $1 \leq i \leq \Delta - M$.

<u>Step 1</u>: For t=1,2,..., M, make the replacements

$$f(s) \leftarrow \sum_{s' \varepsilon \rho(s)} \frac{1}{2} P(\underline{r}_t | \underline{b}(s',s)) f(s'), \text{ for all s.}$$

<u>Step 3</u>: For i= m+1, m+2, ...Δ make the replacements

$$h(s) \leftarrow \sum_{s' \varepsilon \rho(s)} \frac{1}{2} P(\underline{r}_{t+i} | \underline{b}(s',s)) h(s'), \quad \text{for all s.}$$

<u>Step 4</u>: If $t \leq \Delta$, go to step 5; otherwise set $\hat{a}_{t-\Delta}=0$ if

$$\sum_s f(s)/H_{\Delta-M} \geq \frac{1}{2}$$

and set $\hat{a}_{t-\Delta} = 1$ otherwise.

<u>Step 5</u>: Increase t by 1, make the replacements

$$H_{\Delta-i+1} \leftarrow H_{\Delta-i}, \text{ for } i = m+1, \ldots, \Delta-1;$$

and set

$$H_1 = \sum_s g(s).$$

Step 6: Make the replacement

$$f(s) \leftarrow \sum_{s' \varepsilon \rho(s)} \frac{1}{2} P(\underline{r}_t | \underline{b}(s',s)\ f(s') \text{ for all } s.$$

and then return to step 2

(Note: For simplicity we have omitted the obvious "end game" modifications needed when t>L which, of course, are necessary only if a finite frame length is used. It should also be pointed out that our "trick" of storing the Δ-M past values of the g(s) summation actually results in a "true" decoding delay of 2Δ-M branches since we require the use of $\underline{r}_{t+\Delta-M}$ in step 4 at time t (in the algorithm) when $a_{t-\Delta}$ is decoded. To reduce the true decoding delay to Δ branches requires the storage of Δ-M+2 branches rather than 2 real numbers per state since f(s) must be updated by Δ-M+1 branches and the Δ-M previous values of f(s) stored for each state, or, alternatively, the storage of 3 real numbers and considerable extra computation within the algorithm).

The algorithm as given above is directly suited for software implementation. It should be noted that steps 3,4, and 6 call for both addition and multiplication of the computed probabilities that floating point arithmetic would normally be chosen for the calculation. (This contrasts with the Viterbi algorithm when fixed arithmetic is normally used as will be seen in section 3.4) For each t, a total of Δ-M+1 calculations are made in steps 3 and 6, each involving a sum over all 2^M states (whereas, as shall see, only one similar calculation is needed for the Viterbi algorithm.

A hardware realization of the real-time minimal-bit-error algorithm could be made employing 2^M micro-processors each of which corresponds to an encoder state s. Each micro-processor, would receive f(s') and h(s') from the two micro-processors corresponding to the two states s' which are predecessors of s and with the aid of the received branches as inputs, would compute new values of f(s) and h(s) pass these values on in-turn to the two microprocessors corresponding to the two states for which s is a predecessor. Each micro-processor would execute $\Delta-M+1$ computational cycles for each t (as contrasted to one cycle for the microprocessors in a hardware Viterbi decoder).

The implementation requires the micro-processor to perform both multiplication and addition in floating point arithmetic. Similar to what is done in the Viterbi decoder, we would still quantize the logarithm of the probabilities into integers. The multiplication of the probabilities then is performed by adding their quantized logarithms, whereas addition in probabilities is carried out by table look-up. Since the result is relatively insensitive to the addition operation and to the quantization, very little inaccuracy is caused by this technique. With the aid of Read-Only-Memory (ROM) to store the addition table, this implementation of the RTMbEP algorithm, though considerable more complicated than the Viterbi algorithm, might not be excessively expensive.

If the symbol size, m, considered is smaller or equal to the size of the memory of the convolutional code, but larger than one bit, this algorithm can be modified easily to minimize the symbol-

error-rate. In this case the decision rule is to choose $\hat{A}_T$ as the estimate if it maximizes the probability

$$P(\underline{A}_T \mid \underline{r}_{[1,t+\Delta]}) = \frac{P(\underline{A}_T, \underline{r}_{[1,t+\Delta]})}{P(\underline{r}_{[1,t+\Delta]})}, \qquad \text{for } m < \Delta$$

But since there are always, 2^m possible $\underline{A}_T$ in each cycle of the decoding procedure, we obtain very little advantage from calculating the last possible $P(\underline{A}_T, \underline{r}_{[1,t+\Delta]})$ by substracting all other $P(\underline{A}_T, \underline{r}_{[1,t+\Delta]})$'s already calculated from $P(\underline{r}_{[1,t+\Delta]})$ as we did in the RTMbEP algorithm. Thus, we shall not store H's. But we required the storage of 2^m real numbers to keep $P(\underline{A}_T, \underline{r}_{[1,t+\Delta]})$ which we shall denote as $H'(\underline{A}_T)$. The modified algorithm is the following:

<u>The RTMBEP Decoding Algorithm</u> (forward)

<u>Step 0</u>: Set t=1, set $f(s) = \frac{1}{2} P(\underline{r} \mid \underline{b}(0,s))$ for the two states having $\underline{0}$ as a predecessor, and set f(s) = 0 for all other states.

<u>Step 1</u>: For t=1,2,..., M make the replacements

$$f(s) \leftarrow \sum_{s' \varepsilon \rho(s)} \frac{1}{2} P(\underline{r}_t \mid \underline{b}(s',s))\, f(s'), \text{ for all } s.$$

<u>Step 2</u>: For each of all 2^m possible $\underline{A}_T$, set h(s) = f(s) if the last m digit of s are $\underline{A}_T$ and set h(s)=0 otherwise.

<u>Step 3</u>: For i=M+1, M+2,..., Δ, make the replacements

$$g(s) \leftarrow \sum_{s' \varepsilon \rho(s)} \frac{1}{2} P(\underline{r}_{t+i} \mid \underline{b}(s',s)\, h(s') \text{ for all } s.$$

<u>Step 4</u>: Set $H'(\underline{A}_t) = \sum_s h(s)$,

If all the 2^m possible $H'(\underline{A}_T)$ have not been calculated go to step 2. Otherwise, choose $\hat{\underline{A}}_T = \underline{A}_T$ such that $H'(\underline{A}_t)$ is the maximum. Set i=0.

Step 5: Increase t by 1 and make the replacement

$$f(s) \leftarrow \sum_{s' \varepsilon \rho(s)} \frac{1}{2} P(\underline{r} | \underline{b}(s',s))\ f(s'), \text{ for all } s.$$

Increase i by 1.

Step 6: If i=m go to step 2, otherwise return to step 5.

We should note here that although we have only 1/m as may decisions to make in each frame as in the RTMbEP decoding case, each decision requires the calculation of $P(\underline{A}_T, \underline{r}_{[1,t+\Delta]})$ for 2^m values of $\underline{A}_T$, whereas only one calculation is required in the bit case. Thus, the number of calculations can be prohibitively large for even moderate byte size. Fortunately, we find that we can use a backwards recursion (which will be discussed in the next section), instead of the forward recursion described above, to handle the decoding procedure more efficiently for all (n_o, k_o) convolutional code with $k_o \neq 1$, and most efficiently for the unit-memory codes. But this backwards recursion offers no significant advantage for decoding $(n_o, 1)$ convolutional codes, and moreover, since the forward recursion discussed in this section seems to be a natural approach to set up "real-time" decoders, we feel it warrants the description given here.

3.3 REAL-TIME MINIMAL-BYTE-ERROR PROBABILITY DECODING ALGORITHM FOR UNIT-MEMORY CONVOLUTIONAL CODES.

In the previous chapter, we have noted that the unit-memory (n_o,k_o) convolutional codes always have the largest minimal free distance among all convolutional codes of the same state-complexity, and that the state trellis structure of the unit-memory codes are quite different from that of $(n_o,1)$ convolutional codes. Therefore, we pay particular attention to implementing decoding procedures for unit-memory codes. As commented in the last section, we are able to make use of the fact that every one of the 2^{k_o} states of the unit-memory code can reach any particular state at the next clock instant in order to derive a backwards recursive decoding algorithm, which is much more efficient than the forward recursion algorithm discussed previously. For the backwards algorithm, the joint probabilities

$$P(\underline{a}_t, \underline{r}_{[1,t+\Delta]})$$

for all the 2^{k_o} possible $\underline{a}_t$'s can be calculated in a single cycle of the algorithm.

The decision rule for this algorithm is to choose the estimate $\underline{a}_t$ for the information branch

$$\underline{a}_t = [a_{t1}, a_{t2}, \ldots, a_{tk_o}]$$

as the symbol which maximizes the conditional probability

$$P(\underline{a}_t \mid \underline{r}_{[1,t+\Delta]}).$$

Similar to the approach in the previous section, we have

$$P(\underline{a}_t \mid \underline{r}_{[1,t+\Delta]}) = \frac{P(\underline{a}_t, \underline{r}_{[1,t+\Delta]})}{P(\underline{r}_{[1,t+\Delta]})},$$

or equivalently

$$P(\underline{a}_t \mid \underline{r}_{[1,t+\Delta]}) = \frac{P(\underline{a}_t, \underline{r}_{[1,t+\Delta]})}{\sum_{\underline{a}_t} P(\underline{a}_t, \underline{r}_{[1,t+\Delta]})}. \tag{3.18}$$

We shall proceed to derive a recursive algorithm to calculate the numerator on the righthand side of equation (3.18) for all possible $\underline{a}_t$'s, and simultaneously obtain their sum as the denominator on the righthand side of (3.18). Since

$$\begin{aligned} P(\underline{a}_t, \underline{r}_{[1,t+\Delta]}) &= P(s_{t+1}, \underline{r}_{[1,t]}, \underline{r}_{[t+1,\ t+\Delta]}) \\ &= P(s_{t+1}, \underline{r}_{[1,t]})\ P(\underline{r}_{[t+1,t+\Delta]} \mid s_{t+1}, \underline{r}_{[1,t]}) \end{aligned}$$

and since the channel is memoryless, we have

$$P(\underline{a}_t, r_{[1,t+\Delta]}) = P(s_{t+1}, \underline{r}_{[1,t]})\ P(\underline{r}_{[t+1,t+\Delta]} \mid s_{t+1}) \tag{3.19}$$

The last term of equation (3.19) can be expressed as the summation over all states of the joint probabilities including the state to give

$$P(\underline{a}_t, \underline{r}_{[1,t+\Delta]}) = P(s_{t+1}, \underline{r}_{[1,t]})\ \left(\sum_{s_{t+2}} P(\underline{r}_{[t+1,t+\Delta]}, s_{t+2} \mid s_{t+1})\right).$$

But

$$P(\underline{r}_{[t+1,t+\Delta]}, s_{t+i+1} | s_{t+1})$$

$$= \sum_{s_{t+i+2}} P(\underline{r}_{t+i}, \underline{r}_{[t+i+1,t+\Delta]}, s_{t+i+1}, s_{t+i+2} | s_{t+i}) \qquad (3.20)$$

$$= P(\underline{r}_{t+i}, s_{t+i+1})\left(\sum_{s_{t+i+2}} P(\underline{r}_{[t+i+1,t+\Delta]}, s_{t+i+2} | s_{t+i+1})\right)$$

where here again we have made use of the fact that the channel is memoryless and of the state property. Then it is clear that if we are given the transition probabilities

$$P(\underline{r}_{t+i}, s_{t+i+1} | s_{t+i}) \qquad 1 \leq i \leq \Delta \qquad (3.21)$$

for all possible encoded branches, we can obtain the last term of equation (3.19) by successively applying the recursion of equation (3.20) for $i=\Delta-1, \Delta-2, \ldots, 1$. The first term on the righthand side of equation (3.19) can be obtained recursively by

$$P(s_{t+1}, \underline{r}_{[1,t]}) = \sum_{s_t} P(s_t, \underline{r}_{[1,t-1]})\, P(\underline{r}_t, s_{t+1} | s_t) \qquad (3.22)$$

as derived in section 3.1. Hence, we now have obtained a complete recursive procedure for RTMBEP decding algorithm for the unit-memory (n_o, k_o) convolutional codes.

We now describe an algorithm for implementing the decoding procedure just derived. Assuming the 2^{2k_o} transition probabilities indicated in (3.21) are available to us, we find that our algorithm requires again the storage of two real number for each of the 2^{k_o}

states. We denote these two stored quantities for state s as f(s) and h(s). Similarly, at time t in the algorithm, the first of these quantities will store the value

$$f(s) = P(\underline{r}_{[1,t]}, s_t = s),$$

and the other will store the value

$$h(s) = P(\underline{r}_{[t+j,t+\Delta]} \mid s_{t+j} = s)$$

$$= \sum_{s'} P(\underline{r}_{[t+j,t+\Delta]}, s_{t+j+1} = s' \mid s_{t+j} = s)$$

The only additional storage required is that for the received branches $\underline{r}_{t+1}, \underline{r}_{t+2}, \ldots, \underline{r}_{t+\Delta}$.

The recursions of (3.21) and (3.22) directly correspond to the following:

<u>The RTMBEP Decoding Algorithm for Unit-Memory Convolutional Codes</u> (backwards)

<u>Step 0</u>: Wait until $\underline{r}_1, \underline{r}_2, \ldots, \underline{r}_{\Delta+1}$ are received, set t=1, and set $f(s) = P(\underline{r}_1, \underline{s} \mid \underline{0})$ for all states s.

<u>Step 1</u>. Set $h(s) = \sum_{s'} P(\underline{r}_{t+\Delta}, \underline{s}_{t+\Delta+1} = s' \mid s_{t+\Delta} = s)$ for all states s.

<u>Step 2</u>. For $j = \Delta-1, \Delta-2, \ldots, 1$ make the replacements

$$h(s) \leftarrow \sum_{s'} P(\underline{r}_{t+j}, s' \mid s) \quad h(s'), \text{ for all states } s.$$

<u>Step 3</u>. Put out the estimate

$$\hat{a}_t = \{s: s \text{ maximizes } f(s)\ h(s))\}$$

and the reliability indicator

$$P(\hat{\underline{a}}_t \mid \underline{r}_{[1,t+\Delta]}) = \frac{f(s=\underline{a}_t)\ h(s=\underline{a}_t)}{\sum_s f(s)\ h(s)}$$

Step 4. Make the replacement

$$f(s) \leftarrow \sum_{s'} P(\underline{r}_t, s \mid s')\ f(s')$$

and then return to step 1.

The obvious "end game" modification is again omitted here.

It should also be pointed out that this algorithm is computationally similiar to the RTMbEP decoding algorithm proposed in section 3.1 in the sense that each decision requires only one cycle of the algorithm, but this algorithm decodes the whole byte each cycle instead of one bit each time. But the trellis is fully-connected and the summation is therefore taken over all states instead of two specific predecessors. The backwards recursion of step 2 enables us to calculate $P(\underline{a}_t \mid \underline{r}_{[1,t+\Delta]})$ for all $\underline{a}_t$ at the same time and, therefore, is more powerful than the forward recursion of step 3 in the RTMBEP (forward) decoding algorithm. However, if we wish to minmize bit-error rate, the advantage of the backwards recursion no longer exists. Further, the algorithm shares much the same difficulty in hardware implementation as the forward RTMBEP decoding algorithm described in section 3.2, therefore, neither RTMBEP decoding algorithms are attractive alternatives to Viterbi

decoding when only hard-decisions are required. But the decoding delay required for unit-memory convolutional codes is usuallly much smaller (as counted in branches) than that required for $(n_o, 1)$ convolutional codes of moderate momory size so that the disadvantage of computing Δ-M branchs of state probabilities for each of the RTMBEP decisions is relatively minor in the case of a unit-memory code.

Although we have repeatedly emphasized that the RTMBEP decoding algorithm requires the calculation of Δ-M branches of state probabilities, which seems a significant disadvantage of the RTMBEP decoding algorithm as an alternative to Viterbi decoding algorithm, this is primarily due to the real-time constraint of "fixed" decoding delay and the fact that the algorithms are written in such a way as to minimize the use of storage. If we relax this constraint of "fixed" decoding delay but still perform "real-time" decoding, it is then possible to reduce the amount of computation by increasing the storage requirements. This is particularly true for the unit-memory convolutional codes because of the short decoding delay they require. In the following we shall demonstrate this fact.

We note that

$$P(\underline{a}_t \mid \underline{r}_{[1,t+\Delta]}) = P(\underline{a}_t \mid \underline{r}_{[1,t]})\ P(\underline{a}_t \mid \underline{r}_{[t+1,t+\Delta]}),$$

and

$$P(\underline{a}_{t+1} \mid \underline{r}_{[1,t+\Delta]}) = P(\underline{a}_{t+1} \mid \underline{r}_{[1,t+1]})\ \ P(\underline{a}_{t+1} \mid \underline{r}_{[t+2,t+\Delta]}),$$

but $P(\underline{A}_t | \underline{r}_{[t+1,t+\Delta]})$ and $P(\underline{A}_{t+1} | \underline{r}_{[t+2,t+\Delta]})$ are calculated by the same backwards recursion of step 2 in the same cycle. It is thus possible to store the values of $P(\underline{a}_{t+1} | \underline{r}_{[t+2,t+\Delta]})$ while calculating $P(\underline{a}_t | \underline{t}_{[t+1,t+\Delta]})$. We can obtain $P(\underline{a}_{t+1} | \underline{r}_{[1,t+\Delta]})$ without going through the backwards recursion again. As will be seen from Table 3.2, any decoding delay greater than 8 branches will perform virtually the same as $\Delta=8$. We can then let Δ greater than 8, (which is now the maximal decoding delay) and store the values of $P(\underline{a}_{t+i-1}, \underline{r}_{[t+i,t+\Delta]})$ for i= 2, 3,....., $\Delta-7$, while calculating $P(\underline{a}_{t+1}, \underline{r}_{[t+1,t+\Delta]})$. These values will be used immediately to estimate $\underline{a}_{t+1}, \underline{a}_{t+2}, \ldots, \underline{a}_{t+\Delta-8}$. Therefore, $\Delta-7$ branches of decisions can be made with Δ branches of backwards recursion. Δ can be made as large as the size of storage permits, in the limit, when Δ equals to the frame length, the real-time algorithm becomes the algorithm proposed by Bahl _et al_. [21], and the total amount of computation is about twice that of the Viterbi decoder. For an example, we let $\Delta=16$, and let the actual decoding delay vary between 8 and 16, for every 8 decoded branches, only 16 passes of the backwards recusion are required in contrast to the 64 passes required in the "fixed" decoding delay technique.

Natural extensions to implement similar decoders for other (n_o, k_o) convolutional codes $(M \neq 1)$ are obvious and are omitted.

3.4 REAL-TIME VITERBI DECODING

As mentioned in the beginning of this chapter, the Viterbi decoding algorithm chooses $\underline{a}_{[1,L]}$ to be the information sequence which maximizes the conditional probability $P(\underline{a}_{[1,L]} | \underline{r}_{[1,L+M]})$. The following decoding rule, which we call <u>real-time Viterbi decoding</u> (RTV), is the natural modification of this rule to satisfy the "real-time constraint" that $\underline{A}_T$ be decoded from $\underline{r}_{[1,t+\Delta]}$ $(m < \Delta)$: Choose $\hat{\underline{A}}_T$ as the byte $\underline{A}_T$ in the information segment $\underline{a}_{[1,t+\Delta]}$ which maximizes the conditional probability $P(\underline{a}_{[1,t+\Delta]} / \underline{r}_{[1,t+\Delta]})$.

