https://ntrs.nasa.gov/search.jsp?R=19710023029 2020-03-11T20:14:22+00:00Z

General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.

- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.

- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.

- This document is paginated as submitted by the original source.

- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

Produced by the NASA Center for Aerospace Information (CASI)

CR 114358
Available to the Public

CONCATENATION OF CONVOLUTIONAL AND BLOCK CODES

FINAL REPORT

N71-32505
(ACCESSION NUMBER) 196 (PAGES) CR-114358 (NASA CR OR TMX OR AD NUMBER)
(THRU) G3 (CODE) 08 (CATEGORY)
FACILITY FORM 602

NASA CR 114358

AVAILABLE TO THE PUBLIC

CONCATENATION OF CONVOLUTIONAL AND BLOCK CODES

FINAL REPORT

by A. J. Viterbi

J. P. Odenwalder

W. J. Rosenberg

G. F. Zeoli

June 10, 1971

Distribution of this report is provided in the interest of information exchange. Responsibility for the contents resides in the author or organization that prepared it.

Prepared under Contract No. NAS-2-5478 by
School of Engineering and Applied Science
University of California,
Los Angeles, California

for

AMES RESEARCH CENTER

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

TABLE OF CONTENTS

		Page
Section I.	Summary of Results	1
Section II.	Comparison of Concatenated Decoding and Sequential Decoding System Performance.	3
Section III.	Encoder - Interleaving Complexity.	7
Appendix A -	Optimal Short Constraint-Length Convolutional Codes and Concatenated Coding and Decoding.	A1
Appendix B -	Coupled Decoding of a Convolutional-Block Concatenated System	B1
Appendix C -	Structural Properties of Convolutional Codes. . . .	C1

I. Summary of Results

Under this contract three types of studies were conducted:

a) fundamental properties of optimal short constraint-length convolutional codes and the corresponding maximum likelihood decoders. This includes the determination of optimum rate 1/2 and 1/3 convolutional code and the effect of quantization and memory truncation;

b) various approaches to concatenation of block and convolutional codes using short constraint length convolutional inner codes and long block length Reed-Solomon outer codes;

c) structural results involving catastrophic convolutional codes, finite-state representation, and consecutive zeros in convolutional codewords.

Appendix A contains most of the results on (a), which is prerequisite to the study of concatenated coding systems employing convolutional inner codes. Appendix C deals with the theoretical material of (c), which is useful for selecting good convolutional codes. The main purpose of this contract was to evaluate the various approaches to concatenated coding, as outlined in b. These are contained in Chapter 6 of Appendix A and in Appendix B.

Appendix A treats a relatively conventional concatenated system wherein the inner code is convolutional with constraint length 8 and rate 1/3 and the outer code is a 255 8-ary symbol Reed-Solomon code, for a total block length of 2040 bits. The only modification required by the convolutional inner code is the use of an interleaving buffer to disperse the burst errors of the inner convolutional decoder. It is found that a total buffer of 64K bits

is sufficient at the receiver. In the following section it is shown that this system outperforms a sequential decoder with this same buffer size and roughly the same order of complexity by 1 to 2 dB depending on the transmitted data rate.

Appendix B treats a more elaborate and ambitious concatenated decoding system, which we have termed coupled decoding. Here we incorporate the capability of erasing inner decoded words on the basis of low metric (decoding quality measure). In effect each block contains one Reed-Solomon outer codeword, but successive blocks are coupled through the inner convolutional code. In decoding, the metric based on a large number of inner codeword symbols is computed and used to determine whether the initial segment should be retained or erased. The outer Reed-Solomon decoder both corrects errors and fills in erasures. The coupled system requires approximately the same size buffer as the basic concatenated system. It outperforms the latter by about 1/4 to 1/2 dB, but this gain is partly illusory. For an error in the concatenated system affects only one block (of 2040 bits), while an error in the coupled system affects all successive blocks. Thus the coupled system would have to be operated at error probabilities which are at least two orders of magnitude lower, thereby eliminating most of the dB margin. Thus while coupled decoding potentially outperforms concatenated decoding, in practice its meager advantage combined with its greater complexity effectively precludes further consideration. On the other hand, in the next Section we show that more conventional concatenated decoding significantly outperforms sequential decoding at data rates up to 1 Mbps.

II. Comparison of Concatenated Decoding and Sequential Decoding System Performance

In order to compare a concatenated convolutional-block coding system with sequential decoding of long constraint-length convolutional codes, we must first impose certain speed-complexity constraints as well as performance requirements. These are

a) a maximum buffer size of 64K bits be used in either system;

b) speed be limited by the capabilities of TTL integrated circuit logic;

c) outer (Reed-Solomon) decoding be performed by a dedicated mini-computer;

d) block error probabilities of 10^{-8} for both systems.

Both systems will be assumed to operate on soft quantized (8-level or 3-bit) data and rate 1/3 codes. We consider two transmitted data speeds: 1 Mbps, which appears to be nearly the limit for soft decision decoding in both systems, and 10 Kbps. The margin of the concatenated decoding system over the sequential decoding system will be shown to increase with increasing data speed.

We shall take the concatenated system to consist of blocks of 2048 bits (256 8-bit characters, the last 8-bit character consisting of known bits being used to tail off the convolutional code), we take the sequential decoding frame (block) length to be also 2048 bits including a known tail* of 30 to 40 bits equal to the code constraint length. The error probability here is essentially just the probability of buffer overflow in decoding the block, because the undetected error probability is negligible for a constraint length 40 code. The overflow probability has been shown by

---

*Although resynchronization without tailing off the data (by inserting a constraint length of known bits) is possible, it is probably impractical except with hard-decision (binary) data.

Jacobs (1967) and others to be approximately

$$P_o \approx k(\mu B)^{-\alpha}$$

where μ is the sequential decoder speed factor

B is the buffer size (in branches)

α is the Pareto exponent, which is a function of E_b/N_o, and k is a constant proportional to the block length

In this case we may take $k \approx 2 \times 10^3$. Also with a buffer of 64 K bits, since we must store 9 bits/branch for a rate 1/3, 8-level quantized nonsystematic code, we have $B = 7.1 \times 10^3$ branches. The maximum computing speed of a sequential decoder employing TTL logic* appears to be approximately 13 M comps/sec. Thus $\mu = 1.3 \times 10^7/R$, where R is the transmission rate in bits/sec. and

$$P_o \approx \frac{2 \times 10^3}{(10^{11}/R)^{\alpha}}$$

To achieve $P_o \approx 10^{-8}$, it follows that with $R = 10^6$ bits/sec., we must have $\alpha \approx 2.25$ while with $R = 10^4$ bits/sec., we must have only $\alpha \approx 1.6$. The corresponding required E_b/N_o for these α, as given by Jacobs (1967) are 4.1 dB and 3.1 dB, respectively. We should add .05 dB in each case for the energy lost in the tail. The results are summarized in Table I.

On the other hand, from Figures 6.4 and 6.5 of Appendix A, we find that the concatenated system requires only $E_b/N_o = 2.15$ dB to achieve a block error probability of 10^{-8}, a margin of 2 db at 1 Mbps data speeds and 1 dB at 10 Kbps speeds. The interleaving buffer required is 64 K bits,

---

*as implemented in a hard-decision TTL decoder by Codex Corporation. We assume that with more elaborate logic, soft-decision decoding can be implemented at this same speed.

as discussed in Chapter 6 of Appendix A. The encoder interleaving buffer can be implemented much more simply than the decoder buffer, and requires in all 2ei bits of storage, where e is the number of errors to be corrected by the outer code and i is the depth of interleaving. Furthermore, as is shown in Section III, this storage can be incorporated into just 4e integrated circuits, consisting of 2e 8-bit registers and 2e 2^r-bit registers. As for the decoder complexities, we have chosen 1 Mbps as the highest data rate for which the inner maximum likelihood (Viterbi) decoder for constraint-length 8 and code rate 1/3 is practically feasible*.

Transmitted Data Rate	1 Mbps	10 Kbps
Pareto exponent α (Sequential Decoding)	2.25	1.6
E_b/N_o required for Sequential Decoding	4.15 dB	3.15 dB
E_b/N_o required for Concatenated Decoding	2.15 dB	2.15 dB
Margin of Improvement	2 dB	1 dB

Table I

Relative Performance of Concatenated and Sequential Decoding

Actually, the inner decoder must operate at a slightly higher data rate (about 1.2 Mbps) to accommodate the outer code check digits, which are on the order of 40 to 50 out of 255. (Note, however, that the E_b/N_o is not proportionately increased because the curves of Figures 6.4 and 6.5 of

---

* a constraint-length 7, code rate 1/2, Viterbi decoder has been implemented by Linkabit Corporation for data rates up to 2 Mbps. It employs only 16 metric calculators in a serial-parallel arrangement. Cutting the speed in half permits increasing the constraint length by unity. A rate 1/3 implementation is only moderately more complex than that for rate 1/2.

Appendix A already contain this adjustment). The outer decoder must decode about $(1.2 \times 10^6)/(2 \times 10^3) \approx 600$ Reed-Solomon words/sec. Utilizing a full-serial computer implementation, the speed is limited primarily by the maximum number of errors which are to be corrected, for this specifies the number of iterations required in the Berlekamp iterative BCH decoding algorithm. Thus it may not be possible to correct as many as 20 errors of the outer code at a decoding speed of 600 words/sec. On the other hand, as can be seen in Figure 6.4 of Appendix A, by increasing E_b/N_o by about 1/2 dB, we can reduce the number of required correctable errors of the outer code to only four or five. At the lower speed (10 Kbps) the implementation of both inner and outer decoders becomes much simpler and full-serial decoding can be used in the inner as well as the outer decoder implementation. Also the outer decoder must process only 6 words/sec., so that no limitations need to be imposed on the maximum number of correctable errors.

III. Encoder - Interleaving Complexity

In Section II we considered the required complexity of the decoder and associated de-interleaving memory for the concatenated system of Appendix A. Now we concentrate on the encoder and associated interleaver, whose complexity needs to be considerably more limited for space telemetry applications.

The inner encoder complexity is minimal because of the short constraint length. The outer encoder is considerably simplified by the fact that the Reed-Solomon code is a high rate cyclic code (with far fewer parity-check symbols than information symbols). In Gallager* (1968) it is shown that such a code is most efficiently generated by a shift register with a number of stages equal to the number of parity-check symbols, with equal numbers of adders and scalers. Of course, in this case each stage must store 8 bits and arithmetic is performed over $GF(2^8)$. A Reed-Solomon code with 2e parity-check symbols will correct e symbol errors. Ignoring for the moment the interleaving requirements, we note that the 2e parity-check symbols of each Reed-Solomon codeword are generated by a shift register of length 2e 8-bit stages, which thus has a total storage capacity of 16e bits. These parity-check symbols are inserted at the end of the codeword, after the 255-2e information symbols**.

---

* p. 225, Figure 6.5.5

** We assume that the information stream can be interrupted for 2e symbol times (2ei symbol times in the interleaved case discussed below) to insert the parity checks. This assumption is valid when the data stream consists of several time-division-multiplexed users (where each is alloted a data burst whose length is a multiple of 2e), or for a pictorial scan application where the parity-check symbols can be inserted between successive scans.

With this encoder implementation, interleaving can be performed in a very simple and inexpensive manner. Suppose we choose to interleave i = 16 Reed-Solomon words. Then the interleaver can be implemented as an integral part of the encoder by inserting fifteen 8-bit symbols (or 120 bits) of delay between successive encoder stages. The outer encoder-interleaver operation can be described with the aid of Figure 6.2 of Appendix A. The first K = 8 bit information symbol of the first Reed-Solomon word is fed to the inner (convolutional) encoder and also inserted into the first (leftmost) stage of the shift register of the outer encoder parity-check generator. This is followed by the first symbols of the remaining i - 1 = 15 Reed-Solomon words. Then all the second 8-bit symbols of the 16 Reed-Solomon words are both fed to the inner encoder and to the parity-check generator. This continues until all 255-2e information symbols of each of the 16 Reed-Solomon codewords have been exhausted. We note that in this manner the outer encoder parity-check generator is time-shared among the 16 words and that the partial parity-check sums are stored in the delay lines between encoder stages for 15/16-th of the time. Once the information symbols have been exhausted, the contents of the leftmost stage of the register contains the first parity-check symbol of the first codeword, which can therefore be fed next to the inner encoder. Continuing to shift for an additional $2ei = 2^5e$ symbol times (or 2^8e bit times), but without inserting any new information symbols, we thereby generate all 2ei parity-check symbols of the i = 16 Reed-Solomon codewords, which are fed from the leftmost stage of the outer encoder to the inner convolutional encoder.

The total storage required in the encoder shift register consists, therefore, of 2e 8-bit registers and 2e 120-bit registers for a total storage of $2^8 e$ bits. Even for e = 20 the storage is reduced by more than an order of magnitude relative to the decoder requirements. Even more important is the form of the encoder storage, which allows for considerable simplification through the use of integrated circuits. In particular if we choose $i - 1 = 2^r$, where r is some integer, the total encoder and interleaver memory can be implemented by using 4e integrated-circuit shift-registers, 2e being 8-bit register and the remaining 2e being 2^r-bit register. At the data rates discussed in Section II, such integrated circuits are available for $r = 7 (2^r = 128)$ and even higher. Thus, with $e \leq 10$, the complexity of the outer encoder and interleaver appears to be on the order of that required for long constraint-length convolutional encoders employed in sequential decoding applications. As noted above, the additional complexity required here for the inner short constraint-length convolutional encoder is nearly negligible by comparison.

REFERENCES

Bahl, L. R., and F. Jelinek (1970), "Free Distance Structure of Convolutional Codes and Complementary Rate 1/2 Codes," unpublished paper (L. R. Bahl is currently at IBM, Yorktown Heights, N.Y., F. Jelinek is currently at Cornell University, Ithaca, N.Y.).

Berlekamp, E. R. (1968), Algebraic Coding Theory, McGraw-Hill, New York.

Bose, R. E. and D. K. Ray-Chaudhuri (1960), "On a Class of Error Correcting Binary Group Codes," Information and Control, 3, 68-79.

Costello, Daniel J., Jr. (1969), "Construction of Convolutional Codes for Sequential Decoding," Ph.D., Department of Electrical Engineering, University of Notre Dame, Notre Dame, Indiana.

Elias, P. (1954), "Error-free Coding," IEEE Trans. Inform. Theory, IT-4, 29-37.

Fano, R. M. (1961), Transmission of Information, M.I.T. Press, Cambridge, Mass.

Fano, R. M. (1963), "A Heuristic Discussion of Probabilistic Decoding," IEEE Trans. Inform. Theory, IT-9, 64-74.

Feinstein, A. (1954), "A New Basic Theorem of Information Theory," IEEE Trans. Inform. Theory, IT-4, 2-22.

Feinstein, A. (1958), Foundations of Information Theory, McGraw-Hill, New York.

Forney, G. D., Jr. (1966), Concatenated Codes, M.I.T. Press, Cambridge, Mass.

Forney, G. D., Jr. (1967), "Coding System Design for Advanced Solar Missions," Natl. Aeron. and Space Adm., Final Report, Contract NAS2-3637, Codex Corp., Watertown, Mass., December.

Forney, G. D., Jr. (1970), "Convolution Codes I: Algebraic Structure," IEEE Trans. Inform. Theory, IT-16, 720-738.

Gallager, R. G. (1965), "A Simple Derivation of the Coding Theorem and Some Applications," IEEE Trans. Inform. Theory, IT-11, 3-18.

Gallager, R. G. (1968), Information Theory and Reliable Communication, Wiley, New York.

Golomb, S. W. (1967), Shift Register Sequences, Holden-Day, San Francisco.

Heller, J. A. (1968), "Short Constraint Length Convolutional Codes," Jet Propulsion Laboratory, Space Programs Summary, 37-54, Vol. III, 171-177.