In keeping with our previous notation, we let

$$\underline{s}_{[t,t']} = [s_t, s_{t+1}, \ldots, s_{t'}]$$

denote the sequence of encoder states from time t to time t' inclusive, and let the $(n_o, k_o=1)$ convolutional codes be first considered. It follows from (3.1) that $\underline{s}_{[1,t+\Delta+1]}$ and $\underline{a}_{[1,t+\Delta]}$ uniquely determine one another and, moreover, that $\underline{A}_T$ is the first m digits of $[s_{t+1}, s_{t+2}, \ldots, s_{t'+1}]$. Hence we may rephrase the real-time Viterbi decoding rules as: Choose $\underline{A}_T$ as the first m digits of the state subsequence $\underline{s}_{[t+1,t'+1]}$ in the state sequence $\underline{s}_{[1,t+\Delta+1]}$ which maximizes the conditional probability

$$P(\underline{s}_{[1,t+\Delta+1]} | \underline{r}_{[1,t+\Delta]}) = \frac{P(\underline{s}_{[1,t+\Delta+1]}, \underline{r}_{[1,t+\Delta]})}{P(\underline{r}_{[1,t+\Delta]})} . \quad (3.23)$$

Since the denominator on the righthand side of (3.23) is independent of $\underline{s}_{[1,t+\Delta+1]}$, we can equivalently maximize the

numerator alone.

To obtain a recursion for the numerator in (3.23) we use the same arguments as in Section 3.2 which, for $t \geq 2$ give

$$P(\underline{s}_{[1,t+1]},\underline{r}_{[1,t]}) = P(\underline{s}_{[1,t]},\underline{r}_{[1,t-1]})\ P(s_{t+1},\underline{r}_t \mid \underline{s}_{[1,t]},\underline{r}_{[1,t-1]})$$

$$= P(\underline{s}_{[1,t]},\underline{r}_{[1,t-1]}\ P(s_{t+1},\underline{r}_t \mid s_t)$$

$$= P(\underline{s}_{[1,t]},\underline{r}_{[1,t-1]})P(s_{t+1} \mid s_t)\ P(\underline{r}_t \mid s_t,s_{t+1})$$

$$= \begin{cases} \frac{1}{2}\ P(\underline{r}_t \mid \underline{b}(s_t,s_{t+1}))\ P(s_{[1,t]},\underline{r}_{[1,t-1]}) & \text{if } s_t \varepsilon P(s_{t+1}) \\ 0 & \text{otherwise} \end{cases} \qquad (3.24)$$

Equation (3.24) is our desired recursion. The initial condition to be used for t=2 is

$$P(\underline{s}_{[1,2]},\underline{r}_{[1,1]}) = \begin{cases} \frac{1}{2}\ P(\underline{r}_1,\ \underline{b}(\underline{0},s_2)) & \text{if } \underline{0}\varepsilon P(s_2) \\ 0 & \text{otherwise} \end{cases}$$

Just as for the ordinary Viterbi algorithm, the key to the efficient implementation of real-time Viterbi decoding is the fact [readily seen from (3.24)] that the best state sequence [in the sense of maximizing the joint probability on the lefthand side of (3.24)] $\underline{s}_{[1,t+1]}$ with $s_{t+1} = s$ must be the extension of the best sequence $s_{[1,t]}$ with $s_t = s'$ or $s_t = s''$ where s' and s'' are the predecessors of s. Hence at each time t the only storage required for each state is the best sequence to that state. Actually, since real-time Viterbi decoding requires only knowledge

of the first digit of the state Δ states previously along the sequence, we need store only Δ bits per state together with the joint probability needed for the recursion in (3.24). We write $B_i(s)$, $1 \le i \le \Delta$ and $h(s)$ for these stored quantities which at time t (in the algorithm) have the values

$$h(s) = P(\underline{s}^*_{[1,t+1]}, \underline{r}_{[1,t]})$$

where $\underline{s}^*_{[1,t+1]}$ is the best path $\underline{s}_{[1,t+1]}$ with $s_{t+1}=s$ and

$B_i(s)$ is the first digit of state s_{t+1-1} in the patch $\underline{s}^*_{[1,t+1]}$.

Then we may state the following:

<u>The Real-Time Viterbi (RTV) Decoding Algorithm</u>

<u>Step 0</u>: Set t=1, set $h(s)= \frac{1}{2} P(\underline{r}_1 | \underline{b}(\underline{0},s))$ for the two states s having $\underline{0}$ as a predecessor, and set $h(s) = 0$ otherwise. Set $B_i(s) = 0$ for $1 \le i \le \Delta$ and all states s.

<u>Step 1</u>: If $t \le \Delta$, go to Step 2. Otherwise set

$$\underline{A}_{t-\Delta} = B_{\Delta-m+1}(s) \quad B_{\Delta-m+2}(s) \ldots B_{\Delta}(s)$$

where s is the state for which h(s) is maximum. Set t'=0.

<u>Step 2</u>: Increase t by 1. Make the replacement $B_{\Delta-i+1}(s) \leftarrow B_{\Delta-i}(s)$ for $i = 1, 2, \ldots, \Delta-1$ and for all s. Increase t' by 1.

<u>Step 3</u>: For each s, make the replacement $h(s) \leftarrow \frac{1}{2} P(\underline{r} | \underline{b}(s',s)) h(s')$ where s' is the predecessor of s which maximizes the replacing quanity, and set

$B_1(s)$ equal to the first digit of s'.

Step 4: If $t' = m$, go to Step 1; otherwise, return to Step 2.

The algorithm just given is directly suited for software implementation. Since the algorithm calls for only multiplication of the computed probabilities and selection of a maximum, logarithms may be used with the result that only computer additions are required and hence fixed-point arithmetic would normally be chosen for the calculation. For each t, only one maximum over all states need be taken, an operation equivalent in complexity to a sum over all states as is required $\Delta+i$ times in the algorithm of the preceeding section. In a hardware realization of the real-time Viterbi algorithm, the microprocessor corresponding to state s would receive h(s') from the two microprocessor corresponding to the two predecessors s' of s and, with the aid of the received branch, would compute the new value of h(s) and pass this value on in turn to the two microprocessors for the states having s as a predecessor. Each microprocessor would execute only one computational cycle for each m time units and would be somewhat simpler than the microprocessor described in the preceeding section since only one quanity, h, (rather than two, f and h) would be processed and only additions need be performed.

It should be emphasized that real-time Viterbi decoding may be used for $L=\infty$, i.e, when the convolutional encoder is not periodically resynchronized. Moreover, ordinary Viterbi decoding can be considered the special case of real-time Viterbi decoding for

finite L when $\Delta=L+M-1$. Generalization of the algorithm for general (n_o,k_o) convolutional codes $(k_o \neq 1)$ is straight-forward and is omitted in the discussion.

3.5 SIMULATION RESULTS

To evaluate the performance of real-time minimum-byte-error probability decoding (hereafter called RTMBEP decoding), a rate 1/2, (2.1) convolutional coding system was implemented for a simulated additive white Gaussian noise (AWGN) channel with binary antipodal signaling and with 8-level output quantization for the bit-by-bit decoding [(RTMbEP). Also implemented was an (18.6) unit-memory convolutional coding system] (byte size equal to 6) for the same channel. The results of the simulation for the bit case are given in Table 3.1 where E_b is the energy per information bit, N_o is the one-sided noise power spectral density, and K=M+1 is the constraint lengh measured in information bits. The results for the unit-memory convolutional code are given in Table 3.2. From the tables, we observe very little improvement of P_{BE} or P_{bE} when the RTMBEP or the RTMbEP algorithm is used in place of Viterbi decoding. Approximately 0.1 dB to 0.2 dB of improvement in E_b/N_o is observed in the bit case when the channel is very noisy (E_b/N_o = 1) while no improvement is observed when the signal is strong. Similar but slightly smaller gains are seen in the case of RTMBEP decoding of a unit-memory convolutional code. Since it is of no practical interest to operate a coding system in such a noisy channel where the RTMBEP (or RTMbEP) algorithm shows a slight advantage over the Viterbi decoding, we conclude that the slightly better performance of the RTMBEP (or RTMbEP) decoding algorithm does not justify the increased decoder complexity required

$\frac{E_b}{N_o}$	k	L(bits)	No. Frames Decoded	Δ	P_{bE} RTMbEP Decoding	Real-Time Viterbi	Ordinary Viterbi
0dB	3	2400	1	9	.109	.113	.119
2dB	3	2400	2	9	.0165	.0186	.0173
4dB	3	2400	5	9	.00083	.00083	.00083
0dB	5	2400	1	19	.154	.169	
2dB	5	2400	2	19	.0165	.0184	
4dB	5	2400	5	19	.00033	.0033	

Table 3.1: Results of Decoding of (2.1) convolutional codes for a simulated AWGN channel.

$\frac{E_b}{N_o}$	L'(bytes)	No. Frames Decoded	Δ	P_{BE} REMBEP Decoding	Real-Time Viterbi
1.00dB	400	10	8	0.02950	0.03050
1.25dB	400	10	8	0.01925	0.02000
1.50dB	400	10	8	0.01150	0.01175
1.75dB	400	10	8	0.00625	0.00650

Table 3.2: Results of Decoding of the (18,6) unit-memory convolutional code for a simulated AWGN channel.

when only hard-decisions are made.

In Tables 3.3 and 3.4, we show the effect of the decoding delay on P_{bE} and P_{BE} for the M=2, (2.1) code and the (18.6) unit-memory code. From this table we see that the error proabilities decrease as longer decoding delay is employed. However, they saturate rapidly. From the tables, we conclude that a decoding delay (in branches) of about 3(M+1) to 4(M+1) is sufficient for near-optimal performance with both the RTMBEP and the RTMbEP algorithm.

$\frac{E_b}{N_o}$	L(bits)	No. of frames Decoded	Δ (bits)	P_{bE}
0dB	2400	1	4	0.133
0dB	2400	1	5	0.125
0dB	2400	1	7	0.113
0dB	2400	1	9	0.109
0dB	2400	1	19	0.106
0dB	2400	1	29	0.102
2dB	2400	2	4	0.0325
2dB	2400	2	9	0.0165
2dB	2400	2	19	0.0140

Table 3.3: Effect of Decoding Delay for RTMbEP Decoding of the M=2, (2.1) Convolutional Code for a Simulated AWGN Channel

$\frac{E_b}{N_o}$	L'(bytes)	No. of frames decoded	Δ (bytes)	P_{BE}
1.25dB	400	10	4	0.02850
1.25dB	400	10	6	0.02475
1.25dB	400	10	8	0.01925
1.25dB	400	10	16	0.01925

Table 3.4: Effect of Decoding Delay for RTMBEP Decoding of the (18.6) Unit-Memory Convolutional Code for a Simulated AWGN Channel.

3.6 REMARKS

We have given a fairly comprehensive treatment of optimal real-time decoding of convolutional codes and introduced algorithms to minimize the decoding byte (as well as bit) error probability. We also stated a real-time Viterbi decoding algorithm which, although not previously given in the literature, is probably the form of the Viterbi algorithm which has actually been used in many previous investigations.

Our conclusion from simulations of the RTMBEP decoding algorithm is that, although it does not reduce P_{BE} sufficiently to be a practical alternative to Viterbi decoding in hard-decision applications, the fact that it provides a direct measure of the quality of its decoded decisions, makes it an attractive candidate for the inner decoder in concatenated coding systems, as we shall see in the later chapters. In particular, in a system which will be discussed in section 5.2, the RTMBEP decoder receiving feedback from an outer decoder performs much better than the Viterbi decoder even in terms of hard-decisions.

CHAPTER IV

OPTIMAL SOFT-DECISION DEMODULATION *

In chapter III, we derived algorithms which minimize byte-error-probability and furnish the a posteriori probability $p(\underline{A}_T | \underline{r}_{[1,t+\Delta]})$ as the reliability function of the estimated symbol $\underline{A}_T$. To make effective use of this reliability information for the outer decoder in a concatenated coding system, it is necessary to process the information in such a way that the required complexity of the outer decoder is within practical limits. In both the generalized minimum distance (GMD) and the errors-and-erasures decoding proposed by Forney [2], the reliability information is either hard-limited or quantized according to certain empirical rules. In practice, it is necessary to quantize the reliability information into a few levels according to a set of thresholds. Before studying the feasibility of optimizing the thresholds analytically, we are motivated to study in this chapter a similar but much simpler problem, that is, the problem of optimizing soft-decision demodulation.

The block diagram of a one-way, coded communication system is given in Figure 4.1. Comparing Figure 4.1 to Figure 1.1, which shows a block diagram representation of a one-way, concatenated coding system, we find that the two problems are similar in the sense that both systems contain a discrete channel which emits soft-decision symbols that are fed into a decoder.

---

*Most of this chapter is taken from [29]. A portion of this chapter was presented orally by the author at the IEEE International Symposium on Information Theory, Notre Dame, Indiana, October 27-31, 1974.

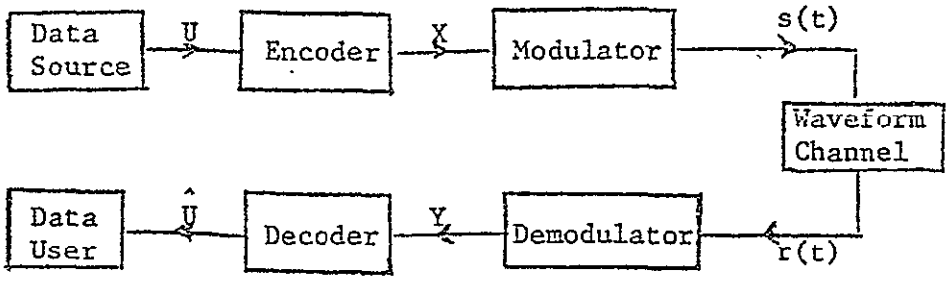

Figure 4.1. A one-way, coded, digital communications system.

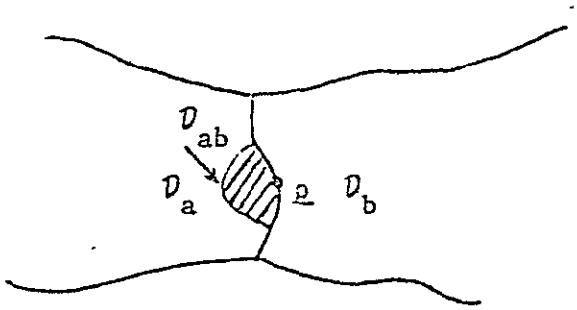

Figure 4.2. The variation of the decision regions $\mathcal{D}_a$ and $\mathcal{D}_b$ by transfer of the small region $\mathcal{D}_{ab}$.

But in the case of soft-decision demodulation, the messages are assumed to be sent independently, which in turn implies that the signals are received independently provided that the channel is memoryless; whereas the messages are not independent in the case of soft-decision decoding. Although this subject deviates somewhat from our main-line of research, we find that it is of practical interest by itself and also gives guidance to the task of optimizing soft-decision decoding.

From Figure 4.1, it is apparent that modulation and coding are both aspects of the "signal design" problem, whereas demodulation and decoding are both aspects of the "signal detection" problem. The natural question then is how to coordinate the design of the modulation system and the coding system so as to produce an efficient and effective communications system.

Suppose that the modulator is M-ary; then, without loss of generality, we may consider the modulator input alphabet to be the set $\{0, 1, 2, \ldots, M-1\}$. Suppose the demodulator is restricted to J different decisions, then we may take its output alphabet to the $\{0, 1, 2, \ldots, J-1\}$. We say that the demodulator makes "hard-decisions" or "soft-decisions" according as to whether $J=M$ or $J>M$ respectively. Clearly, the "classical" modulation system design criterion of "error probability" is applicable only

for hard-decisions. Unfortunately for classicists, the use of a hard-decision demodulator generally reduces substantitally the effectiveness of the coding system.

Wozencraft and Kennedy [10] were the first to suggest that the proper modulation system design criterion is the "cut-off rate," R_o, of the M-input, J-output discrete memoryless channel (DMC) presented by the modulation system to the coding system. This DMC is completely described by the transition probabilities, $P(j|m)$, that the demodulator decision is j given that the modulator input was m, $0 \leq j < J$, $0 \leq m < M$. Mathematically, the cut-off rate is given by

$$R_o = -\log_2 \left\{ \min_Q \sum_{j=0}^{J-1} \left[\sum_{m=0}^{M-1} Q(m) \sqrt{P(j|m)} \right]^2 \right\} \qquad (4.1)$$

where Q is a probability distribution for the channel input letters, i.e., $Q(m) \geq 0$ for all m and $Q(0) + Q(1) + \ldots + Q(M-1) = 1$

Wozencraft and Kennedy were led to the choice of R_o (or as it was then usually denoted, "R_{comp}") because R_o is the upper limit of code rates for which the average decoding computation per digit is finite when sequential decoding is used. More recently, Massey [11] has pointed out a more persuasive reason for choosing R_o as a design criterion. Viterbi [20] has shown that, when convolutional coding is used with maximum likelihood decoding on the DMC, then the decoding error probability is upper bounded by

$$P_e < c_R L 2^{-NR_o}, \text{ if } R < R_o, \qquad (4.2)$$

where N is the code constraint length, R is the code rate (number of data bits per decoded letter), L is the number of bits encoded

and c_R is an unimportant constant independent of N and L. Hence, as Massey observed, the single number R_o determines both a range of rates over which reliable operation is possible as well as a measure of the necessary coding complexity to obtain a specified error probability. R_o is thus even more informative than the channel capacity of the DMC which, although it determines the entire region of rates over which reliable communications is possible, says nothing about the coding complexity needed for a specified decoding error probability at any given code rate.

In the same paper [11], Massey established a number of fundamental results about modulation systems under the R_o criterion. He gave a general expression for R_o for unquantized demodulation ($J=\infty$), and proved that, for any given M, the M-ary simplex signal maximizes the unquantized R_o for the additive white Gaussian noise (AWGN) channel. For binary modulation ($M=2$) and any given J, Massey also gave a necessary condition for the demodulator decision regions to be optimal, and showed how to use this condition as the basis of an iterative computational technique for finding the optimal decision regions.

In this chapter, we extend Massey's necessary condition for optimal demodulation to the non-binary ($M>2$) case, and we give an example which shows that the condition is not sufficient even in the binary case. We show that, in likelihood space, the optimal demodulator decision regions are always bounded by hyperplanes, and we give some examples to illustrate the nature of these regions.