Heller, J. A. (1969), "Improved Performance of Short Constraint Length Convolutional Codes," Jet Propulsion Laboratory, Space Program Summary, 37-56, Vol. III, 82-83.

Hocquenghem, A. (1959), "Codes Correcteurs d'Erreurs," Chiffres, 2, 147-156.

I.B.M. Staff (1959), "Random Number Generation and Testing," I.B.M. Data Processing Techniques Report, C20-8011.

Jacobs, I. M. (1967), "Sequential Decoding for Efficient Communication," IEEE Trans. Communication Technology, COM-15, 492-501.

Littlewood, D. F. (1950), A University Algebra, William Heinemann Ltd., London.

Liu, C. L. (1969), Introduction to Combinatorial Mathematics, McGraw-Hill, New York.

Massey, J. L. (1963), Threshold Decoding, M.I.T. Press, Cambridge, Mass.

Massey, J. L. (1968), "Catastrophic Error-Propagation in Convolutional Codes," Proceedings, Eleventh Midwest Symposium on Circuit Theory, University of Notre Dame, Notre Dame, Indiana, May 13-14, 1968, Ave Maria Press, Notre Dame, Indiana.

Massey, J. L. and M. K. Sain (1968), "Inverses of Linear Sequential Circuits," IEEE Trans. Computers, C-17, 330-337.

Massey, J. L. (1968), "Some Algebraic and Distance Properties of Convolutional Codes," Error Correcting Codes, Editor H. B. Mann, Wiley, New York.

Miczo, A. and L. D. Rudolph, "A Note on the Free Distance of a Convolutional Codes," IEEE Trans. Inform. Theory, IT-16, 646-648.

Mood, A. M. (1950), Introduction to the Theory of Statistics, McGraw-Hill, New York.

Morrissey, Thomas N., Jr. (1968), A Unified Markovian Analysis of Decoders for Convolutional Codes, Ph.D. dissertation, Department of Electrical Engineering, University of Notre Dame, Notre Dame, Indiana.

Odenwalder, Joseph P. (1970), "Optimal Decoding of Convolutional Codes," Ph.D. in Engineering, University of California at Los Angeles.

Peterson, W. W. (1961), Error-Correcting Codes, M.I.T. Press, Cambridge, Mass.

Reiffen, B. (1960), "Sequential Encoding and Decoding for the Discrete Memoryless Channel," M.I.T. Lincoln Laboratory, Lexington, Mass., Report No. 25, G-0018.

Riodran, J. (1958), An Introduction to Combinatorial Analysis, Wiley, New York.

Robinson, J. P. (1965), "An Upper Bound on the Minimum Distance of a Convolutional Codes," IEEE Trans. Inform. Theory, IT-11, 567-571.

Rosenberg, W. J. (1971), Structural Properties of Convolutional Codes, Ph.D. in Engineering, University of California at Los Angeles.

Shannon, C. E. (1948), "A Mathematical Theory of Communication," Bell System Tech. J., 27, 379-423 (Part I), 623-656 (Part II).

Shannon, C. E. (1949), "Communication in the Presence of Noise," Proc. IRE, 37, 10-21.

Shannon, C. E., R. G. Gallager, and E. R. Berlekamp (1967), "Lower Bounds to Error Probability for Coding on Discrete Memoryless Channels," Information and Control, 10, 65-103 (Part I), 522-552 (Part II).

Varga, R. S. (1962), Matrix Iterative Analysis, Prentice-Hall, Englewood Cliffs, New Jersey.

Viterbi, A. J. (1967), "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," IEEE Trans. Inform. Theory, IT-13, 260-269.

Viterbi, A. J. (1969), "The State-Diagram Approach to Optimal Decoding and Performance Analysis for Memoryless Channels," Jet Propulsion Laboratory, Space Programs Summary, 37-58, Vol. III, 50-55.

Viterbi, A. J. and J. P. Odenwalder (1969), "Further Results on Optimal Decoding of Convolutional Codes," IEEE Trans. Inform. Theory, IT-15, 732-734.

Wozencraft, John M. (1957), "Sequential Decoding for Reliable Communications," Sc.D. Thesis, Department of Electrical Engineering, Massachusetts Institute of Technology, Cambridge, Mass.

Wozencraft, John M. and Barney Reiffen (1961), Sequential Decoding, M.I.T. Press, Cambridge, Mass.

Wozencraft, John M. and Irwin M. Jacobs (1965), Principles of Communication Engineering, Wiley, New York.

Yudkin, H. L. (1964), Channel State Testing in Information Decoding, Ph.D. in Electrical Engineering, Massachusetts Institute of Technology, Cambridge, Mass.

APPENDIX A*

Optimal Short Constraint-Length Convolutional Codes and Concatenated Coding and Decoding

by

J. P. Odenwalder

*extracted from Ph.D. Dissertation of J. P. Odenwalder "Optimal Short Constraint-Length Convolutional Codes and Concatenated Coding and Decoding", 1970. Chapter numbers correspond to dissertation chapters.

CHAPTER 3

THE VITERBI DECODING ALGORITHM IN TERMS OF THE TRELLIS DIAGRAM

The primary purpose of this chapter is to present the trellis diagram [Forney (1967)] for convolutional codes and to use it to give a simple explanation of the Viterbi decoding algorithm. This explanation is in terms of an example, which also shows some of the practical problems which would be encountered in building or simulating a Viterbi decoder. This discussion is similar to an explanation of the Viterbi decoding algorithm given by Heller.† The chapter concludes with a simple proof that the Hamming distances between the received sequence and the retained paths in the Viterbi decoding algorithm cannot differ by more than a fixed amount. This fact was first noted by Morrissey (1968).

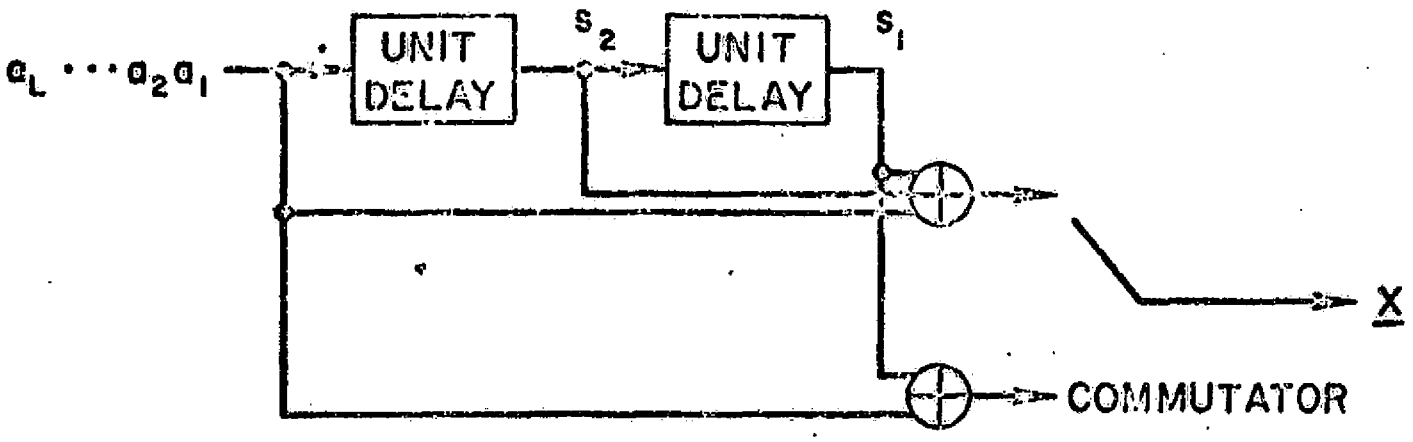
3.1 The Trellis Diagram

Convolutional codes have long been thought of as tree codes which continue to grow until the tree is terminated by inserting K-1 known data symbols into the shift register. However, this obscures the fact that after the input symbols have been identical for K-1 consecutive symbols, the paths remerge (i.e., they produce the same channel symbols). Indeed, this observation is the basis of the Viterbi algorithm. Recently, Forney (1967) has described a trellis diagram which clearly shows this merging property. This trellis diagram will be described in terms of a simple example, and the method can easily be extended to more complex systems.