4.1. THE SYMMETRIC CUT-OFF RATE

We define <u>the symmetric cut-off rate</u>, $\tilde{R}_o$ to be the value of the righthand side of (4.1) when Q is the uniform distribution [$Q(m) = 1/M$ for $0 \leq m < M$] rather than the minimizing distribution. Thus,

$$\tilde{R}_o = \log_2 M - \log_2 \{\frac{1}{M} \sum_{j=0}^{J-1} [\sum_{m=0}^{M-1} \sqrt{P(j|m)}]^2\} \tag{4.3}$$

Evidently, $\tilde{R}_o \leq R_o$. Moreover, $\tilde{R}_o = R_o$ in the binary case (M=2) for which the uniform distribution is always the minimizing distribution, and also $\tilde{R}_o = R_o$ in most other cases of practical interest where the modulation signal set and the demodulator decision regions are reasonably "symmetric". Furthermore, the bound of (4.1) becomes

$$P_e < c_R L 2^{-N\tilde{R}_o}, \text{ if } \tilde{R} < R_o, \tag{4.4}$$

when the code is such that each letter in the code alphabet appears in the same fraction of codewords, a situation that always occurs in the conventional convolutional codes that would be used in practice. Thus, both to reflect this practical situation and to obviate the awkward minimization over Q in (4.1) we henceforth take $\tilde{R}_o$ of the resultant DMC as the measure of quality for the modulation system.

4.2 A NECESSARY CONDITION FOR OPTIMAL DEMODULATION

Henceforth, we assume that we have made the standard transformation [26] from waveforms to signal space so that s(t) and the "relevant" component of r(t) in Figure 1 may be replaced by the corresponding vectors $\underline{s}$ and $\underline{r}$ in n-dimensional Euclidean space.

We let $\underline{s}_m$ denote the transmitted signal when the modulation input is m. Any demodulator then may be viewed as a partition $\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{J-1}$ of m-space, where the "decision region" $\mathcal{D}_j$ is the set of all received vectors $\underline{r}$ that cause the demodulator to emit the decision j. We now derive a necessary condition for the decision regions to be optimal for a given signal set and channel.

Let $p(r|m)$ be the probability density function, which we assume to be everywhere continuous, for the received vector $\underline{r}$ given that signal $\underline{s}_m$ is transmitted over the channel. The transition probabilities of the resultant DMC, for a given demodulator, are then given by

$$P(j|m) = \int_{\mathcal{D}_j} p(\underline{r}|m)\, d\underline{r}. \qquad (4.5)$$

Let a and b, $a \neq b$, be two output letters of the demodulator such that $\mathcal{D}_a$ and $\mathcal{D}_b$ are adjacent regions, i.e. the boundary between $\mathcal{D}_a$ and $\mathcal{D}_b$ is hypersurface in n-space, and let $\underline{\rho}$ be any point on this boundary. Next, consider transferring from $\mathcal{D}_a$ to $\mathcal{D}_b$ a small region, $\mathcal{D}_{ab}$, which includes the point $\underline{\rho}$. [We show this situation in Figure 4.2 for the case n=2]. The resulting variation in the transition probabilities is then seen from (4.5) to be

$$\delta P(j|m) = \begin{cases} +\,p(\rho|m)\,\delta V & j=b \\ -\,p(\rho|m)\,\delta V & j=a \\ 0 & \text{otherwise} \end{cases} \qquad (4.6)$$

where we have now assumed that each $P(\underline{r}|m)$, $0 \leq m < M$, is continuous at $\underline{r}=\underline{\rho}$, and where

$$\delta V = \int_{\mathcal{D}_{ab}} d\underline{r}$$

is the volume of the small region $\mathcal{D}_{ab}$. If the decision regions are optimal, the resulting variation, $\delta\tilde{R}_o$, of the symmetric cut off rate, $\tilde{R}_o$, must be 0. As we see from (4.3) the condition $\delta\tilde{R}_o = 0$ is equivalent to the condition $\delta S=0$ where

$$S = \sum_{j=0}^{J-1} \left[\sum_{m=0}^{M-1} \sqrt{P(j|m)} \right]^2 . \qquad (4.7)$$

We then begin with

$$\delta S = \sum_{j=0}^{J-1} \sum_{m=0}^{M-1} \frac{\delta S}{\delta P(j|m)} \, \delta P(j|m)$$

which, with the aid of (4.6) becomes

$$\delta S = \sum_{m=0}^{M-1} \left[\frac{\delta S}{\delta P(b|m)} - \frac{\delta S}{\delta P(a|m)} \right] p(\rho|m)\,\delta V . \qquad (4.8)$$

We next note that direct differentiation in (4.7) gives

$$\frac{\delta S}{\delta P(j|m)} = \left[\sum_{i=0}^{M-1} \sqrt{P(j|i)} \right] \frac{1}{\sqrt{P(j|m)}} . \qquad (4.9)$$

provided that $P(j|m) \neq 0$. Then, by using (4.9) in (4.8) we obtain

$$\delta S = \sum_{m=0}^{M-1} [\frac{1}{\sqrt{P(b|m)}} \sum_{i=0}^{M-1} \sqrt{P(b|i)} - \frac{1}{\sqrt{P(a|m)}} \sum_{i-1}^{M-1} \sqrt{P(a|i)}] \cdot$$

$$\cdot P(\rho|m)\, \delta V$$

Thus, the condition that $\delta S = 0$ for an arbitrary δV becomes

$$\sum_{m=0}^{M-1} [\frac{1}{\sqrt{P(b|m)}} \sum_{i=0}^{M-1} \sqrt{P(b|i)} - \frac{1}{\sqrt{P(a|m)}} \sum_{i=0}^{M-1} \sqrt{P(a|i)}]\ p(\underline{\rho}|m) = 0. \tag{4.10}$$

We have thus proved:

<u>Theorem:</u> The demodulator decision regions $\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{J-1}$ in signal space can maximize R_o only if, for every a and b, $a \neq b$, such that $P(b|m) \neq 0$ and $P(a|m) \neq 0$ for $0 \leq m < M$, and such that $\mathcal{D}_a$ and $\mathcal{D}_b$ share a hypersurface boundary, it is the case that (10) holds at every point $\underline{r} = \underline{\rho}$ on this boundary which is a point of continuity of $P(\underline{r}|m)$ for $0 \leq m < M$.

In the next section we shall give a more illuminating form of condition (4.10).

4.3 DECISION REGIONS IN LIKELIHOOD SPACE

For the recieved vector $\underline{r}$, we define the waveform channel <u>likelihood-ratio vector</u>, $\underline{\Lambda}(\underline{r}) = [\Lambda_1(\underline{r}), \Lambda_2(\underline{r}), \ldots, \Lambda_{M-1}(\underline{r})]$, by

$$\underline{\Lambda}(\underline{r}) = \left[\frac{p(\underline{r}|1)}{p(\underline{r}|0)}, \frac{p(\underline{r}|2)}{p(\underline{r}|0)}, \ldots, \frac{p(\underline{r}|M-1)}{p(\underline{r}|0)} \right] \quad (4.11)$$

We note that, as pointed out by Massey [11], the demodulator can always, at its "front end", map $\underline{r}$ to $\underline{\Lambda}(\underline{r})$ with no loss of optimality. Thus, it becomes of interest to determine the form of the decision regions $D_0, D_1, \ldots, D_{M-1}$ in likelihood space which correspond to the optimal decision regions $\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{M-1}$ in signal space. But, seeing from (4.11) that (4.10) may, after division by $p(\underline{\rho}|0)$ (which we now assume to be non-zero) be rewritten as the linear equation

$$\sum_{m=1}^{M-1} \left[\frac{1}{\sqrt{P(b|m)}} \sum_{i=0}^{M-1} \sqrt{P(b|i)} - \frac{1}{\sqrt{P(a|m)}} \sum_{i=0}^{M-1} \sqrt{P(a|i)} \right] \Lambda_m(\underline{\rho}) +$$

$$+ \frac{1}{\sqrt{P(b|0)}} \sum_{i=0}^{M-1} \sqrt{P(b|i)} - \frac{1}{\sqrt{P(a|0)}} \sum_{i=0}^{M-1} \sqrt{P(a|i)} = 0, \quad (4.12)$$

we have immediately our main result

<u>Corollary 1</u>: The demodulator decision regions $\mathcal{D}_0, \mathcal{D}_1, \ldots, \mathcal{D}_{J-1}$ in signal space can maximize $\tilde{R}_o$ only if, for every a and b, $a \neq b$, such that $P(b|m) \neq 0$ and $P(a|m) \neq 0$ for $0 \leq m < M$ and such that $\mathcal{D}_a$ and $\mathcal{D}_b$, share a hypersurface boundary, if is the case that every point $\underline{r} = \underline{\rho}$ on this boundary, which is a point of continuity of $\underline{\Lambda}(\underline{r})$ lies on the hyperplane defined by (4.12)

In other words, the optimal decision regions in likelihood space are always bounded by hyperplanes. This fact has considerable practical significance as it is difficult to implement circuitry

which determines to what decision region some vector $\underline{\Lambda}(\underline{r})$ belongs except in the case when the decision regions are bounded by hyperplanes.

Condition (4.12) can be placed in an even more transparent form. We note that when (4.12) is satisfied, then

$$T_m = - \frac{\frac{1}{\sqrt{P(b|0)}} \sum_{i=0}^{M-1} \sqrt{P(b|i)} - \frac{1}{\sqrt{P(a|0)}} \sum_{i=0}^{M-1} \sqrt{P(a|i)}}{\frac{1}{\sqrt{P(b|m)}} \sum_{i=0}^{M-1} \sqrt{P(b|i)} - \frac{1}{\sqrt{P(a|m)}} \sum_{i=0}^{M-1} \sqrt{P(a|i)}} \tag{4.13}$$

(provided the denominator is non-zero) is just the intercept on the m-th axis in likelihood space of the boundary hyperplane. [See Figure 4.3 for a graphical interpretation of T_i.]. Thus, we have

<u>Corollary 2</u>: The demodulator decision regions $D_0, D_1, \ldots, D_{J-1}$ in likelihood space can maximize $\tilde{R}_o$ only if, for every a and b, $a \neq b$, such that $P(a|m) \neq 0$ and $P(b|m) \neq 0$ for $0 \leq m < M$, and such that D_a and D_b share a hypersurface boundary, it is the case that this boundary is a hyperplane whose intercepts with the coordinate axes are given by equation (4.13)

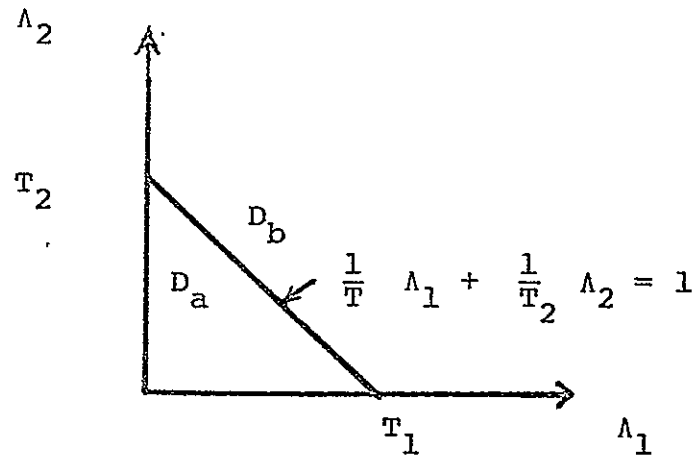

Figure 4.3. A hyperplane boundary in likelihood space separating the decision regions D_a and D_b for the case M=3.

Analogous to our definition of $\underline{\Lambda}(\underline{r})$ we now define the likelihood vector, $\underline{\lambda}(j) = [\lambda_1(j), \lambda_2(j), \ldots, \lambda_{M-1}(j)]$, of the DMC, which is created by the modulation system, by

$$\lambda_m(j) = \frac{P(j|m)}{P(j|0)} \tag{4.14}$$

where we assume $P(j|0) \neq 0$ for $0 \leq j < J$. From (4.12) (4.13) and (4.14) it follows after some tedious algebraic manipulation that, when the decision regions are optimal

$$\sum_{m=1}^{M-1} \frac{\sqrt{\lambda_m(b)\lambda_m(a)}}{T_m} = 1 \tag{4.15}$$

For the case of binary modulation (M=2), we note that (4.15) reduces to the necessary condition for optimality,

$$T = \sqrt{\lambda(b)\ \lambda(a)} \tag{4.16}$$

that was given by Massey [11].

4.4 EXAMPLES AND AN ITERATIVE OPTIMIZATION TECHNIQUE

In this section we give some examples to illustrate the use of the necessary condition for demodulator optimality given by Theorem 1 and its corollaries. We begin with a binary signaling example in which Massey's algorithm [11] based on equation (4.16) can be used conveniently to demonstrate effects of quantization. In the later examples of non-binary signaling cases, we also formulate a systematic method for finding the optimal demodualtor decision regions by iteration from an initial guess.

Example 4.1: Binary anti-podal signals in additive white Guassian noise (AWGN). In this case, the signal space may be taken as one-dimensional. The received signal r may be written

$$r = s + n$$

where $s_o = +\sqrt{E}$, $s_1 = \sqrt{E}$, E is the signal energy, and n is a zero-mean Gaussian random variable with variance $N_o/2$. N_o is the one-sided noise power spectral density. The likelihood ratio, $\Lambda(r) = p(r|0)/p(r|1)$, becomes

$$\Lambda(r) = e^{(4\sqrt{E}/N_o)r}$$

Because likelihood space is one-dimensional, the hyperplanes between decision regions are just points or "thresholds". Because $\Lambda(r)$ is monotonic in r, each such threshold T between decision regions in likelihood space can identified with the threshold

$$t = (N_o/4\sqrt{E}) \log_e(T)$$

between the corresponding decision regions in signal space. The demodulator can then be specified by the J-1 theresholds $t(1), t(2), \ldots, t(J-1)$ in the manner that the demodulator output is j when $t(j) < r \leq t(j+1)$ with the convention that $t(0) = -\infty$ and $t(J) = +\infty$. It truns out to be more convenient to use the normalized thresholds $t'(j) = (\sqrt{2/N_o})\, t(j)$ in order to weaken the dependence of the optimal thresholds on the energy-to-noise-power-spectral-density ratio.

In Table 4.1, we give the normalized threshold values $t'(1), t'(2), \ldots, t'(j-1)$ that maximize $\tilde{R}_o$ for the case J=3, J=4 and J=8 over a wide range of E/N_o ratios. These optimal thresholds were determined by Massey's iterative technique [11]. The resulting R_o for each case is given in Table 4.2 where we have included the value of R_o for an unquantized demodulator ($J=\infty$) to show the loss due to quantization. For comparison, we give also the value of $\tilde{R}_o$, acheived by using the heuristically-chosen "good" thresholds given by Jacobs [27]. For the case J=4, Jacobs normalized thresholds are $t'(3)=-t'(1)=1$ and $t'(2)=0$; while for J=8 they are $t'(7)=-t'(1) = 1.5$, $t'(6)=-t'(2)=1$, $t'(5)=-t'(3)=0.5$ and $t'(4)=0$. From Table 4.2, we see that the optimum thresholds offered scant superiority in $\tilde{R}_o$ over Jacobs' thresholds.

E/N_o (dB)	J=3	J=4		J=8 *			
	t'(2)=t'(1)	t'(3)=-t'(1)	t'(2)	t'(7)	t'(6)	t'(5)	t'(4)
-5.0	0.6247	0.9973	0.0	1.7636	1.057	0.504	0.0
-4.0	0.6280	1.0015	0.0	1.7678	1.060	0.505	0.0
-3.0	0,6322	1.006	0.0	1.773	1.062	0.506	0.0
-2.0	0.6374	1.013	0.0	1.779	1.065	0.507	0.0
-1.0	0.6440	1.021	0.0	1.788	1.070	0.509	0.0
+0.0	0.6523	1.032	0.0	1.799	1.075	0.511	0.0
1.0	0.6628	1.045	0.0	1.819	1.081	0.514	0.0
2.0	0.6760	1.061	0.0	1.829	1.090	0.518	0.0
3.0	0.6924	1.082	0.0	1.850	1.100	0.522	0.0
4.0	0.7130	1.108	0.0	1.876	1.113	0.528	0.0
5.0	0.7386	1.141	0.0	1.903	1.126	0.532	0.0

* t'(1) = -t'(7), t'(2) = -t'(6) and t"(3) = - t'(5)

Table 4.1 The normalized thresholds for J-ary demodulation, maximizing $\hat{R}_o$ for binary antipodal signals transmitted through the AWGN.

E_o/N_o (DB)	J=2	J=3	J=4		J=8		J=∞
	Optimal	Optimal	Optimal	Jacobs	Optimal	Jacobs	Unquantized
-5.0	0.13641	0.17195	0.18657	0.18657	0.20298	0.20298	0.21015
-4.0	0.16905	0.21251	0.23029	0.23029	0.25015	0.25015	0.25878
-3.0	0.20864	0.26133	0.28272	0.28272	0.30645	0.30645	0.31670
-2.0	0.25615	0.31929	0.34467	0.34466	0.37258	0.37258	0.38451
-1.0	0.31240	0.38692	0.41643	0.41640	0.44853	0.44853	0.46208
+0.0	0.37786	0.46396	0.49740	0.49730	0.53320	0.53320	0.54806
+1.0	0.45230	0.54899	0.58553	0.58533	0.62385	0.62385	0.63940
+2.0	0.53445	0.63894	0.67697	0.67659	0.71572	0.71572	0.73100
+3.0	0.62165	0.72888	0.76593	0.76528	0.80227	0.80227	0.81607
+4.0	0.70961	0.81238	0.84547	0.84449	0.87633	0.87633	0.88748
+5.0	0.79273	0.88287	0.90927	0.90799	0.93234	0.93234	0.94109

Table 4.2: Values of $\tilde{R}_o$ for the DMC obtained by optimally quantizing the output of AWGN channel for binary antipodal signals employing the thresholds listed in Table 4.1; values of $\tilde{R}_o$ obtained using Jacob's heuristically-chosen thresholds are listed for comparison.

Example 4.2: Hard-decision demodulation (i.e., J=M) for M-ary phase modulation in additive white Gaussian noise (AWGN). In this case, the signal space is 2-dimensional and the signal vector, $\underline{s}_m$ for $\leq 0 < m$ M, may be taken as the point on the circle of radius $\sqrt{E}$ (where E is the signal energy) at an angle of $(2\pi/M)m$. The ternary (i.e., M=3) case is shown in Figure 4. The heavy lines in this figure are the boundaries of the decision regions for

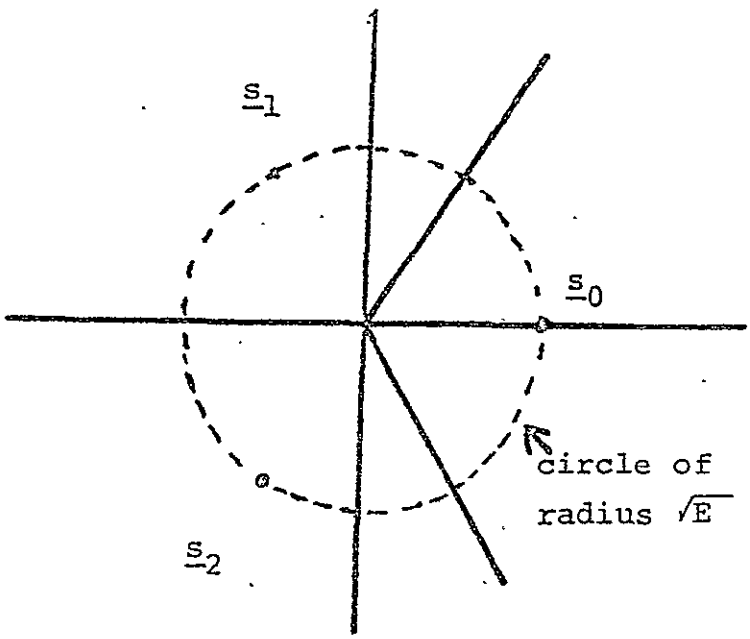

Figure 4.4: Maximum Likelihood Demodulation of ternary phase-modulated signals.

a maximum likelihood (ML) demodulator which, of course, is the hard-decision demodulator that minimizes errors probability when the signals are equally likely.