Consider the binary-input rate $R = \frac{1}{2}$ (in bits/channel symbol) constraint length K=3 convolutional encoder shown in Figure 3.1.

---

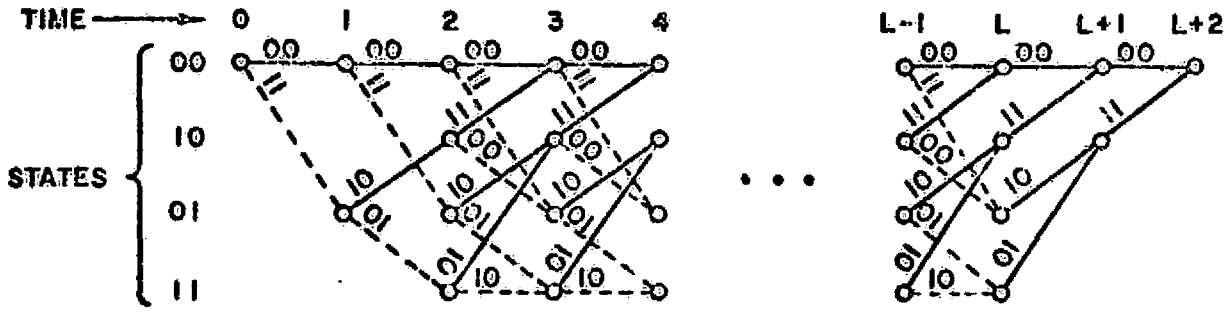
†"Short Constraint Length Convolutional Codes," short talk presented at the IEEE International Communications Conference, June 9-11, 1969, Boulder, Colorado.



CONVOLUTIONAL ENCODER

FIGURE 3.1

The additions are mod 2 and the state $\underline{S} = (s_1, s_2)$ is initially (00). Now let the first bit a_1 be shifted into the shift register. Since a_1 could be a "0" or a "1" the state at time 1 could be (00) or (01) with output symbols 00 or 11, respectively. Similarly when the other symbols are shifted into the shift register the state transitions and corresponding output symbols can be computed. All of the possible state transitions can be conviently shown on a trellis diagram, as in Figure 3.2.



TRELLIS FOR THE ENCODER OF FIGURE 3.1

FIGURE 3.2

The dotted lines represent "1" input bits and the solid lines represent "0" input bits. The output symbols generated in each transition are also shown. Note that since the input sequence is terminated with K 1=2 zeros, the last two transi ions contain only solid lines. Also the output symbols on branches coming from the same node are complements of each other. This is always true when there is a connection between every mod 2 adder and the input.. If there is a connection between every mod 2 adder and the output of the last unit delay, then the output symbols on branches entering the same node are also complements. In Chapter 4 it is shown that one of these cases may be assumed without loss of generality and all of the optimum (in a sense to be defined in Chapter 4) codes examined have a complete set of connections at the encoder input and at the output of the last unit delay. Thus, even though the number of branches in a binary tree doubles with each unit time increase, all of the possible paths can be displayed on a trellis diagram with 2^{K-1} states, regardless of the length of the input sequence.

3.2 A Viterbi Decoding Algorithm Example

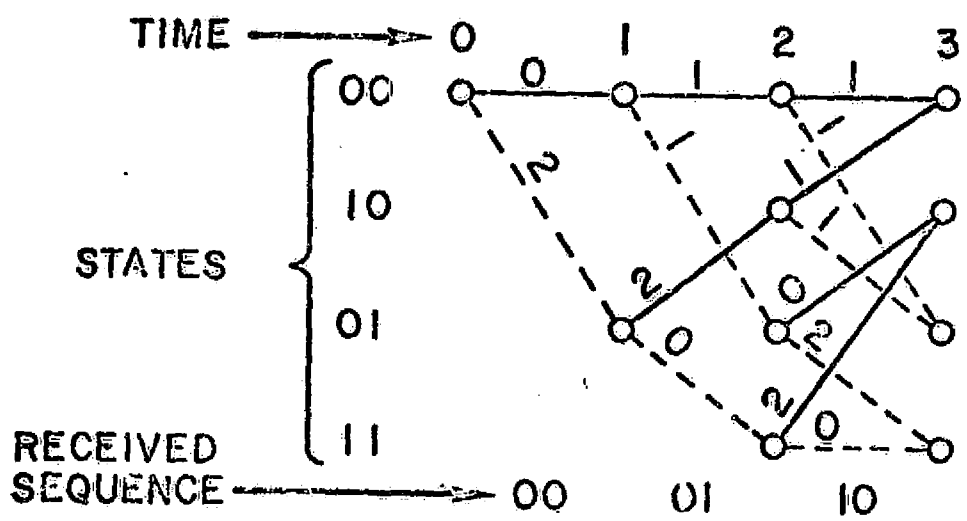
Now let us examine a digital communication system which uses the convolutional encoder of Figure 3.1. In particular, let us transmit L=4 zero message bits over a binary symmetric channel and use a Viterbi decoder, which, as was shown in Chapter 2, is equivalent to a maximum-likelihood decoder. The encoder must then decide which of the 2^L possible received sequences were transmitted. It does this by choosing a path with the largest log-likelihood function, and if more than one path has the same highest log-likelihood function, some tie-breaking rule must be used. This tie-breaking rule will not affect the path log-likelihood function, but it can change the bit probability of error. In practice, the channel output will usually be a quantized

version of a continuous random variable, and as the number of quantization levels increases, the probability of a tie becomes negligible.

For a binary symmetric channel with crossover probability $p < \frac{1}{2}$, choosing the path with the largest log-likelihood function is equivalent to choosing the path closest in Hamming distance to the received sequence. Now assume the first six received symbols are

00 01 10 . . .

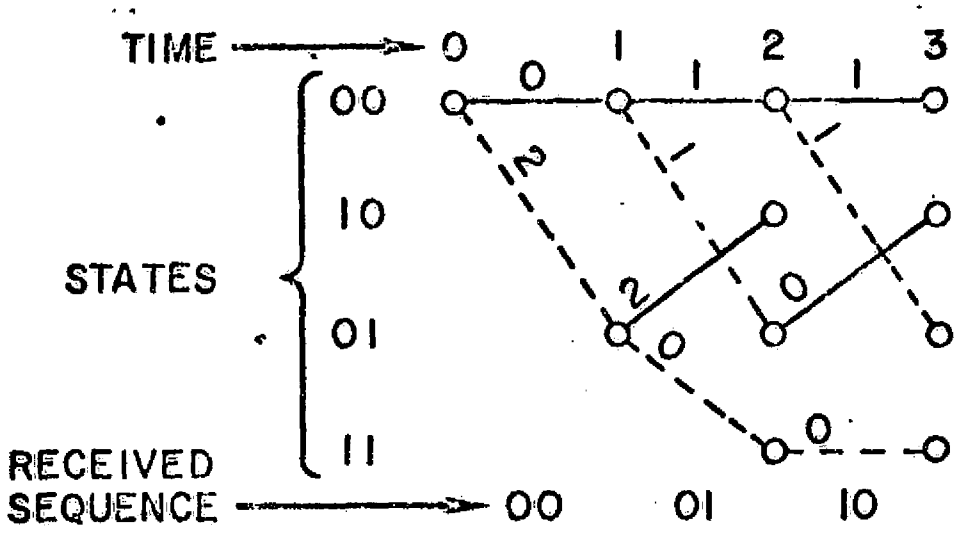
Then consider another trellis showing all the possible transmitted sequences up to time 3 (i.e., the first K branches), and lable each branch with the Hamming distance between the output symbols of that branch and the corresponding bits of the received sequence, as shown in Figure 3.3.