We now show that the ML demodulator for phase modulation is also the hard-decision demodulator which maximizes $\tilde{R}_o$. Let $\underline{s}_a$ and $\underline{s}_b$ be any two adjacent signals, i.e, their phase difference is $2\pi/M$. By the symmetry of the signal set and by the spherical symmetry of the additive white Gaussian noise, it follows that the ML demodulator causes the probabilties $P(b|0)$, $P(b|1), \ldots, P(b|M-1)$ to be a permutation of $P(a|0)$, $P(a|1)$, ..., $P(a|M-1)$, and also that for each m such that $P(b|m) \neq P(a|m)$ there is a corresponding m' such that $P(b|m) = P(a|m')$, $P(a|m) = P(b|m')$ and $p(\rho|m) = p(\rho|m')$ for ρ on the boundary between $\mathcal{D}_a$ and $\mathcal{D}_b$. Thus, the terms in the summation on the lefthand side of (4.10) either vanish singly [when $P(b|m) = P(a|m)$] or cancel in pairs. Thus, the ML decision regions satisfy the necessary condition for maximizing $\tilde{R}_o$ given by Theorem 1. Symmetry considerations indicate this is the only local maximum of $\tilde{R}_o$ and hence is the global maximum.

As a specific numerical example, we take the M=3 case of Figure 4.4 where $E/N_o = 1$, N_o being the one-sided noise power spectral density so that the variance of the noise in each dimension of signal space is $N_o/2$. The value of R_o yielded by the optimal hard-decision demodulator is 0.3971. The unquantized $\tilde{R}_o$ for this case can be found from Massey's results [11] to be 0.6254 so that the penalty for hard-decisions is 1.97 dB.

<u>Example 4.3</u>: Quaternary demodulation (J=4) for ternary phase modulation (M=3) in AWGN. Symmetry considerations suggest

that the optimal decision regions will consist of three regions, $\mathcal{D}_0$, $\mathcal{D}_1$, and $\mathcal{D}_2$, having 120° rotational symmetry and containing the signals $\underline{s}_0$, $\underline{s}_1$ and $\underline{s}_2$ respectively, together with an "erasure" region $\mathcal{D}_3$ containing the origin in signal space. In this case, there will be probabilities p and q such that $P(a|m) = q$ for all m, $P(j|m) = p$ for $j \neq m$ and $j \neq 3$, and $P(j|m) = 1-2p - q$ for $j = m$. Substituting these parameters into the necessary condition for optimality (4.13) we find that the resulting optimal intercepts correspond to the straight lines.

$$\Lambda_1 + \Lambda_2 = c$$

$$c\Lambda_1 - \Lambda_2 = 1$$

$$-\Lambda_1 + c\Lambda_2 = 1$$

where

$$c = 2\sqrt{p}/\sqrt{1-2p-q}$$

Thus, the optimal decision regions in likelihood space are known up to the parameter c. By trying various choices of c for the specific case $E/N_o = 1$ and calculating R_o for the DMC resulting from the demodulator corresponding to these decision regions, we find that, for the optimal decision regions, c= .486 and the attained value of R_o is 0.4402 which is a 0.45 dB improvement over hard decisions. In Figure 4.5 (a), we show the optimal decision regions in likelihood space, while in Figure 4.5 (b) we show the corresponding regions in signal space.

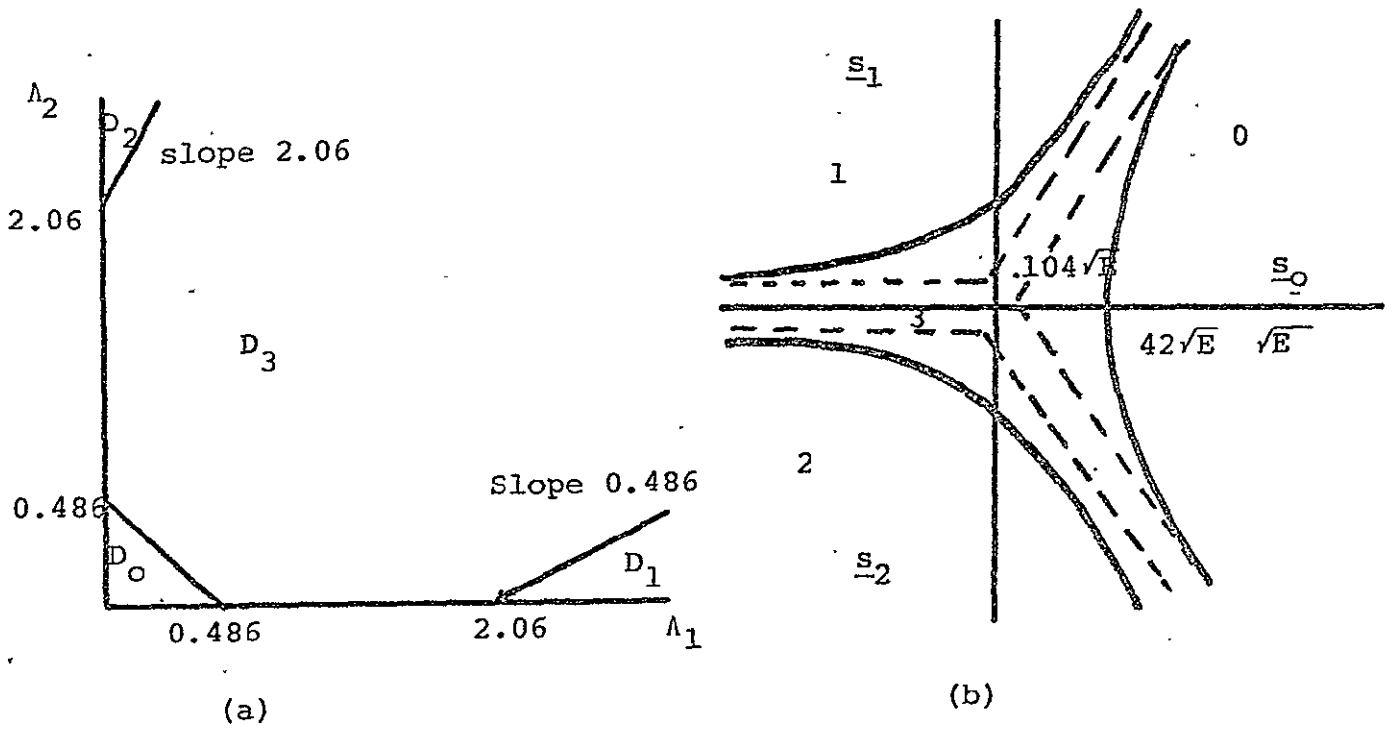

Figure 4.5: Optimal decision regions for quaternary demodulation of ternary phase modulation in AWGN with $E/N_o = 1$ shown (a) in likelihood space, and (b) in signal space.

We now describe a general iterative procedure that may be used to find the optimal decision regions for J-ary demodulator given a particular M-ary signal set and a given channel. The basic idea is quite simple. Given decision regions D_o, D_1, ..., D_{J-1} bounded by hyperplanes, we note that, for some $a \neq b$ such that D_a and D_b are adjacent regions, the intercepts of the bounding hyperplane with the coordinate axes in likelihood space will satify (4.13) if the decision regions are optimal. If they are not optimal, we can use the numbers determined by (4.13) as the intercepts of a hyperplane which will be a better approximation to the optimal bounding hyperplane between D_a and D_b. Our procedures may be stated as:

<u>Iterative Demodulator Optimization</u>:

<u>Step 0</u>: Make an initial guess, $D_0^{(1)}$, $D_1^{(1)}$, ..., $D_{J-1}^{(1)}$, for the optimal hyperplane-bounded decision regions in likelihood space. Set k=1.

<u>Step 1</u>: Calculate $P(j|m)$, $0 \leq j < J$ and $0 \leq m < M$, for the DMC created by the decision regions $D_o^{(k)}$, $D_1^{(k)}$, ..., $D_{J-1}^{(k)}$.

<u>Step 2</u>: Choose an a and b, $a \neq b$, such that $D_a^{(k)}$ and $D_b^{(k)}$ are adjacent and calculate $T_1^{(k+1)}$, $T_2^{(k+1)}$, ..., $T_{M-1}^{(k+1)}$ from equation (13). [Note: If the decision regions are optimal, then these T's will be the intercepts of the boundary between $D_a^{(k)}$ and $D_b^{(k)}$ with the coordinate axes.]

<u>Step 3</u>: Take the boundary between $D_a^{(k+1)}$ and $D_b^{(k+1)}$ as the hyperplane whose intercepts with the coordinate axes are $T_1^{(k+1)}$, $T_2^{(k+1)}$, ..., $T_{J-1}^{(k+1)}$.

Step 4: Repeat steps 2 and 3 until all such pairs a and b have been considered.

Step 5: If $D_0^{(k+1)}$, $D_1^{(k+1)}$ are "sufficiently close" to $D_0^{(k)}$, $D_1^{(k)}$, ...,$D_{J-1}^{(k)}$, stop and take the former decision regions as the result of this optimization method. Otherwise, increase k by 1 and return to step 1.

Two remarks about the above iterative method are in order. First, the most time-consuming part of the procedure is the calculation of the transition probabilities $P(j|m)$ in step 1. This would ordinarily be done by mapping from the decision regions $D_0^{(k)}$, $D_1^{(k)}$,...., $D_{J-1}^{(k)}$ in likelihood space to the corresponding decision regions $\mathcal{D}_0^{(k)}$, $\mathcal{D}_1^{(k)}$,..., $\mathcal{D}_{J-1}^{(k)}$ in signal space, then evaluating the intergral in (4.5) either analytically or numerically. Secondly, we note that when M=2, the above iterative procedure requires more calculation than the iterative method given by Massey [11] which is based on equation (4.16). Unfortunately, Massey's method does not generalize to cases where M>2.

We now give two examples to illustrate the use of the above iterative optimization method. The first of these is a "hard-decision" case which serves also to illustrate the fact that, when the signal set is not sufficiently symmetric, even in this case the decision regions which minimize demodulator error probability may not maximize $\tilde{R}_o$.

Example 4.4: Hard-decision demodulation for the two-dimensional signal set $\underline{s}_0 = [0,0]$, $\underline{s}_1 = [2,0]$ and $\underline{s}_2 = [0,2]$ in AWGN with variance $N_o/2 = 1/2$ in each component. [The average

signal-energy-to-noise-power-spectral-density-ratio, E/N_o, is 2.67 (or 4.26 dB).] In this case, the received vector $\underline{r} = [x,y]$ has a likelihood ratio vector

$$\Lambda(\underline{r}) = [\Lambda_1, \Lambda_2] = [\, e^{4x-4},\ e^{4y-4}].$$

Thus, the boundary straight-line (or hyperplane in two-dimensional space)

$$\frac{1}{T_1}\ \Lambda_1 + \frac{1}{T_2}\ \Lambda_2 = 1$$

corresponds to the curve

$$\frac{1}{T_1}\ e^{4x-4} + \frac{1}{T_2}\ e^{4y-4} = 1 \qquad (4.17)$$

in signal space.

As the boundaries between decision regions in the initialization step 0 of the algorithm, we choose those which minimize demodulator error probability, viz., the straight lines $\Lambda_1 = 1$, $\Lambda_1 = \Lambda_2$ shown in Figure 4.6(a). Note that the boundary between D_o and D_2 is a reflection around the line $\Lambda_1 = \Lambda_2$ of that between D_o and D_1. This symmetry is preserved at successive iteration of the algorithm because of the corresponding symmetry of the signal set about the line x=y and the symmetry of the AWGN. Thus, it suffices to determine the boundary between D_o and D_1. Letting (T_1, T_2) be the intercepts of this boundary with the Λ_1 and Λ_2 axes, respectively, and taking $(1, -10^6)$ as an approximation to

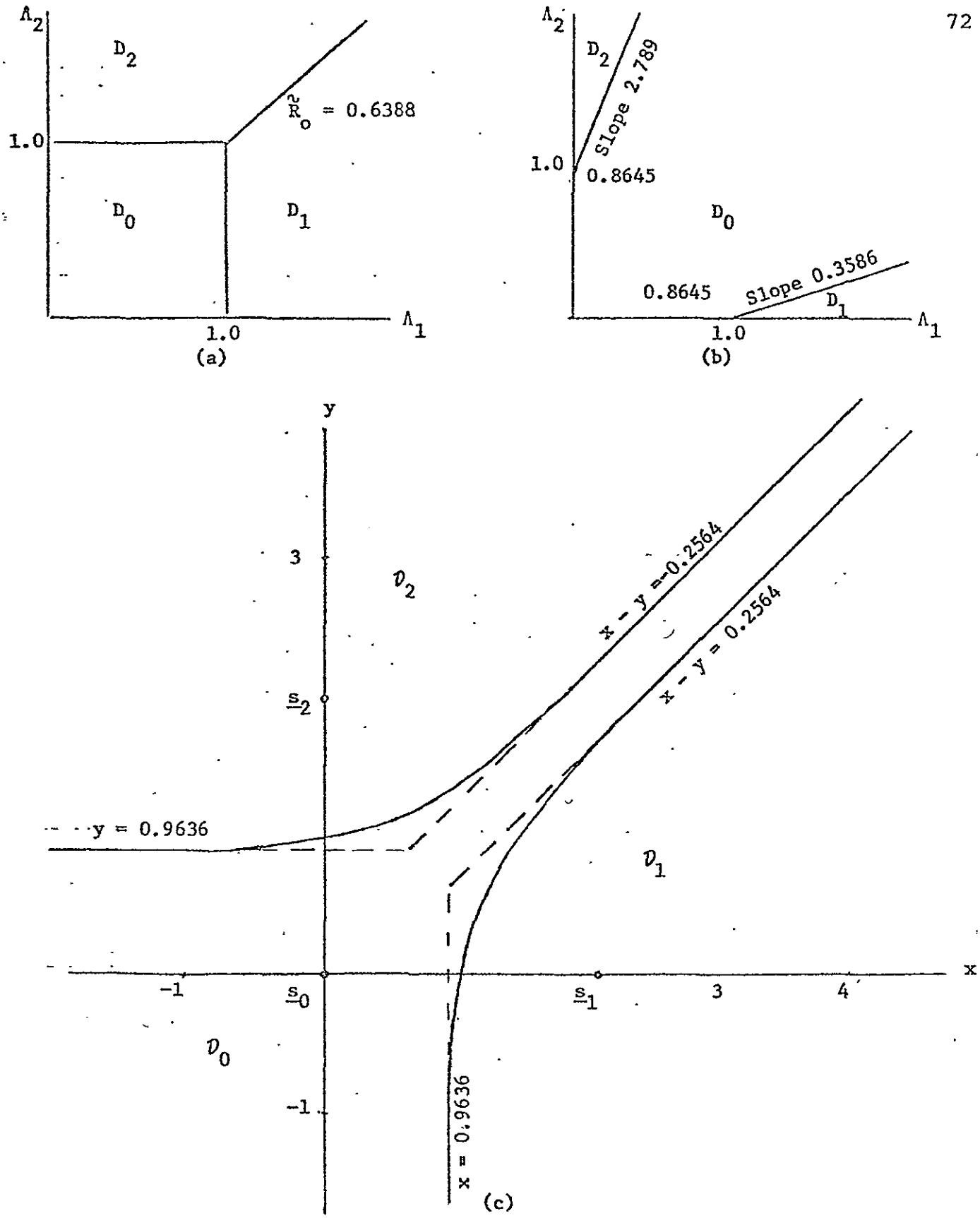

Figure 4.6: Hard decision demodulation of ternary signals:
(a) The initial decision regions in likelihood space,
(b) the optimal decision regions in likelihood space,
(c) the optimal decision regions in signal space.

the initial $(1,-\infty)$, we find, from application of the interative procedure in which we use (4.17) to determine the region over which the integral in (4.5) is to be evaluated numerically, the following succession of intercepts:

$(1, -10^6)$	initialization
(0.8307, -0.5732)	1st iteration
(0.8145, -0.4030)	2nd iteration
(0.8252, -0.3501)	3rd iteration
⋮	⋮
(0.8646, -0.3100)	15th iteration

up to the point where there is no further change in the 4th significant digit on further iteration. In Figures 4.6 (b) and 4.6 (c), we show the optimal decision regions for this example, as found by the iterative optimization procedure, in likelihood space and in signal space, respectively.

The values of $\hat{R}_o$ obtained as successive iterations were as follows:

0.6388	initialization
0.6462	1st iteration
0.6476	2nd iteration
0.6480	3rd iteration
⋮	⋮
0.6482	15th iteration

The optimal value of $\tilde{R}_o$ is about 1.5% (or .06 dB) above that obtained for the hard-decision demodulator which minimizes error probability. We see also that, for this example, the iterative procedure converged rapidly to the optimal demodulator--after only one iteration the resulting demodulator was effectively optimal.

As can be seen from Figure 4.6 (c), the optimal decision boundaries have asymptotes which are straight lines (hyperplanes in two-dimensional space). These asymptotes are shown by the dashed lines in the figure. If one uses these asymptotes as the boundaries of the decision regions for a sub-optimal demodulator, one finds the resultant R_o to be 0.6459 which is only .015 dB inferior to the optimal hard-decision demodulator. We shall later discuss the practical significance of the near-optimality of these asymptotic linear boundaries in signal space.

Example 4.5: Quaternary demodulation (J=4) for the same ternary signal set (M=3) and noise as in example 4.3.

In Figure 4.7 (a), we show the J=4 decision regions used to initiate the iterative optimization procedure. The optimal decision regions in likelihood space and in signal space are shown in Figures 4.7 (b) and 4.7 (c), respectively. Convergence to four significant digits of accuracy in the intercepts of the boundary lines with the coordinate axes required 25 iterations. The values of $\tilde{R}_o$ at successive steps were as follows:

0.6535	initialization
0.6986	1st iteration
0.7045	2nd iteration
0.7057	3rd iteration

⋮ ⋮

0.7062 25th iteration

We see that the decision regions after only two iterations were effectively optimal. The optimal value of R_o for quaternary demodulation is about 8% (or 0.34 dB) above that for optimal hard-decision demodulation.

In Figure 4.7 (c), the dashed lines (or hyperplanes in two-dimensional space) again are the asymptotes to the optimal decision boundaries in signal space. If these asymptotes are used as the actual boundaries betwen the decision regions in signal space, we find R_o of the resultant demodulator to be .7031 which is only .02 dB below optimal.

In Examples 4.3 and 4.5 we have seen that the linear (hyperplane)asymptotes to the optimal decision regions in signal space themselves bound the decision regions for a demodulator which is virtually optimal. The practical significance of this fact is that the resulting sub-optimal decision rule can be as easily implemented directly in signal space as can the optimal decision rule in likelihood space. There is no need for the conversion from signal space to likelihood space in order to obtain conveniently-implemented decision regions with linear (hyperplane) boundaries.