TRELLIS AT TIME 3

FIGURE 3.3

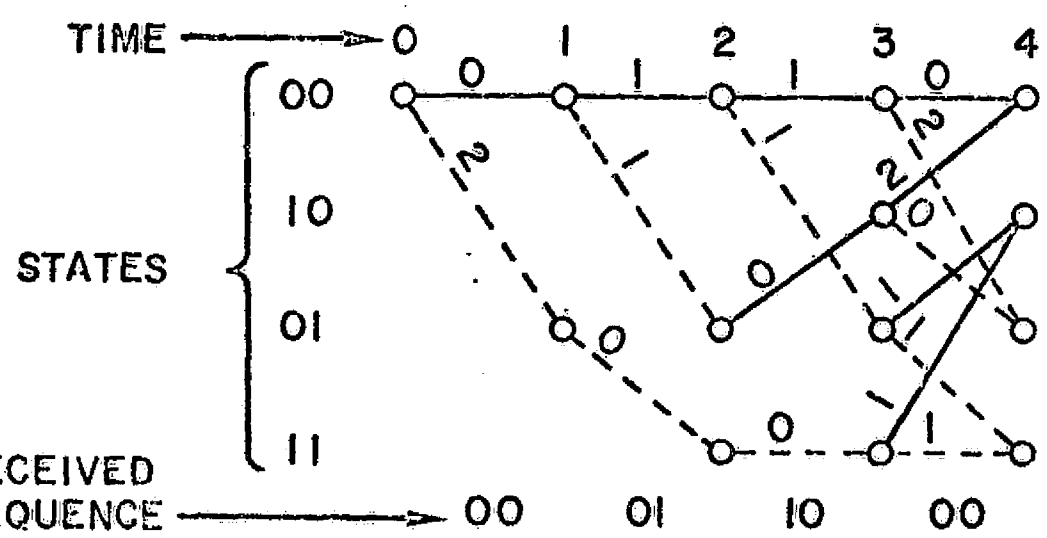
As in Chapter 2, denote the paths by their corresponding input sequences with the initial bit written first. Then the Viterbi decoder begins by examining the first K branches and making $2^{K-1} = 4$ pairwise comparisons among those paths with the same last K-2=2 symbols. After these comparisons are made, one of the two branches entering each state at time 3 can be eliminated, as in Figure 3.4.



TRELLIS AT TIME 3 AFTER COMPARISONS

FIGURE 3.4

Figure 3.4 can obviously be simplified by removing the branch from state (01) at time 1 to state (10) at time 2. This leaves four possible paths. Now assume two more bits are received, for example, 00. This is shown on the trellis of Figure 3.5

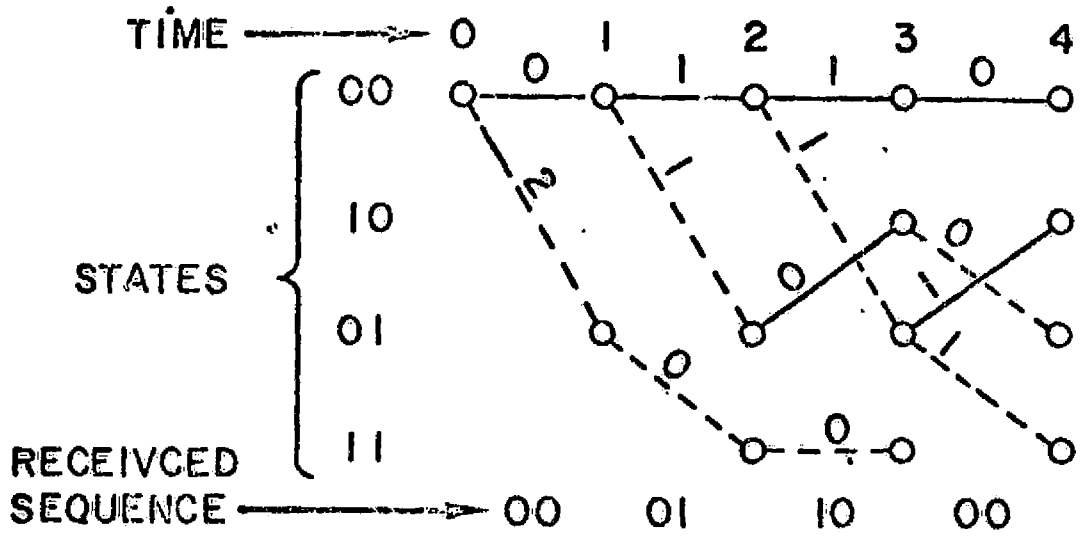


TRELLIS AT TIME 4

FIGURE 3.5

Again the Viterbi decoder makes 2^{K-1} pairwise comparisons among those paths with the same last K-1 symbols. This time two of the

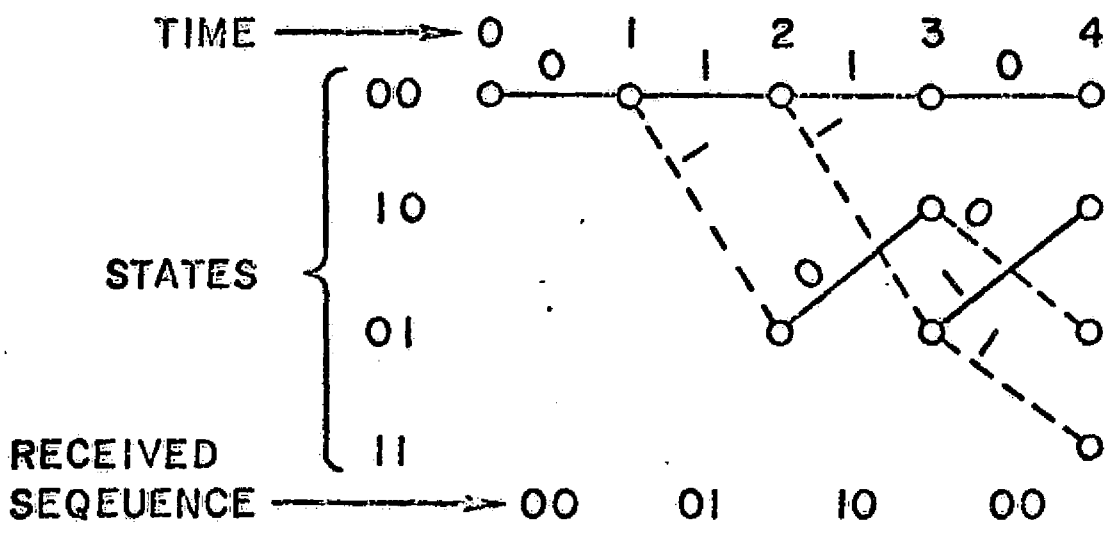
comparisons end up as ties. Assume these ties are resolved as in Figure 3.6.



TRELLIS AT TIME 4 AFTER COMPARISONS

FIGURE 3.6

Figure 3.6 can then be simplified to Figure 3.7.

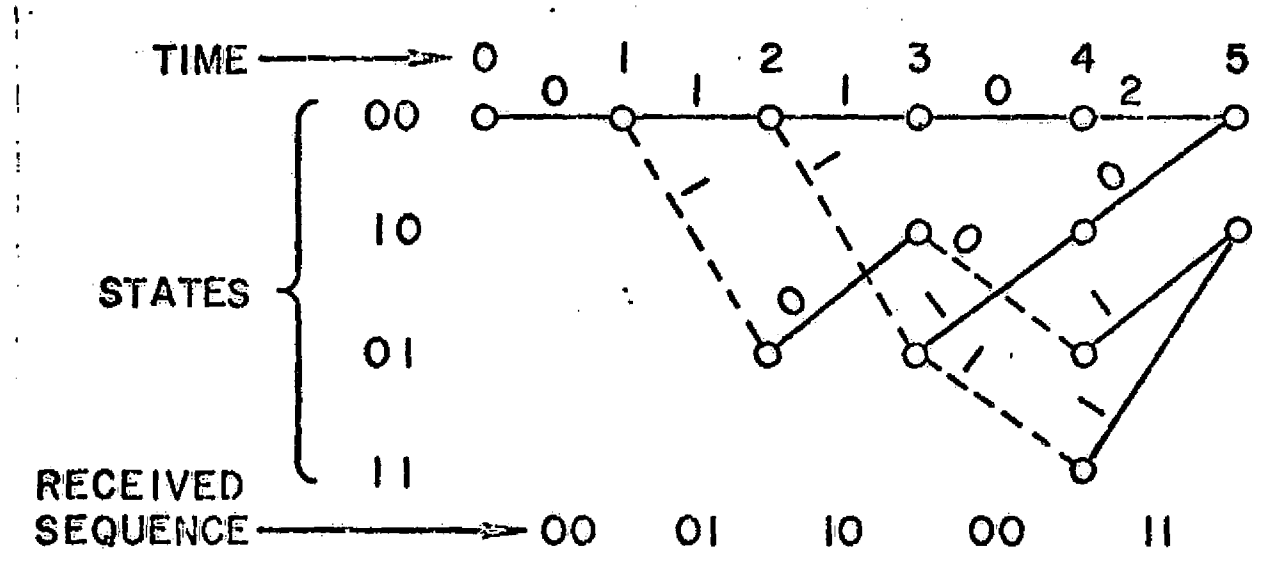


TRELLIS AT TIME 4 AFTER SIMPLIFICATIONS

FIGURE 3.7

At this point all four paths have the same first bit, so the decoder can make a decision on this bit before the remainder of the message is received.