In fact, it can be shown generally, for AWGN in an n-dimensional signal space, that the optimal demodulation regions in signal space are such that each bounding hypersurface has (at most) 2^{n-1} hyperplane asymptotes. We conjecture that these hyperplane asymptotes form the boundaries of decision regions for a demodulator that is virtually optimum, and hence that the mapping from signal space to likelihood space is not necessary to obtain virtually optimal

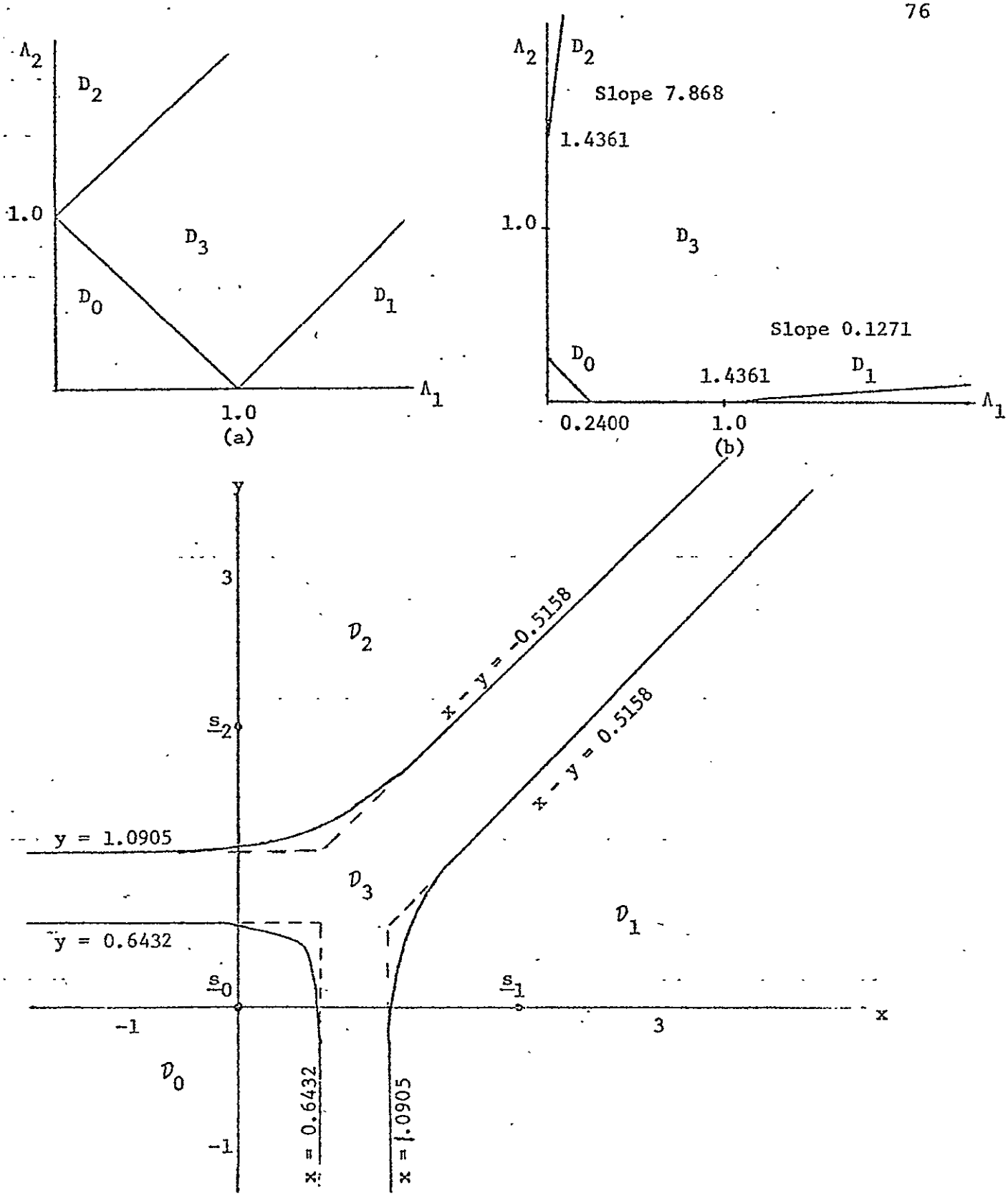

Figure 4.7: Quarternary demodulation of ternary signals:
(a) the initial decision regions likelihood space,
(b) the optimal decision regions in likelihood space,
(c) the optimal decision regions in signal space.

demodulation together with an easily-implemented decision rule.

4.5 A COUNTEREXAMPLE TO THE SUFFICIENCY OF OPTIMALITY CONDITION (4.10).

As we have pointed out above, condition (10) is actually the condition for an extremum of R_o, and hence not in general a sufficient condition for optimal demodulation. In the examples which we have studied wherein the noise was additive with a "smooth" density function, there has generally been only one set of decision regions satisfying (10) so that the extremum was necessarily the global maximum of $\tilde{R}_o$. We now give an example to show, however, that it is possible for (10) to be satisfied for decision regions that define only a local maximum, or even a local minimum, of $\tilde{R}_o$.

Example 4.6: Hard-decision demodulation for binary signals such that the conditional density functions for the likelihood ratio Λ are

$$p_0(\Lambda) = \begin{cases} 0.25 & 0 < \Lambda \leq 0.9 \\ 2.75 & 0.9 < \Lambda \leq 1.1 \\ 0.25 & 1.1 < \Lambda \leq 2.0 \\ 0 & 2.0 < \Lambda \end{cases}$$

and $p_1(\Lambda) = p_0(\Lambda)$ when the signals $\underline{s}_0$ and $\underline{s}_1$ are transmitted, respectively. [These are valid choices for these conditional density functions as they satisfy the constrains

$$\int_0^\infty p_0(\Lambda)\,d\Lambda = \int_0^\infty p_1(\Lambda)\,d\Lambda = 1$$

and $\Lambda = p_1(\Lambda)/p_0(\Lambda)$

that are the only ones that must be observed in the binary case].

For hard-decision demodulation with binary signalling, condition (4.10) reduces to Massey's condition (4.16) namely,

$$T = \sqrt{\lambda(1)\lambda(0)} \qquad (4.18)$$

Where T is the threshold between decision regions in likelihood space.

In Figure 4.8 (a), we show the conditional density functions for the likelihood ratio Λ, and in Figure 4.8 (b) we show $\sqrt{\lambda(1)\lambda(0)}$ as a function of the threshold T between the decision regions in likelihood space. We see that condition (4.18), the necessary condition for optimal demodulation, is satisfied at three places, viz., 0.56, 1.06, and 1.16; the corresponding values of $\tilde{R}_o$ are 0.0123, 0.0066, and 0.0069, respectivley. The third of these corresponds to a local, but not global, maximum of $\tilde{R}_o$. The second corresponds to a local minimum of $\tilde{R}_o$, while the first corresponds to the desired global maximum of $\tilde{R}_o$. That is, T=0.56, is the threshold between decision regions in likelihood space for the optimal demodulator.

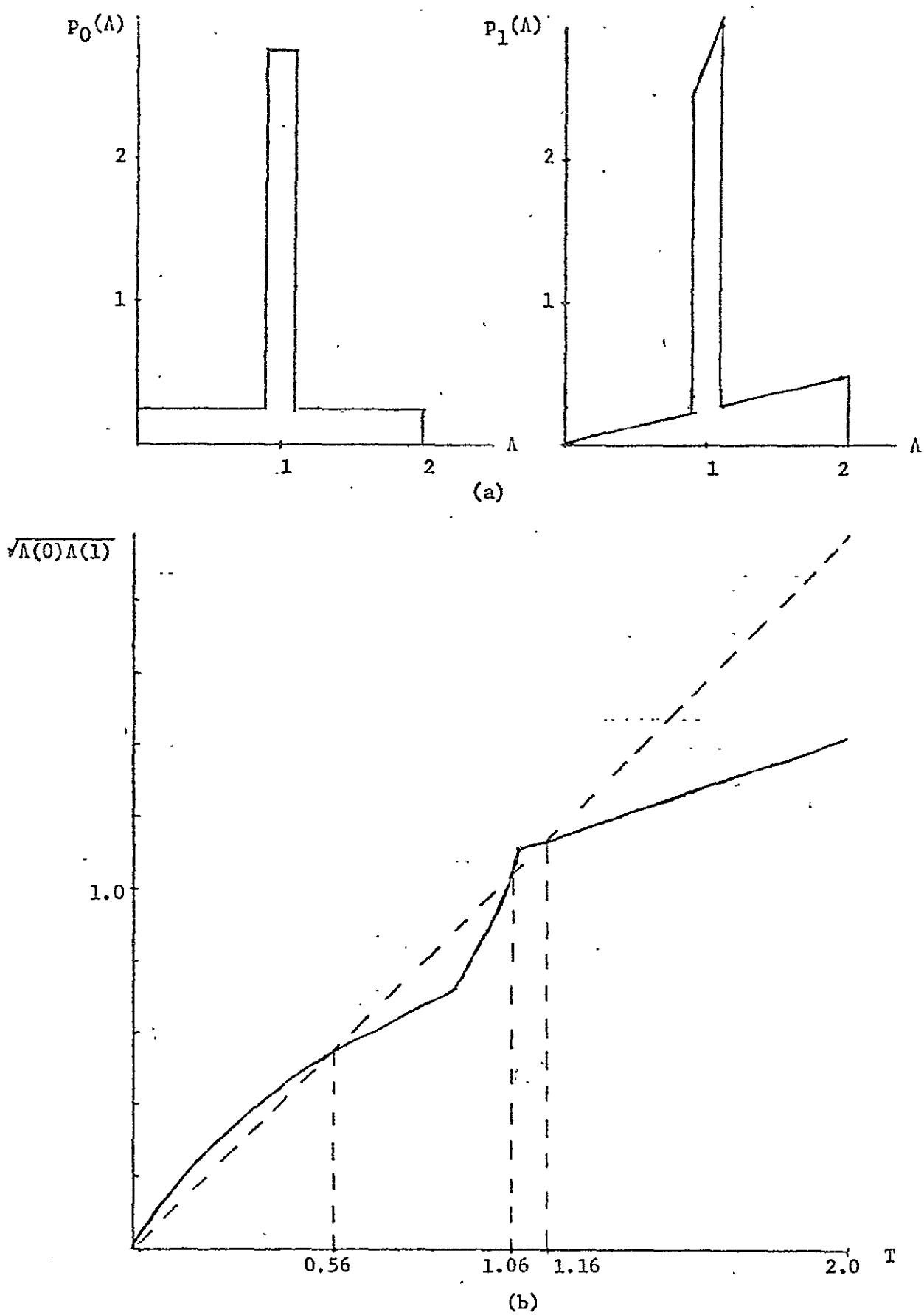

Figure 4.8: The binary, hard-decision, demodulation situation used to demonstrate the insufficiency of optimality condition (10).

4.6 SUMMARY

In this chapter, we have derived a necessary condition for optimal J-ary demodulation of M-ary signals, where optimality is taken to mean maximality of the symmetric cut-off rate, $\tilde{R}_o$ of the resulting discrete memoryless channel. By means of a counterexample, we have shown that this condition is not in general sufficient for optimality. We have also used this necessary condition for optimality as the foundation for an iterative optimization method to find the optimal demodulator decision regions from an initial "good guess".

In general, the optimal demodualtor decision regions are bounded by hyperplanes in likehood space. For the important case of additive white Gauusian noise, the corresponding optimal decision regions in signal space have hyperplane asymptotes. In some examples we have shown that the regions in signal space bounded by these asymptotic hyperplane define demodulator decision regions which are virtually optimal, and we conjectured that this happy state of affairs (which permits near optimal performance with a decision rule that can be simply-implemented directly in signal space) holds in general.

In regards to the application of this results to the problem of soft-decision decoding for the concatenated separate coding system, we note here that the super-channel of Figure 1.1, is again processed by a decoder; therefore, the criterion of optimality of maximizing $\tilde{R}_o$ of the resultant discrete channel is still valid. Unfortunately, since this super-channel is no longer memoryless, the analysis appears to be extremely difficult and we shall not pursue it further.

CHAPTER V

PERFORMANCE OF CONCATENATED CODING SYSTEMS ON A SIMULATED AWGN CHANNEL

In this chapter, the performance of several RS-convolutional concatenated coding systems utilizing the unit-memory codes found in Chapter II will be compared with similar systems utilizing the usual $(n_o,1)$ convolutional codes. The first or basic system is the hard-decision Viterbi decoded concatenated system with no feedback between the inner and outer decoders as proposed by Odenwalder [5]. In the second system studied, the Viterbi decoder is replaced by a RTMBEP decoder while the RS "errors-only"[2] decoder is replaced by an "erasures-and-errors" [2] decoder. In the third, fourth, and fifth systems, the inner decoder are restarted whenever the outer decoder corrects an error and feeds back the correction to the inner decoders. In the sixth and final system, we annex a tail to the unit-memory code (the code is, then, no longer of "true" unit memory) to enhance the capability of the Viterbi decoder to detect unreliable decoded symbols as was proposed by Zeoli [8] and Jelinek [9] for the $(n_o, 1)$ convolutional code. The various systems studied are summarized in Table 5.1. In each case, we choose the (18.6) unit-memory code as the inner code because it has practically minimum complexity in terms of decoder implementation and because of its reasonably large free distance (d_{free}=16). We choose the Reed-Solomon codes over $GF(2^6)$ with block length 63 symbols as the outer codes so that the symbol size of the RS code is matched to the byte-size of the

SYSTEM	Inner Decoder	Outer Decoder	Feed-back	Erasure	Inner Code Tail Annexa-tion
I	Viterbi	EO	NO	NO	NO
II	RTMBEP	EE	NO	YES	NO
II	Viterbi	EO	YES	NO	NO
IV	RTMBEP	EO	YES	NO	NO
V	RTMBEP	EE	YES	YES	NO
VI	Viterbi	EE	YES	YES	YES

* EO: Errors Only Decoders
EE: Erasures and Erros Decoders

Table 5.1: Various Block-Convolutional Concatenated Coding Systems Studied in Chapter 5.

information sequence of the unit-memory code. We also assume the interleaving to be "perfect". In other words, we assume that the decoded symbols at the output of the scrambler are statistically independent. With this assumption, we can calculate the error probability of the Reed-Solomon block by means of the statistics for the inner decoder which we obtain from simulations. Figure 5.1 shows an encoder and a Viterbi decoder for the unit-memory code interleaved to degree N. Note that the interleaving is done on a byte-by-byte basis, so that it "destroys information" to a much less extent than if it been done on a bit-by-bit basis. Further, we make use of the fact that almost all the incorrectly decoded Reed-Solomon codewords are d_{min} symbols away from the correct codewords to estimate the byte-error probability of the over-all system, where d_{min} is the minimum distance of the RS code. (We assume that the outer decoder always decodes, even when two codewords are equally likely.)

5.1. ODENWALDER'S CONCATENATED CODING SYSTEM AND SOFT-DECISION MODIFICATION WITH THE RTMBEP DECODING ALGORITHM.

The concatenated coding system proposed by Odenwalder [5] is shown in Figure 1.1. His original scheme employed a hard-decision Viterbi decoder as the inner convolutional decoder and a t-error correcting "errors-only" RS decoder as the outer block decoder.

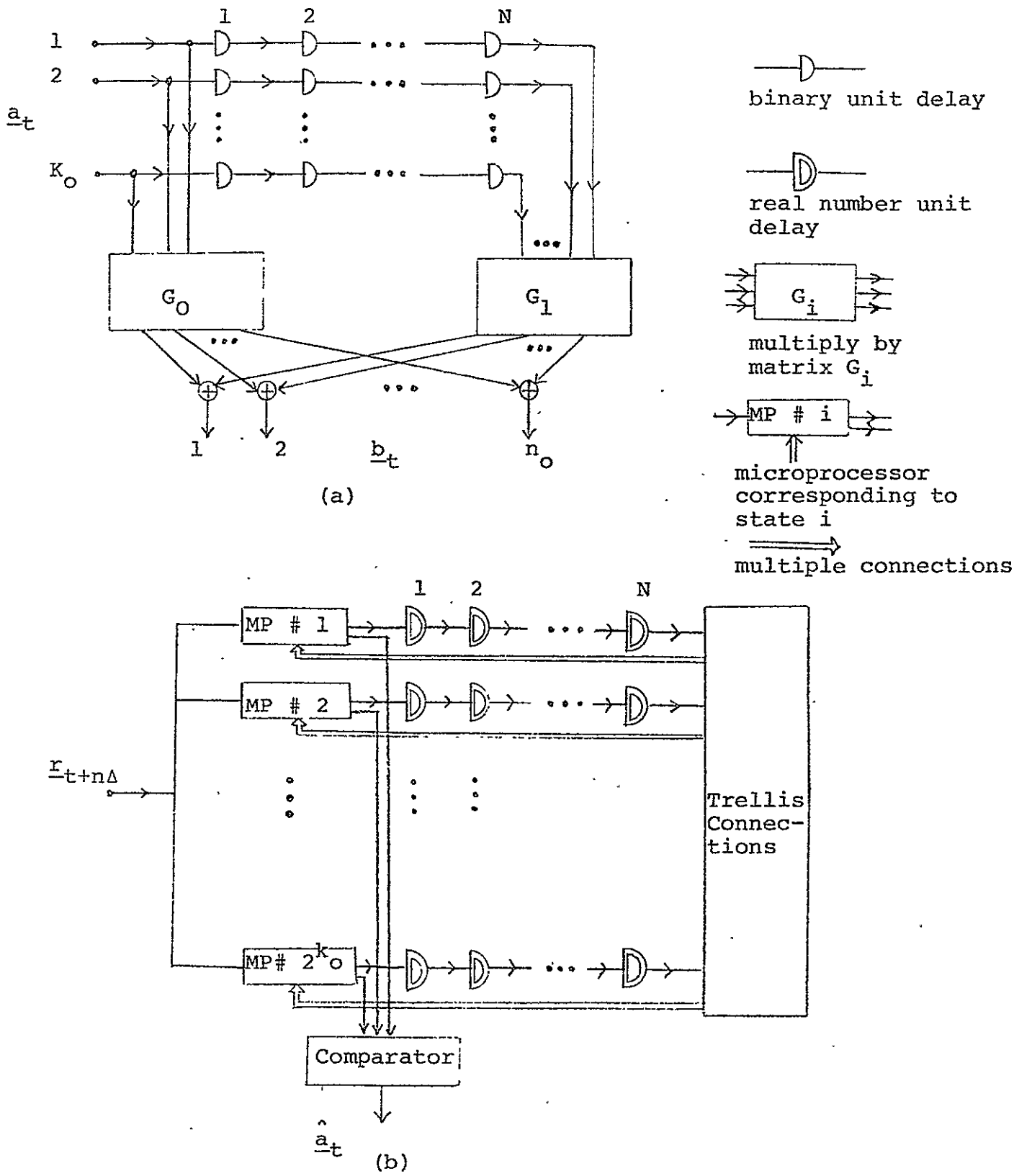

Figure 5.1 (a) A convolutional encoder, (b) a Viterbi decoder, for the (n_o, k_o) unit-memory convolutional code interleaved to degree N.

Then, the probability of error of a Reed-Solomon block is given by

$$P_{BERS} = \sum_{i=t+1}^{63} \binom{63}{i} p^i (1-p)^{63-i}, \qquad (5.1)$$

where p is the byte-error probability of the Viterbi decoder and $t = (d_{min} - 1)/2$. Then, the byte-error probability of the concatenated coding system assuming complete maximum likelihood (ML) decoding is given by

$$P_{BE} = \frac{2t+1}{63} P_{BERS} \qquad (5.2)$$

The byte-error-probability of the inner convolutional coding system at signal energy per information bit to one-sided noise-power-spectral-density ratios for the inner coding system, $(E_b/N_o)_I$, equal to 1.0 dB, 1.25 dB, 1.50 dB, and 1.75 dB are given in Table 2.3, which in turn correspond to signal-energy per inner channel digit to one-sided noise-power-spectral-density ratios E_s/N_o, of -3.77 dB, -3.52 dB, -3.27 dB, and -3.02 dB. This simulation result is used to determine the performance of the system. For $(E_b/N_0)_I$ = 1.25 dB, we show in Figure 5.2 the calculated error probability versus the <u>over-all signal-energy per information bit to one-sided noise-power-spectral-density</u>, which is given by

$$(E_b/N_o)_o = \frac{1}{R_{RS}} (E_b/N_o)_I = \frac{1}{R_{RS}R_{CON}} (E_s/N_o) \qquad (5.3)$$

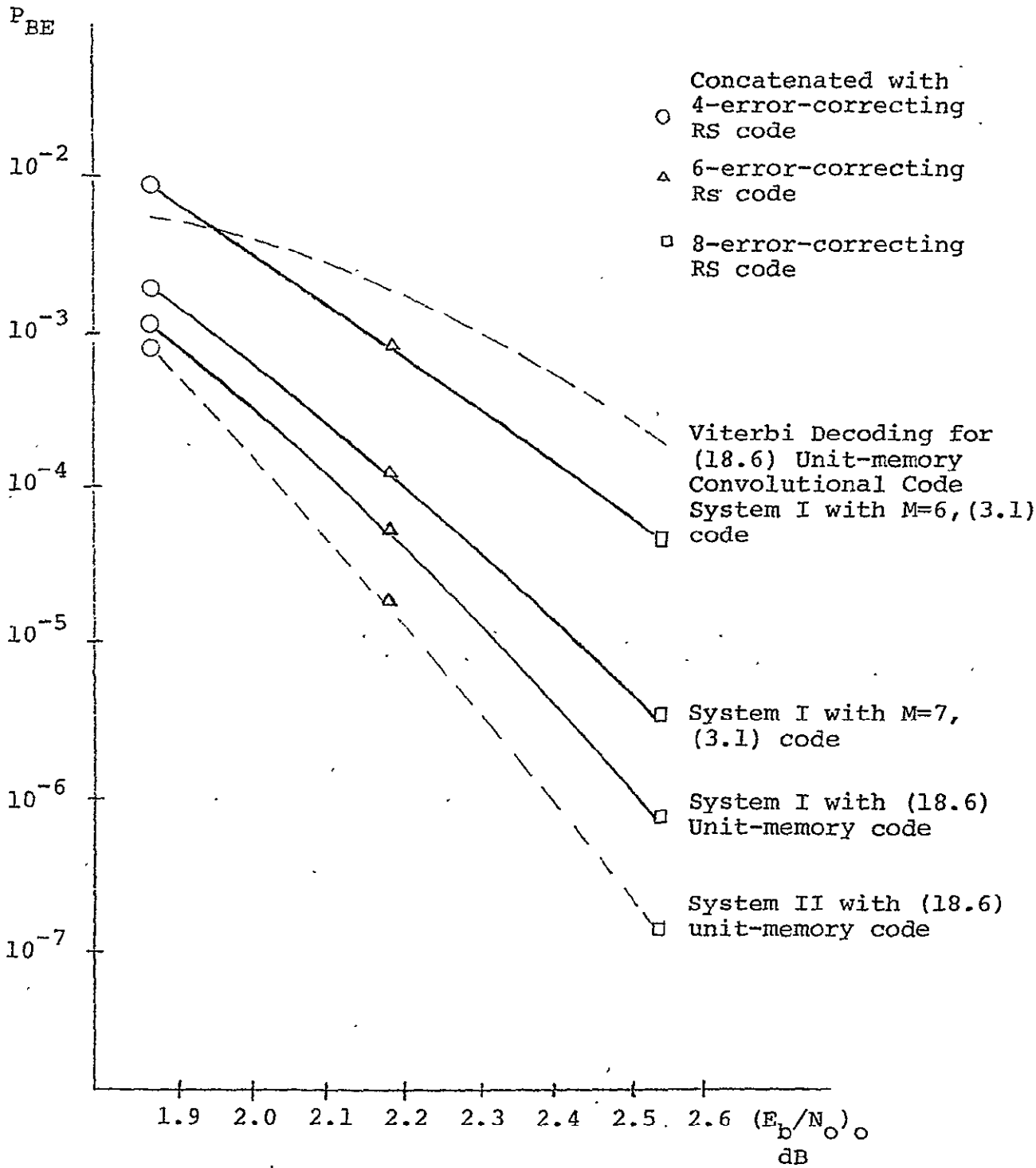

Figure 5.2: The Performance of Concatenated Coding Systems I and II with different RS code for a simulated AWGN channel with $E_s/N_o = -3.52$ dB, and the performance of Viterbi decoding for the (18.6) unit-memory code without concatenation.

C.2

where R_{RS} and R_{CON} are the rates of the RS code and the convolutional code, respectively.

Also plotted in Figure 5.2 are the performance curves at the same $(E_b/N_o)_I$ for similar concatenated coding systems using the M=6 (3.1) convolutional code and the M=7, (3.1) convolutional code as inner codes. The performance of the coding systems employing these three convoltional codes concatenated with a 6-error correcting Reed-Solomon code is shown in Figure 5.3. From both Figures, an approximatley 0.3 dB advantage in $(E_b/N_o)_o$ for the (18.6) unit-memory code, consistent with the result shown in Table 2.3 is observed. But the performance of concatenated systems is, in general very sensitive to $(E_b/N_o)_o$, this 0.3 dB advantage corresponds to a factor of 10 to 100 advantage in the error rate of the outer decoder, which, of course, should not be neglected. Even when we compare the performance of the unit-memory code to that of the M=7, (3.1) code which is of the same free distance as the unit-memory code approximately 0.1dB to 0.15 dB advantage can be observed due to the byte-oriented structure of the unit memory code.

We now replace the Viterbi decoder by a "Real-Time Minimal-Byte-Error Probability" decoder in the concatenated coding system. When the reliability function of the decoded symbol $P(\underline{a}_t/\underline{r}_{[1,t+\Delta]})$ is less than certain threshold, T, the decoder emits an erasure.

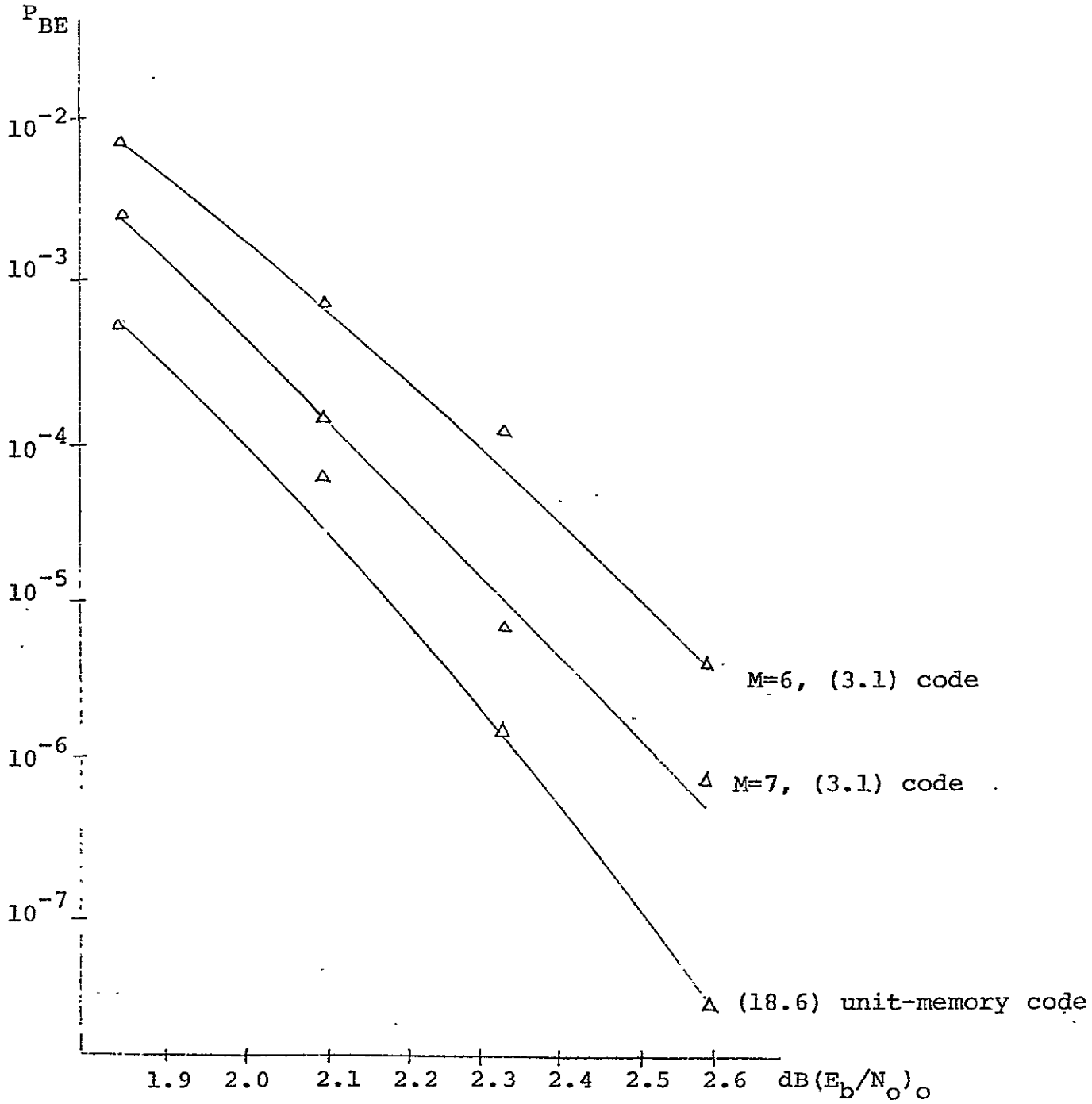

Figure 5.3: The performance of concatenated coding system I empolying 3 different convolutional codes and a 6-error correcting RS code with "errors-only" decoder for a simulated AWGN channel with $-3.77\ dB \leq E_s/N_o \leq -3.02\ dB$.

An erasures-and-errors Reed-Solomon decoder which can correct t errors and e erasured provided

$$2t+e<d$$

is used as the outer decoder. Then the probability of error of a Reed-Solomon block, assuming complete ML decoding, is given by

$$P_{BERS} = \sum_{d'=d_{min}}^{63} \sum_{\substack{t=0 \\ e=d'-2t}}^{d'/2} \binom{63}{t,e} p^t q^e (1-p-q)^{63-t-e} \qquad (5.4)$$

where p is again the probability of byte-error of the inner decoder and q is the probability of erasure of the inner decoder, and

$$\binom{N}{t,e} = \frac{N!}{t!e!(N-t-e)!}$$

Here again, we assume that the decoder always decodes and in case of a tie it always makes an error. Then, the byte-error probability of the overall system is again obtained from (5.2).

The byte-error-probability p and the erasure probability q depend on the particular threshold, T, selected. The optimal threshold is a function of E_b/N_o and the minimum distance, d_{min} of the Reed-Solomon code. Roughly speaking, for a given block length, when d_{min} gets larger, the over-all block error probability is minimized at a higher erasure rate. But there is no simple way to determine the optimal threshold analytically. We then find the p and q for T = 0.5, 0.7, and 0.8 by simulation and use these values of p and q to calcululate the byte-error probability of the coding system.

In Table 5.2, we show the result of this calculation. We see that for $(E_b/N_o)_I$ in the range between 1.25 dB and 1.75 dB, T = 0.7, is the best threshold among the three candidates. The performance of this concatenated coding system with T=0.7 is also plotted in Figure 5.2. The improvement of the performance due to the erasure scheme, as observed from Figure 5.2, is dependent on the error correcting capability of the outer coding system as well as $(E_b/N_o)_I$ and is approximately 0.1 dB. This slight improvement is probably not significant enough to justify the increased complexity of the convolutional decoder. However, as we shall see in the later sections, the RTMBEP decoder coupled with a "erasures-and-error" block decoder performs much better than the Viterbi decoder when feedback from the outer decoder is available.

$(E_b/N_o)_I$ (dB)	T	p	q	Over-all Symbol-Error Probability d=9	d=13	d=17
	0.80	0.01000	0.05000	0.735×10^{-2}	0.407×10^{-3}	0.877×10^{-5}
1.00	0.70	0.01325	0.04150	0.740×10^{-2}	0.477×10^{-3}	0.128×10^{-4}
	0.50	0.02100	0.01950	0.677×10^{-2}	0.555×10^{-3}	0.213×10^{-4}
	0.80	0.00675	0.03400	0.123×10^{-2}	0.244×10^{-4}	0.193×10^{-6}
1.25	0.70	0.00800	0.02650	0.902×10^{-3}	0.179×10^{-4}	0.149×10^{-6}
	0.50	0.01350	0.01125	0.107×10^{-2}	0.332×10^{-4}	0.481×10^{-6}
	0.80	0.00425	0.02125	0.112×10^{-3}	0.691×10^{-6}	0.173×10^{-8}
1.50	0.70	0.00525	0.01625	0.981×10^{-4}	0.684×10^{-6}	0.204×10^{-8}
	0.50	0.00900	0.00400	0.113×10^{-3}	0.136×10^{-5}	0.774×10^{-8}
	0.80	0.00250	0.01050	0.416×10^{-5}	0.636×10^{-8}	0.416×10^{-11}
1.75	0.70	0.00250	0.00825	0.249×10^{-5}	0.334×10^{-8}	0.196×10^{-11}
	0.50	0.00400	0.00250	0.336×10^{-5}	0.816×10^{-8}	0.927×10^{-11}

Table 5.2: The effect of erasing thresholds T on the symbol-error probability of the over-all concatenated coding systems for various $(E_b/N_o)_I$ with various outer codes.

5.2 FEEDBACK FROM THE OUTER DECODER TO THE INNER DECODER

Because of the nature of the convolutional code and the Viterbi decoding algorithm, once an "error event" occurs the decoder often makes a number of closely spaced erroneous estimations before it recovers to correct operation. Since the outer decoder of a concatenated coding system is designed in such a way that it is able to detect and correct almost all of the errors made by the inner decoder, it is then of significant advantage if the corrected estimation of the outer decoder is feedback to restart the inner decoder to avoid these "bursts" of errors. Figure 5.4 illustrates the general concept of such concatenated coding systems.

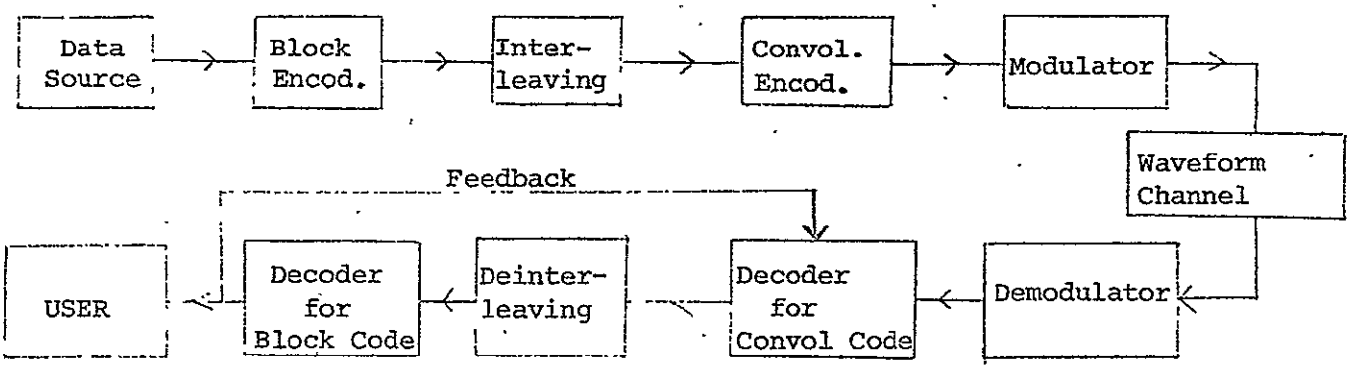

Figure 5.4. A typical Concatenated Coding System with Feeback from Outer Decoder to Inner Decoder

We have implemented a software Viterbi decoder and a software RTMBEP decoder which can be restarted with feedback. We carried out the simulations for this type concatenated coding system by assuming that the outer decoder always makes correct decisions. (This is a valid assumption since the probability of byte-error for the over-all system is negligible compared to that of the inner convolutional decoder.) We summarize the results of this simulation for the (18,6) unit-memory convolutional code at an $(E_b/N_o)_I$ of 1.25 dB in Table 5.3. We see that the RTMBEP decoder performs much better than the Viterbi decoder in this case. That is, the feedback from the outer decoder plays a much more important role in helping the RTMBEP decoder to recover from the error than it does for the Viterbi decoder. In Figure 5.5, we show the performance of these two systems concatenated with the "errors-only" block decoder. From the figure, we conclude that the improvement in performance due to the feedback is approximately 0.3 dB for the Viterbi decoder and about 0.5 dB for the RTMBEP decoder. Also listed in Table 5.3 is the error probability obtained for the M=7, (3,1) convolutional code (which has the same free distance as the (18,6) unit-memory code) with the same kind of Viterbi decoder with feedback. From Figure 5.5, we see this 0.1 dB difference in performance between the unit-memory code and the M=7, (3,1) code

	Byte-Error-Probability for (18,6) Unit-Memory Code	95% Confidence Interval	Byte-Error Prob. for M=7, (3,1) Code	95% Confidence Interval
Viterbi Decoding	0.0110	± 0.0023	0.01325	± 0.0025
RTMBEP Decoding	0.0075	± 0.0075		

Table 5.3: The Performance of Viterbi Decoder and RTMBEP Decoder for the (18,6) Unit-Memory Code and the M=7, (3,1) convolutional Code when Feedback from Outer Decoders is Employed for a Simulated AWGN Channel with $(E_b/N_o)_I$ at 1.25 dB and $\Delta=8$. The Data are Taken from a Sample of 8000 Bytes.

Erasure Criterion	p	q	Error-Prob. of over all system 4-error correcting	6-error correcting	8-error correcting
T=0.6	0.00388	0.01138	1.891×10^{-5}	6.387×10^{-8}	9.416×10^{-11}
T=0.7	0.00288	0.01763	2.498×10^{-5}	7.122×10^{-8}	8.207×10^{-11}
T=0.8	0.00163	0.02575	3.725×10^{-5}	8.138×10^{-8}	6.173×10^{-11}
T=0.9	0.00125	0.03413	1.178×10^{-4}	3.554×10^{-7}	3.283×10^{-10}

Table 5.4: The performance of over all concatenated coding system employing a RTMBEP decoder, which is restarted when feedback from the outer decoder is received, and a "erasures-and-errors" Reed-Solomon decoder for the (18,6) unit-memory code for a simulated AWGN channel with $(E_b/N_o)_I = 1.25$ dB.