Each time a pair of symbols is received, the trellis is extended one time unit, resulting in 2^K possible paths. Then the number is reduced to 2^{K-1} by making pairwise comparisons among those paths with the same last K-1 symbols. In this example there were only four message bits, so the K-1 terminating zeros now enter the shift register. Assume the next two received bits are both ones. The trellis can then be extended as in Figure 3.8.

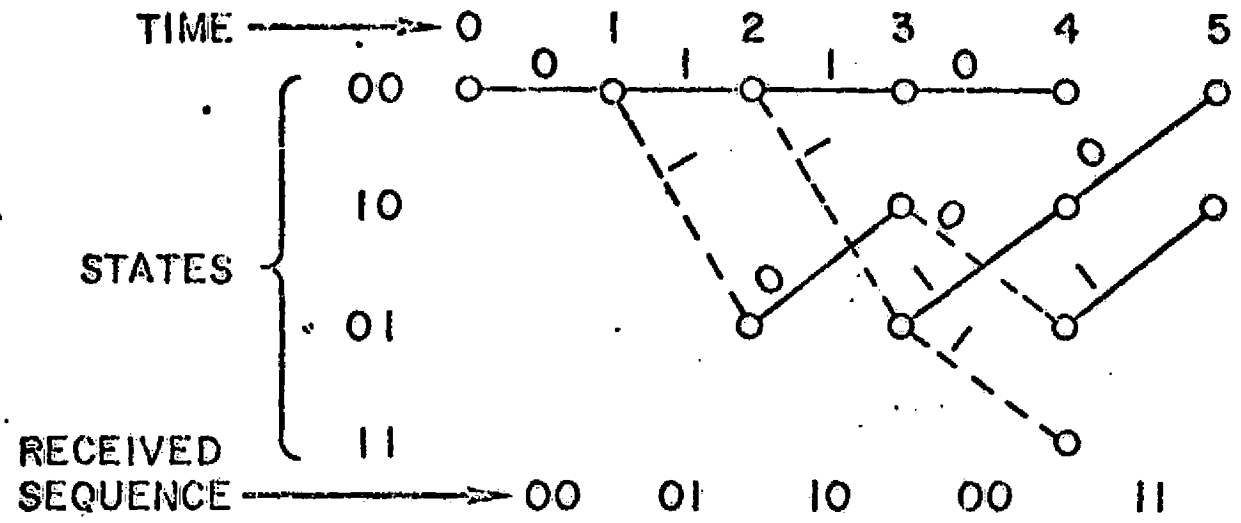


TRELLIS AT TIME 5

FIGURE 3.8

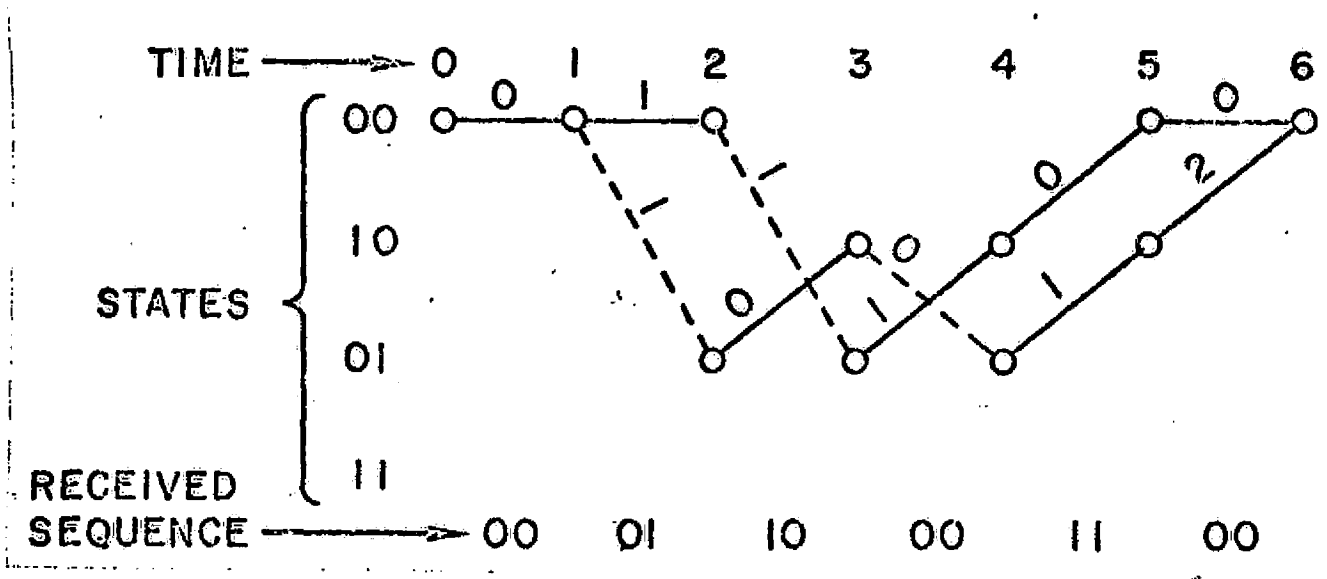
This time only half as many pairwise comparisons are made, since the paths can only end with 00 or 10. The results of these comparisons are shown in Figure 3.9. Finally, assume the last two received bits are both zero. Then simplifying Figure 3.9 and extending it one time unit gives Figure 3.10.

Thus after making the final comparison, the Viterbi decoder chooses the path 001000. Since all zeros were transmitted, it has made a word decoding error and this word contains one bit error. However, no path is closer in Hamming distance to the received sequence than the path chosen, and a maximum-likelihood decoder which compares all 2^L possible paths at the end of the tree would also have



TRELLIS AT TIME 5 AFTER COMPARISONS

FIGURE 3.9



TRELLIS AT TIME 6

FIGURE 3.10

made a word decoding error. For the path chosen is a distance 3 from the received sequence and the all zero message is a distance 4 from the received sequence.

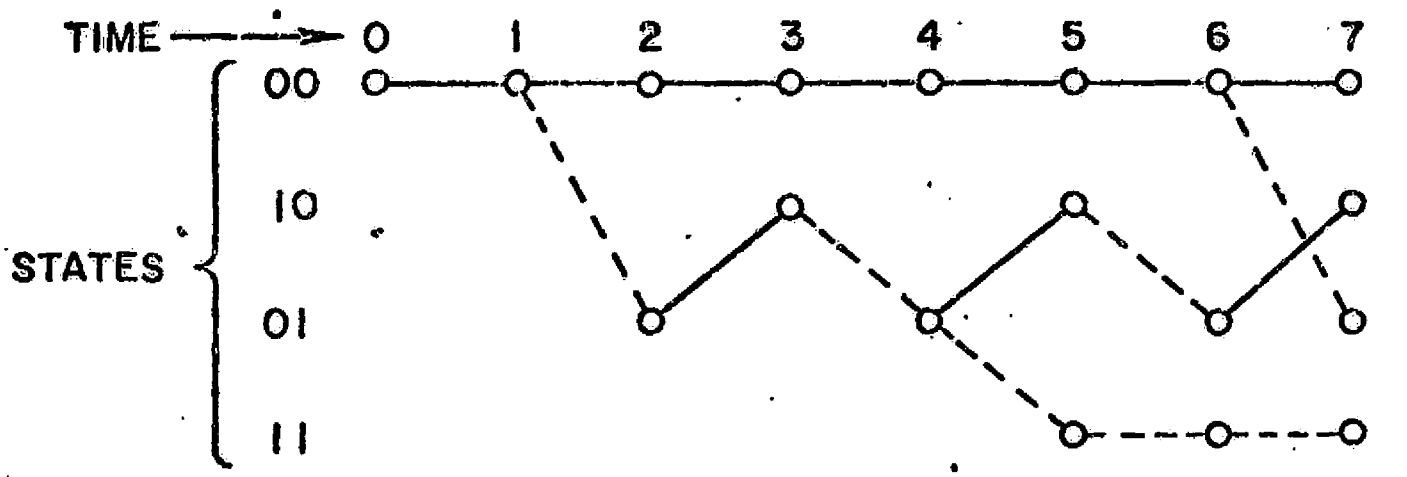
A display similar to this sequence of figures has been programmed on a graphics terminal with an SDS Sigma-7 computer. Besides the binary symmetric channel this program can handle a

binary-input additive white Gaussian noise channel with symmetric output quantization. Since this program is similar to the simulation program discussed in Chapter 5, it will not be discussed further here. This display is a useful aid in understanding the Viterbi decoding algorithm and has been helpful in studying memory truncation.

3.3 Memory Truncation

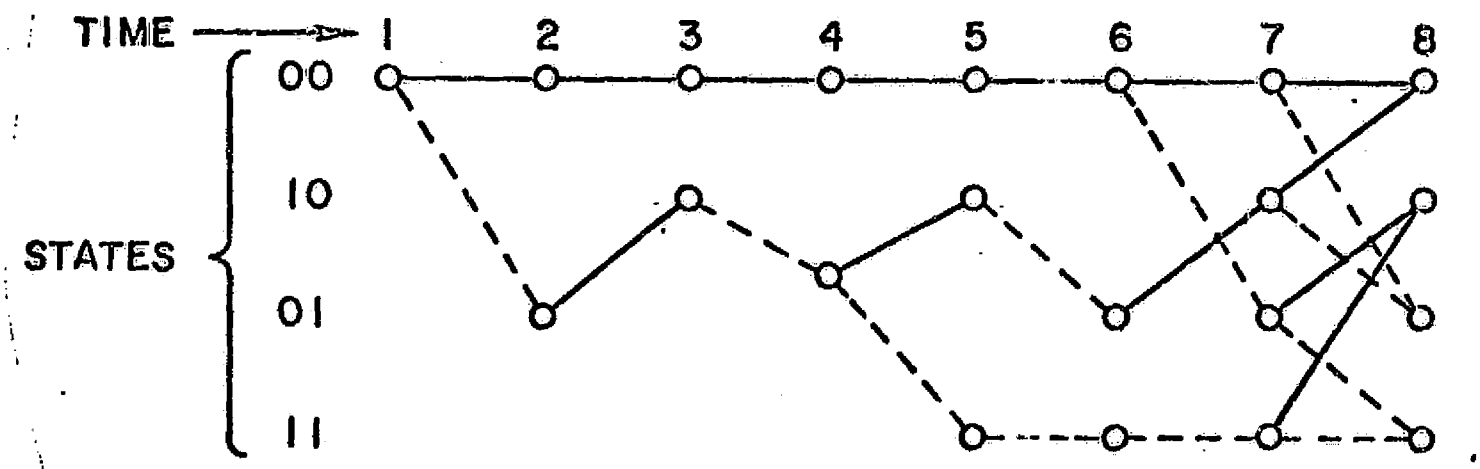
In practice, several thousand information bits may be transmitted before the trellis (or tree) is terminated with K-1 zeros. In fact, with a Viterbi decoder the entire message may be transmitted before the terminating sequence is inserted. However, as the number of information bits becomes large, the storage requirements for the 2^{K-1} paths soon become excessive. Fortunately, as noted in the preceding example, the paths tend to merge several constraint lengths back and decisions can be made on those bits. Thus only the unmerged portion of the paths need be stored. The problem of choosing a suitable memory length is discussed in Chapter 5. Here the problem will simply be illustrated using a trellis diagram.

Consider the encoder used in the previous example, and assume that a long sequence of zero information bits are going to be encoded and transmitted. Also assume that the trellis can only store 2^{K-1}, 7 bit sequences. That is, the trellis is only 7 time units long. Then at time 7 let the trellis be as in Figure 3.11. Since all four paths are merged up to time 1, a decision can be made on the first bit, and when the next pair of symbols are received, the trellis of Figure 3.11 can be shifted to the left, as shown in Figure 3.12. After the comparisons assume that this trellis reduces to that of Figure 3.13. When the next pair of symbols are received, no decision can be made on the second bit, but the trellis must still be shifted to the left, as in Figure 3.14.