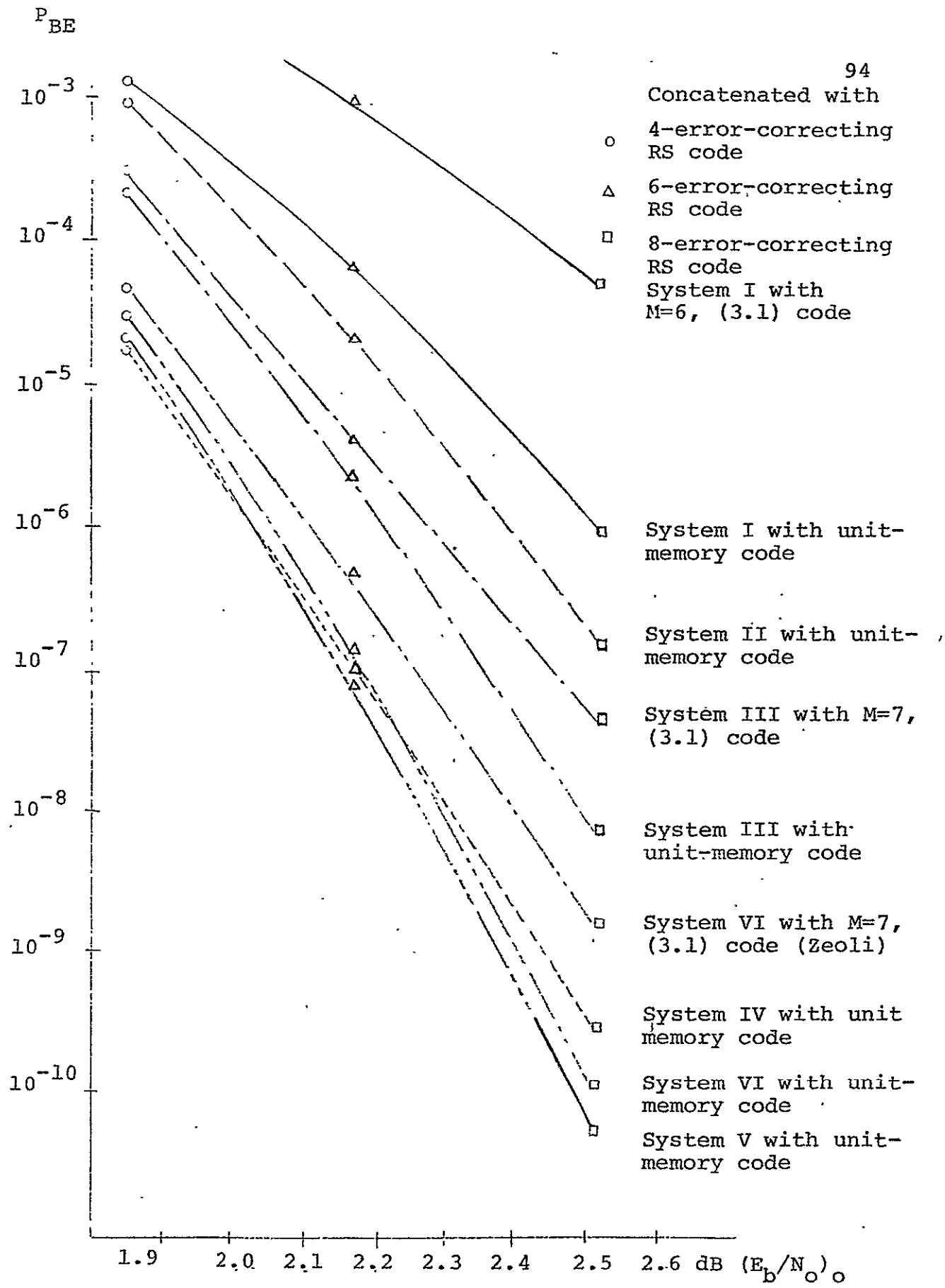

Figure 5.5: Performance of Concatenated Coding Systems Employing the Unit-Memory, (18.6) Convolutional are Compared to that of similar coding systems.

is consistent with the previous observation made about Figures 5.2 and 5.3. We can further make use of the reliability information put out by the RTMBEP decoder to erase some less reliable bytes decoded by the inner decoder as we did in last section. The byte-error-probability, p, erasure probability, q, and the over-all byte-error rate for the concatenated system are calculated and listed in Table 5.4 for thresholds equal to 0.6, 0.7, 0.8 and 0.9. The performance for T=0.8 is also plotted in Figure 5.5. Here, we observe nearly a 0.1 dB improvement, which is again consistent with the result of Section 5.1.

5.3 ZEOLI'S CONCATENATED CODING SYSTEM AND ZEOLI'S MODIFICATION ON THE UNIT-MEMORY CONVOLUTIONAL CODE.

In [8], Zeoli has proposed a concatenated coding system, which employs a rather long constraint length (K=32) convolutional code obtained by annexing a long tail to the M=7, (3,1) convolutional code. While the state complexity of the Viterbi decoder remains the same as that for the M=7, (3,1) code, the annexed tail has absolutely <u>no effect on the hard-decision decoding error-probability until after an error has been made</u>. But the tail provides excellent error-detection once the Viterbi decoder starts to make mistakes. Since the decoder constraint length is much smaller than that of the encoder, the encoded branches in the Viterbi decoder assume the previous several hard-decisions as part of the

encoder state (See [8]). The state metrics of the Viterbi decoder must become extremely ominous after a few branches once a decoding error occurs. We are able to make use of this phenomenon to implement excellent erasure rules for the inner decoder. However, once the decoder makes one mistake, the decoder assumes an incorrect encoder state which leads to endless errors. The feedback from the outer decoder is, therefore, a necessity in order to reset the decoder into correct state sequence and to terminate the error propagation.

Motivated by the success of Zeoli's concatenated coding system, we annexed the unit-memory convolutional code with a three-branch-long "random tail" such that the resultant code is truely an M = 4, (18,6) convolutional code. The encoding matrices of this convolutional code are shown in Table 5.5. The length of the tail is chosen to be compatible in memory to that required for Zeoli's M = 31, (3,1) code. Since the decoder is still a Viterbi decoder for the (18,6) unit-memory convolutional code, the decoder is going to make endless mistakes once an error occurs if the decision of the outer decoder is not fed back to the Viterbi decoder; it makes no difference whether the annexed M = 4, (18,6) code is non-catastrophic [12] or not. A software Viterbi decoder similar to the one described in last section is implemented to evaluate the performance of this system for the same AWGN channel. We chose the logarithm of the conditional probability $P(s_t|\underline{r}_{[1,t]})$ as the state metric for state s_t. The reliability information

is derived from the difference of state metrices between states $\Delta+1$ branches apart. The average of this quantity for correctly decoded state sequences as well as the standard deviation of this quantity may then be calculated. The erasure rule is to erase the decoded byte when the metric difference for $\Delta+1$ brances is larger than T standard deviations away from the average. Because of the law of large numbers, we expect to erase very few correctly decoded bytes but to erase most of the incorrectly decoded bytes. The resultant byte-error-probability, p, and erasure probability, q, and the calculated byte-error-probability for the over-all concatenated coding system for T=1.5, T=1.8 and T=2.0, are listed in Table 5.6. It is seen that the system performs best when the threshold is set at T=1.80. If the metrics were Gaussian, as the central limit theorem would suggest since the number of digit metrics added is large, this threshold of 1.80 correspond to an erasure probability of 3.6% for correct bytes; we see actually that the fraction erased correct bytes is 2%, somewhat less than Gaussian estimate. We also include the performance of this system with T=1.8 in Figure 5.5. The performance of Zeoli's originally proposed system is directly taken from [8]. We observe about 0.2 dB improvement over Viterbi decoding with feedback for the unit-memory code in the last section compared to the same system with Zeoli's modification. This improvement is again consistent with that observed from the Viterbi decoding with feedback for the M=7, (3,1) code compared to Zeoli's original system which makes us confident of our use of Zeoli's

$$G_0 = \begin{matrix} 111000 & 110100 & 110000 \\ 011100 & 011010 & 011000 \\ 001110 & 001101 & 001100 \\ 000111 & 100110 & 000110 \\ 100011 & 010011 & 000011 \\ 110001 & 101001 & 100001 \end{matrix} \qquad G_1 = \begin{matrix} 000011 & 000111 & 001011 \\ 000110 & 001110 & 010110 \\ 001100 & 011100 & 101100 \\ 011000 & 111000 & 011001 \\ 110000 & 110001 & 110010 \\ 100001 & 100011 & 100101 \end{matrix}$$

$$G_2 = \begin{matrix} 000110 & 000001 & 101111 \\ 100011 & 000011 & 010011 \\ 110001 & 100110 & 100001 \\ 111000 & 110101 & 001000 \\ 011000 & 011010 & 011100 \\ 001100 & 011100 & 110110 \end{matrix} \qquad G_3 = \begin{matrix} 011000 & 111001 & 011000 \\ 110001 & 110010 & 110000 \\ 100011 & 100101 & 100001 \\ 000111 & 000011 & 001011 \\ 000110 & 001110 & 010110 \\ 001100 & 011100 & 101100 \end{matrix}$$

$$G_4 = \begin{matrix} 111111 & 010100 & 000000 \\ 000111 & 111010 & 100000 \\ 000000 & 111111 & 010100 \\ 100000 & 000111 & 111010 \\ 010100 & 000000 & 111111 \\ 111010 & 100000 & 000111 \end{matrix}$$

Table 5.5: The encoding matrices of a M=4, (18,6) convolutional code obtained by annexing a random tail to the (18,6) unit-memory convolutional code

			Byte-Error-Probability of Over-all Concatenated Coding System		
T	p	q	4-error correcting	6-error correcting	8-error correcting
1.50	0.00125	0.03788	2.095×10^{-4}	8.175×10^{-7}	9.465×10^{-10}
1.80	0.00263	0.02088	3.602×10^{-5}	1.074×10^{-7}	1.245×10^{-10}
2.00	0.00425	0.01450	4.168×10^{-5}	1.899×10^{-7}	3.708×10^{-10}

Table 5.6: The performance of Zeoli's type of concatenated coding system employing the M=4, (18,6) convolutional code and a Viterbi decoder with feedback for the (18,6) unit-memory code for different erasing thresholds for a simulated AWGN channel at $(E_b/N_o)_o = 1.25$ dB. The data are taken from a sample of 8000 bytes, and the decoding delay for the Viterbi decoder is 8.

data without checking it by our own simulations.

We also point out that our use of the "normalized" thresholds in terms of mean and standard deviations of the increment of branch metric is much more convenient and makes much more sense as an erasure criterion than Zeoli's choice of the state metric itself, since our choice of the thresholds are independent of the particular metric which the Viterbi decoder employs.

5.4 DEGRADATION OF PERFORMANCE FOR EMPLOYING HIGHER RATE INNER CODES.

We have extensively studied block-convolutional concatenated coding systems employing rate 1/3 convolutional codes and Reed-Solomon codes over GF (2^6). However, it is sometimes desired in practice to operate the inner convolutional codes at a higher rate, rate 1/2 in particular, in order to ease the burden imposed on the phase locked loops in the receiver. We shall illustrate a heuristic approach to estimate the performance of similar concatenated coding systems with rate 1/2 coding systems.

From past experience, the performance of rate 1/2 convolutional coding system is about 0.5 dB inferior to that of rate 1/3 convolutional coding system of the same complexity. We therefore tested the performance of a software Viterbi decoder (without feedback) for the M=6, (2.1) convolutional code in an AWGN channel at $(E_b/N_o)_I = 1.75$ dB or, equivalently, $E_s/N_o = -1.25$ dB. The results of this simulation and the calculated overall byte-error-probability when it is concatenated with the Reed-Solomon codes are listed in Table 5.7. Comparing the results of Table

Byte-Error-Prob. of the Viterbi Decoder	Byte-Error-Probability of the Over-all Concatenated Coding System			
	4-error-correcting	6-error-correcting	8-error-correcting	10-error-correcting
0.0305	6.168×10^{-3}	6.287×10^{-3}	3.285×10^{-5}	1.006×10^{-6}

Table 5.7: The Performance of Viterbi Decoder for a M=6, (2,1) Convolutional Code in a Simulated AWGN Channel at $(E_b/N_o)_I$ = 1.75 dB. The data are taken from a sample of 400 bytes and the decoding delay is 48 branches.

5.7 To those given in Table 2.3 for M=6, (3.1) code and the (18,6) unit-memory code, we find that the performance of rate 1/2 code operated at $(E_b/N_o)_I$ = 1.75 dB is almost equivalent to that of the unit memory code operated at $(E_b/N_o)_I$ = 1.0 dB or that of the M=6, (3,1) code operated at $(E_b/N_o)_I$ equal to somewhere above 1.25 dB. This is consistent with the past experience. The performance of the concatenated coding systems using the three codes are plotted in Figure 5.6. Once again the difference between the M=6, (3,1) coding system and the M=6, (2.1) is about 0.5 dB. This is obvious because for the two coding systems to have the same error probability, it is necessary that the two inner decoders have exactly the same error-probability.

Unfortunately, the (12,6) unit-memory convolutional code has exactly the same free distance as the M=6, (2,1) code (See Table 2.1). Therefore, there is no distance advantage in

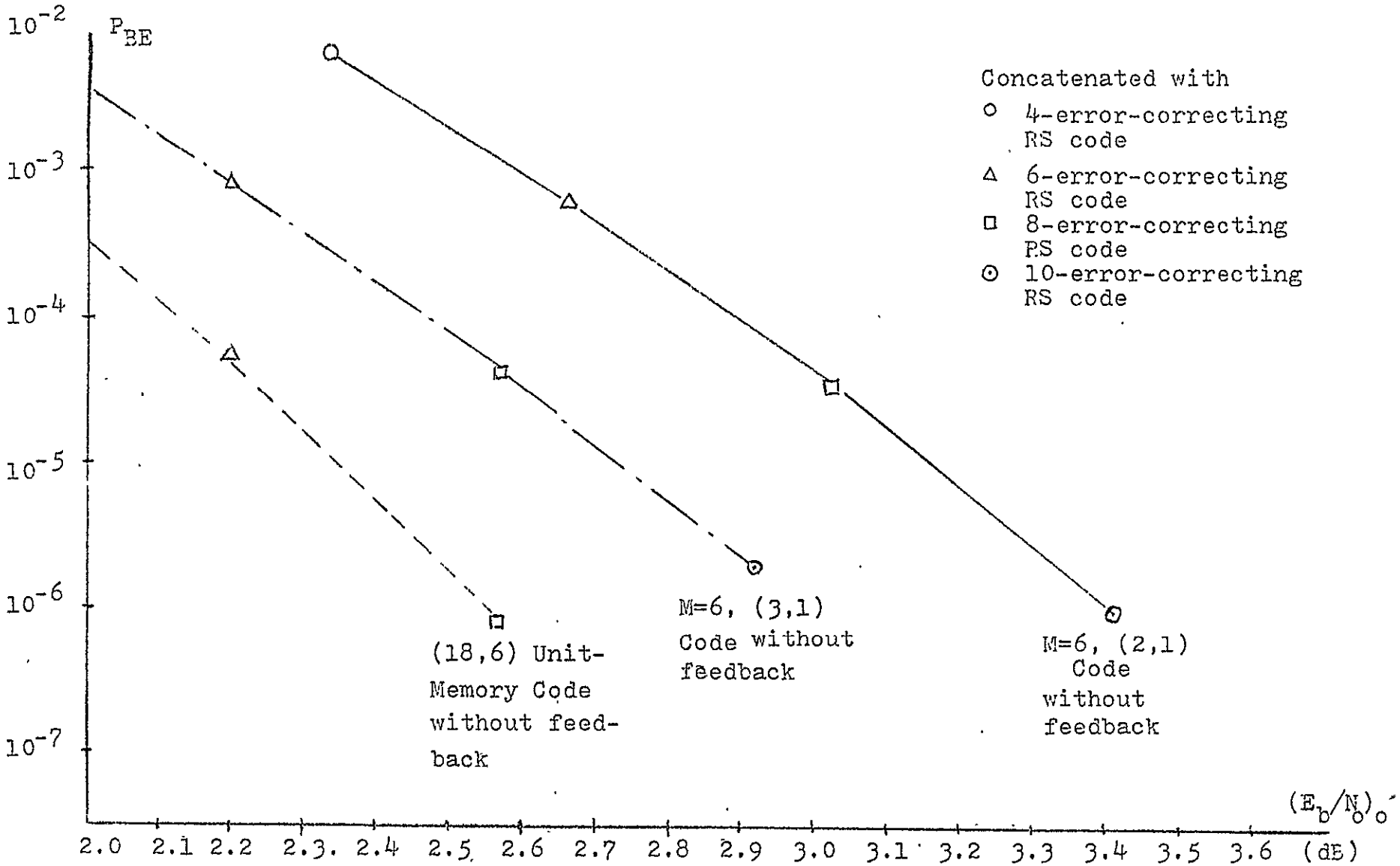

Figure 5.6: Performance of the concatenated system I described in section 5.1. The rate 1/2 system is operated at $(E_b/N_o)_I = 1.75$ dB, whereas the rate 1/3 systems are operated at $(E_b/N_o)_I = 1.25$ dB.

utilizing the unit-memory convolutional code.

In conclusion, a penalty of approximately 0.5 dB is paid when a (3,1) convolutional code is replaced by a (2,1) convolutional code of the same state-complexity; and a unit-memory code which has a free distance one greater than an (n_o, 1) convolutional code of the same state-complexity has 0.3 dB advantage compared to this (n_o, 1) convolutional code. In the case we studied, a sum of about 0.8 dB is sacrified when we choose a rate 1/2 convolutional code to replace the rate 1/3 unit-memory code. In the case where the byte-size is 5 bits, from Table 2.1, we see the free distances of (15,5) and (10,5) unit-memory codes are 15 and 9 which are 2 greater and 1 greater than the (3,1) and (2,1) codes of the same complexity; we expect again an approximately 0.8 dB loss when we choose the (10,5) code in place of the (15,5) unit memory code. But, when the byte-size is 4, on the other hand, we see that there is no distance advantage for the (12,4) unit-memory code compared to the (3,1) code; while, on the contrary, the free distance of the (8,4) unit-memory code is one greater than the (2,1) code; therefore the 0.5 dB loss due to the increase of rate should be somewhat compensated by the better free distance of the (8,4) unit-memory convolutional code. Therefore, we might expect only a 0.2 dB degradation.

5.5. CONFIDENCE INTERVALS FOR THE SIMULATION RESULTS

The performance of various block-convolutional concatenated coding systems over a simulated AWGN channel was studied and compared in the previous sections. The fact that the overall byte-error rate is calculated from the byte-error rate of the inner decoders makes it possible to carry out the simulations with a moderate size of samples. Assuming that the decoder makes a error with probability P_{BE} for each byte-decision, the number of byte errors within n byte-decisions is a binomial random variable with parameters n and P_{BE}. Then, the mean value of this random variable, is, nP_{BE}, and the standard deviation is $\sqrt{nP_{BE}(1-P_{BE})}$. If n is sufficiently large, this binomial random variable can be approximated by a Gaussian random variable with the same mean and variance. since 95% of the samples of a Gaussian random variable are within the interval specified by the mean and twice of the standard deviation, we are confident that with more than 95% probability, the actual byte-error rate for the inner decoder is in the interval specified by $(P_{BE} - 2\sqrt{nP_{BE}(1-P_{BE})}, P_{BE} + 2\sqrt{nP_{BE}(1-P_{BE})})$. These 95% confidence intervals are indicated in Table 2.3.

The performances of System I using the M=6, (3.1) code and M=1, (18.6) unit-memory code corresponding the upper and lower limits of these intervals are calculated and shown in Figure 5.7. We conclude with 95% confidence that the actual performance of the concatenated coding system deviates no more than 0.1 dB from our simulation results Moreover, since all the simulation results are obtained through the same pseudo-random number sequence, the relative difference in performance among various systems are, in fact, much more acturate than the 0.1 dB confidence interval observed here.

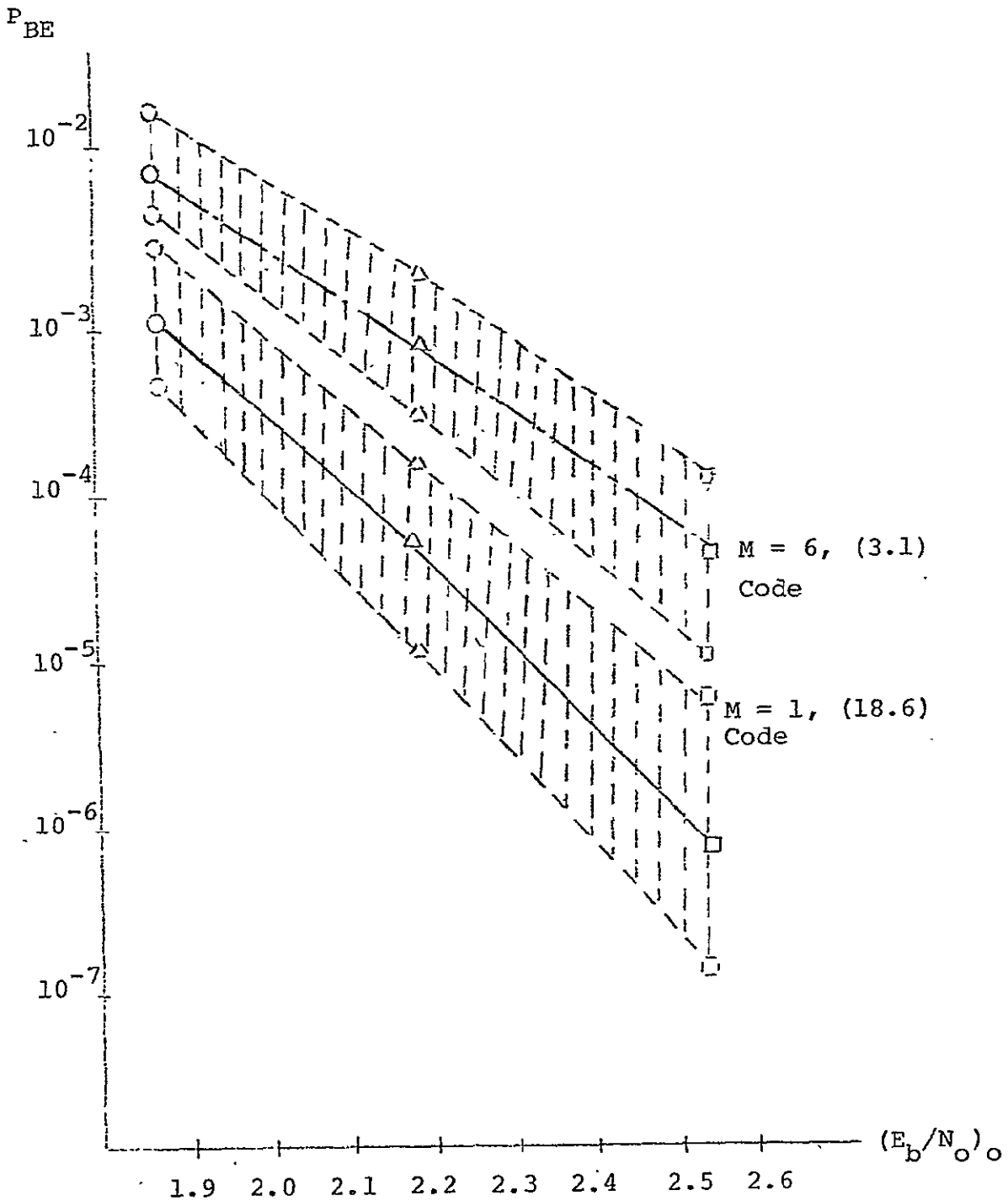

Figure 5.7. The 95% Confidence Intervals for the Performance of System I with M=6, (3.1) Convolutional Code and (18.6) Unit-Memory Code.

CHAPTER VI

SUMMARY AND CONCLUSIONS

In Chapter V, we have extensively studied block-convolutional concatenated coding systems with various modifications. It appears that the advantage of employing unit-memory convolutional codes can improve the performance nearly 0.3 dB. The feedback from the outer decoder to restart the Viterbi decoder also contributes about 0.3 dB. But, surprisingly, the feedback from the outer decoder to restart the RTMBEP decoder offers approximately 0.5 dB advantage, which is 0.2 dB more than the same feedback is able to help the Viterbi decoder. As a result, this might be the principal occasion where the use of RTMBEP decoding is justified. Another unexpected result is that soft-decisions by the inner decoder in conjunction with an erasures-and-errors outer decoder only improves the overall performance by about 0.05 dB to 0.1 dB for RTMBEP decoding. Even with Zeoli's modification which provides the best error detection capability, soft-decisions in conjunction with an erasures-and-errors outer decoder can improve the performance by only approximately 0.2 dB. We summarize the effects of each feature discussed above on the performance of the block-convolutional concatenated coding system in Figure 6.1. The figure is drawn in terms of a dB scale. As a communications engineer starts to choose a coding system, the first question he faces is whether his phase-locked-loop can tolerate the burden of a rate 1/3 coding system, if the

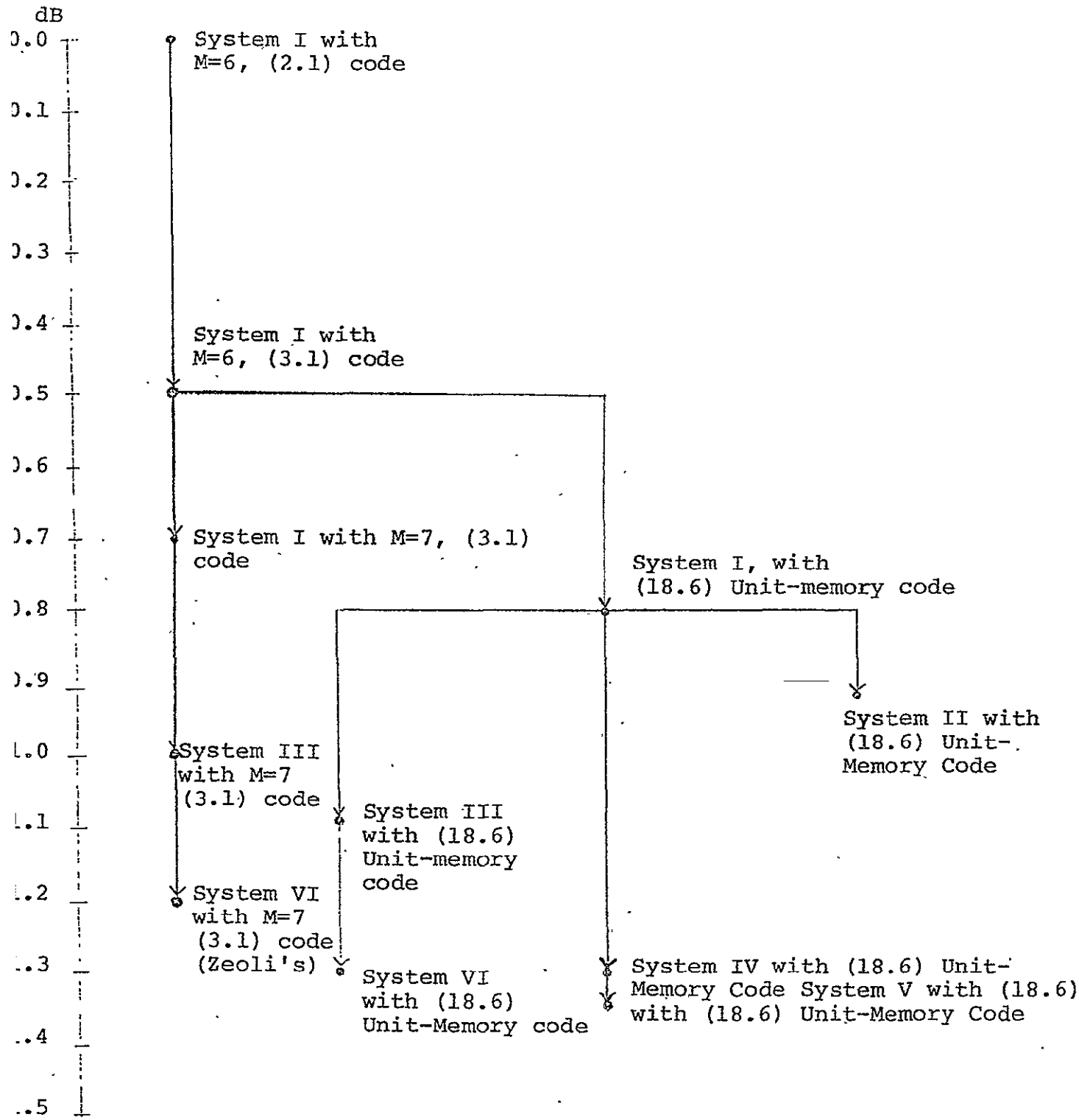

Figure 6.1. The relative dB gains among the concatenated Coding Systems studied.

answer is positive, he gains 0.5 dB. Then, he is to decide which inner code to employ, to choose the M=7, (3,1) code gives 0.2 dB advantage over the M=6, (3,1) code but twice the number of states in the decoder is required; whereas to choose the M=1, (18,6) code has 0.3 dB advantage with the number of states required kept the same, but more branch connections are required. The third question is whether he allows the decisions of the outer decoder to feedback to the inner decoder; if not the obvious choice is Viterbi decoding; otherwise, he can gain 0.3 dB or 0.5 dB depends on whether the Viterbi decoder or the RTMBEP decoder is utilized. And finally, if soft-decisions are desired, he can gain 0.2 dB through Zeoli's type of erasure scheme if he use a Viterbi decoder, or gains only 0.05 dB if the plain RTMBEP erasure scheme is employed.

As we observed from Figure 5.5, the leading contenders for a good concatenated system are the sophisticated schemes of (1) Zeoli's modification with the unit-memory code, (2) hard-decision or (3) the soft-decision RTMBEP decoding with feedback from the outer decoder for the unit-memory code. Among them, the soft-decision RTMBEP decoder with feedback performs the best. In terms of hardware implementation, Zeoli's modification with the unit-memory code and the hard-decision RTMBEP decoder are approximately of the same

complexity. However, since the operation of the Viterbi decoder for the Zeoli's system depends on the correct feedback from the outer decoder, there is always a slim chance that the outer decoder fails to provide correct decisions to the Viterbi decoder. Since the encoder constraint length is much larger than the decoder constraint length, this can cause endless errors as if a catastrophic convolutional code were used. Thus, it is necessary to send synchronization signals periodically to reset the Viterbi decoder to guarantee restoration of normal operation. The RTMBEP decoder has the same constraint length as that of the encoder, therefore, the decoder is able to recover from errors in a few branches by itself without feedback. The feedback from the outer decoder only speeds this process up; therefore, when an error is fed back, the most damage it can cause is for the RTMBEP decoder to make a few more errors before it recovers by itself. This is certainly a very desirable advantage for a concatenated coding system.

Moreover, because the decoder can restore its normal operation quickly, the degree of interleaving required for this scheme is considerably less than the Reed-Solomon block length required for the Zeoli's scheme.

As microprocessors are mass-produced, we foresee this as a promising practical scheme to achieve very reliable communication on a very noisy channel. Moreover, in our simulations. We have studied only the case when the outer decoder sends a corrected feedback to the inner decoder when an error was detected by the outer decoder. It is conceivable that one might improve the performance of the

concatenated coding system by employing the feedback from the outer decoder whenever the decision of the inner decoder is erased. This scheme requires further simulation results for different thresholds to determine its performance, but it is certainly a promising direction for future research.

Finally, as an interesting reminder to information theorists, we note that for the RTMBEP decoder with feedback system employing the unit-memory code concatenated with the (63 53), 6-error-correcting RS code, we can achieve a byte-error-probability of 10^{-7} at $E_s/N_o = -3.25$ dB (or $(E_b/N_o)_I = 1.25$ dB). The cut-off rate, R_o, of this 8-level quantized AWGN channel is 0.275, and the channel capacity is 0.44 whereas the over-all rate of our concatenated coding system is 0.27. It seems that the cut-off rate is still the practical limit of rate for reliable communications, even for a very sophisticated concatenated coding system.

BIBLIOGRAPHY

[1] P. Elias, "Error-Free Coding," IRE Transactions on Information Theory, Vol. IT-4, pp. 29-37, Sept. 1954.

[2] G.D. Forney, Jr., Concatenated Codes, M.I.T. Press, Cambridge, Mass., 1966.

[3] D.D. Falconer, "A Hybrid Sequential and Algebraic Decoding Scheme," Ph.D. Dissertation, Dept. of Electrical Engineering, M.I.T., Cambridge, Mass., 1966.

[4] F. Jelinek and J. Cocke, "Bootstrap Hybrid Decoding for Symmetrical Binary Input Channels," Information and Control, March, 1971.

[5] J.P. Odenwalder, Optimal Decoding of Convolutional Codes," Ph.D. Dissertation, School of Engineering and Applied Sciences, University of California, Los Angels, 1970.

[6] E.R. Berlekamp, Algebraic Coding Theory, McGraw-Hill Book Inc., New York, 1968.

[7] J.L. Massey, "Shift-Register Synthesis and BCH Decoding," IEEE Transcations on Information Theory, Vol. IT-15, No.1, pp. 122-125, Jan. 1969.

[8] G.W. Zeoli, "Coupled Decoding of Block-Convolutional Concatenated Codes," IEEE Transactions on Communications, Vol. COM-21, pp. 219-226, March 1973.

[9] F. Jelinek, "Bootstrap Trellis Decoding," IEEE Transctions on Information Theory, Vol. IT-21, No. 3, pp. 318-325, May 1975.

[10] J.M. Wozencraft and R.S. Kennedy, "Modulation and Demodulation for Probabilistic Coding," IEEE Transactions on Information Theory, Vol. IT-12, pp. 291-297, July 1966.

[11] J.L. Massey, "Coding and Modulation in Digital Communications," Proceedings, International Zurich Seminar on Digital Communications, Zurich, Switzerland, March 12-15, 1974, pp. E2(1)-E2(4).

[12] J.L. Massey and M.K. Sain, "Inverses of Linear Sequential Circuits," IEEE Transactions on Computers, Vol. C-17, pp. 330-337, April 1968.

[13] G.D. Forney, Jr., "Convolutional Codes I: Algebraic Structure," IEEE Transactions on Information Theory, Vol. IT-16, pp. 720-738, Nov. 1970.

[14] A.J. Viterbi, "Convolutional Codes and Their Performance in Communication Systems," IEEE Transactions on Communications, Vol. COM-19, pp. 751-772, October 1971.

[]5] R.J. McEliece and H.C. Rumsey, "Capabilities of Convolutional Codes," Jet Prop. Lab., California Inst. of Tech., Space Program Summary, 37-50, Vol. 3, pp. 248-251, April 1968.

[]6] K.J. Larson, "Short Convolutional Codes with Maximal Free Distance for Rates 1/2, 1/3, and 1/4," IEEE Transactions on Information Theory, Vol. IT-19, pp. 371-372, May 1973.

[]7] E. Paaske, "Short Binary Convolutional Codes with Maximal Free Distance for Rates 2/3. and 3/4," IEEE Transactions on Information Theory, Vol. IT-20, pp. 683-688, Sept. 1974.

[18] L. Calabi and E. Myrvaagnes, "On the Minimal Weight of Binary Group Codes," IEEE Transactions on Information Theory, Vol. IT-10, pp. 385-387, October 1964.

[19] H.J. Helgert and R.D. Stinaff, "Minimum-Distance Bounds for Binary Linear Codes," IEEE Transactions on Information Theory, Vol. IT-19, pp. 344-356, May 1973.

[20] A.J. Viterbi, "Error Bounds for Convolutional Codes and An Asmptotically Optimal Decoding Algorithm," IEEE Transactions on Information Theroy, IT-13, pp. 260-269, April 1967.

[21] L.R. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," IEEE Transactions on Information Theory, Vol. IT-20, pp. 284-288, March 1974.

[22] P.L. McAdam, L.R. Welch and C.L. Weber, "M.A.P. Bit Decoding of Convolutional Codes," 1972 International Symposium on Information Theory, Asilomar, California, Jan 1972.

[23] L.N. Lee, "Real-Time Minimal-Bit-Error Probability Decoding of Convolutional Codes," IEEE Transactions on Communications, Vol. COM-22, pp. 146-151, Feb. 1974.

[24] B.D. Fritchman and J.C. Mixsell, "Comments on 'Real-Time Minimal-Bit-Error Probability Decoding of Convolutional Codes'," IEEE Transactions on Communications, Vol. COM-22, pp. 1892-1894, Nov. 1974.

[25] L.N. Lee, "Author's Reply to 'Comments on Real-Time Minimal-Bit-Error Probability Decoding of Convolutional Codes'," IEEE Transactions on Communications, Vol. COM-22, pp. 1894-1895, Nov. 1974.

[26] J.M. Wozencraft and I.M. Jacobs, Principles of Communication Engineering, John Wiley and Sons, Inc., New York, 1965.

[27] I.M. Jacobs, "Sequential Decoding for Efficient Communications from Deep Space," IEEE Transactions on Communication Technology, Vol. COM-15, pp. 492-501, Aug. 1967.

[28] L.N. Lee, "Short, Unit-Memory, Byte-Oriented, Binary Convolutional Codes having Maximal Free Distance," Notre Dame Technical Report, EE-754, Dept. of Electrical Engineering, University of Notre Dame, Notre Dame, Indiana, July 1975. (To appear in IEEE Transactions on Information Theory, May 1976.)

[29] L.N. Lee, "On Optimal Soft-Decision Demodulation," Notre Dame Technical Report, EE-743, Dept. of Electrical Engineering, University of Notre Dame, Notre Dame, Indiana, July 1975. (To appear in IEEE Transactions on Information Theory, July 1976.)