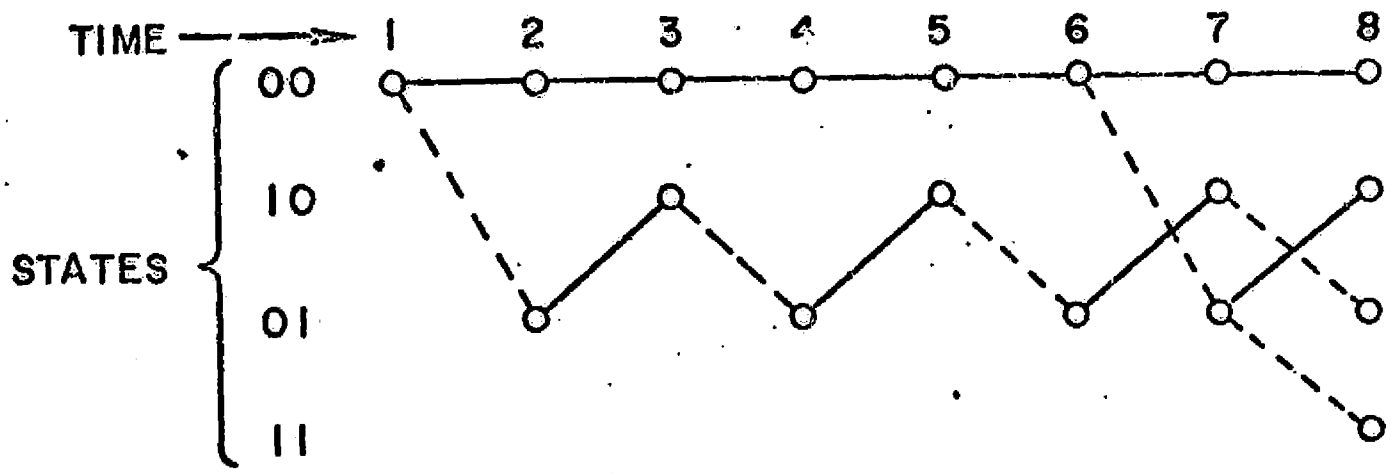
TRELLIS AT TIME 7

FIGURE 3.11



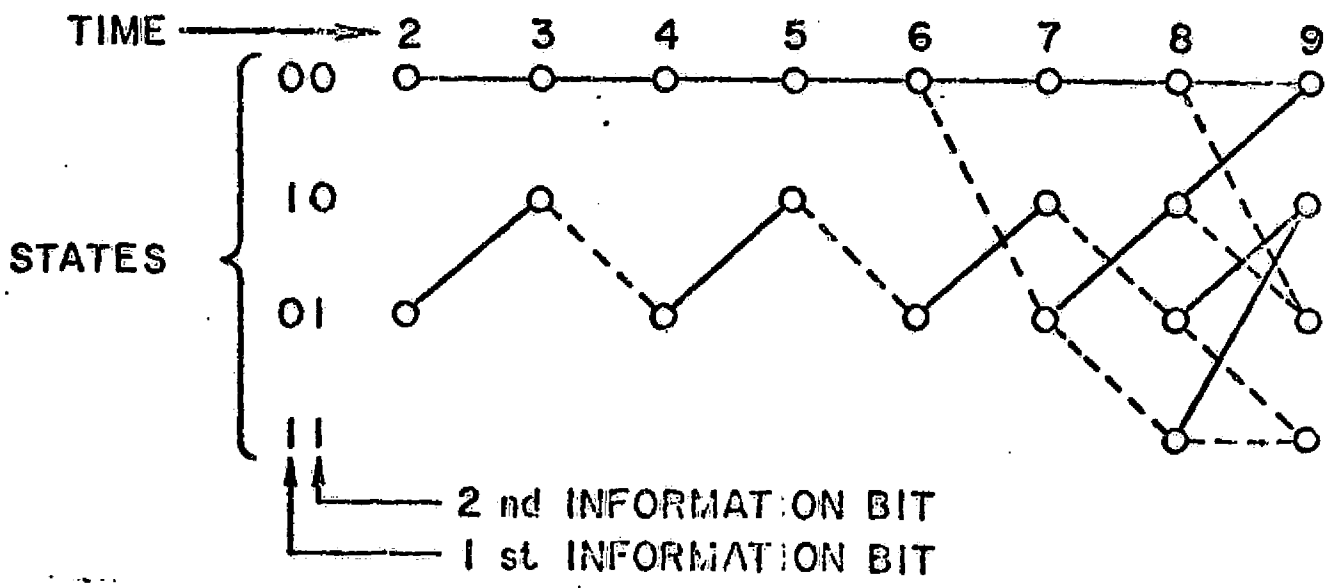
TRELLIS AT TIME 8

FIGURE 3.12



TRELLIS AT TIME 8 AFTER COMPARISONS

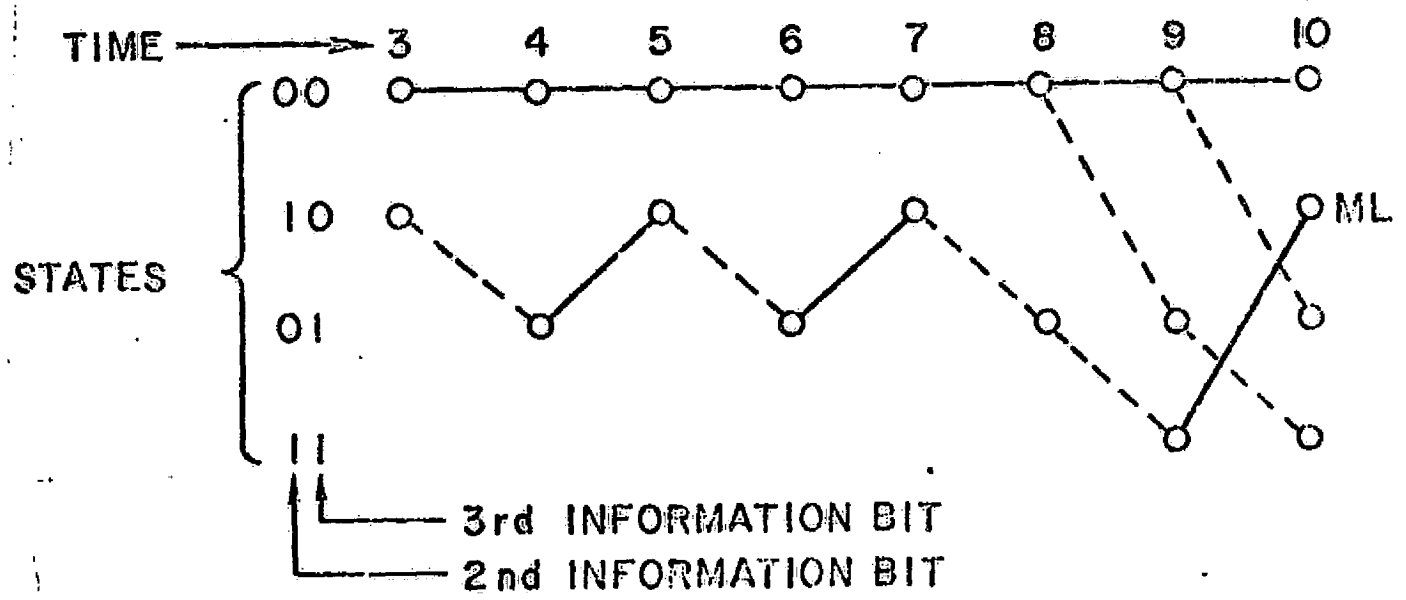
FIGURE 3.13



TRELLIS AT TIME 9

FIGURE 3.14

Now recall that given the state, the previous K-1 symbols can be determined. When the states are chosen as in Figure 3.1 a state $(s_1 s_2)$ means that the two previous information symbols were $s_1 s_2$, as labeled in Figure 3.14. So the trellis of Figure 3.14 gives all nine information symbols of each of the 2^K paths. And since a decision has already been made on the first bit, the trellis may be shifted left again when the next two symbols are received without inserting an error due to memory truncation. After the comparisons at time 9, the shift, and the comparisons at time 10, assume the trellis reduces to that in Figure 3.15.



TRELLIS AT TIME 10

FIGURE 3.15

Now before the trellis can be shifted again, a decision must be made on the second information bit. One method is to choose the bit which belongs to the path with the largest log-likelihood function up to the present time. In Figure 3.15 the path with the largest log-likelihood function is labeled ML. Thus if this method is used, the second bit

would be decoded as a "1". This same procedure is then repeated until all the information symbols have been shifted through the shift register.

3.4 Distance Differences of the Surviving Paths

At each step, besides storing the unmerged portions of the 2^{K-1} possible paths, a Viterbi decoder must also store the log-likelihood functions of these paths, or for the binary symmetric channel the Hamming distance between each retained path and the received sequence must be stored. We will now show that for a rate R= 1/v (in bits/channel symbol) constraint length K convolutional code the difference in the Hamming distances (with respect to the received sequence) of any two of these retained paths is less than or equal to v(K-1).

Let $\underline{a}$ be the minimum Hamming distance path at time j

$$\underline{a} = a_1 \cdots a_i a_{i+1} a_{i+2} \cdots a_{j-(K-1)} a_{j-(K-1)+1} \cdots a_j \ .$$

And let $\underline{a}_x$ be any other surviving path at time j which is merged with $\underline{a}$ up to time i (possibly time 0) where they diverge. Since they are both surviving paths, they cannot have the same last K-1 bits. Thus $\underline{a}_x$ is of the form

$$\underline{a}_x = a_1 \cdots a_i \bar{a}_{i+1} a'_{i+2} \cdots a'_{j-(K-1)} \overline{\overline{a_{j-(K-1)+1} \cdots a_j}} \ ,$$

where $\bar{a}_{i+1}$ is the complement of a_{i+1}, a'_k is any binary symbol (possibly the same as a_k as long as no K-1 consecutive bits of the two paths are identical), and the double overbar means those bits cannot all be the same as the last K-1 bits of $\underline{a}$. Now consider another path $\underline{a}_y$ which agrees with path $\underline{a}$ up to time j-(K-1) and whose last K-1 bits are identical to the last K-1 bits of path $\underline{a}_x$. Then $\underline{a}_y$ is equal to

$$\underline{a}_y = a_1 \cdots a_i a_{i+1} a_{i+2} \cdots a_{j-(K-1)} \overline{\overline{a_{j-(K-1)+1} \cdots a_j}} \ .$$

Since path $\underline{a}_x$ is a survivor in the Viterbi decoding algorithm at time j, it has a Hamming distance less than or equal to that of any other path with the same last K-1 bits at time j. Thus

$$D(\underline{a}_x) \leqq D(\underline{a}_y) \quad , \tag{3.1}$$

where $D(\underline{a}_x)$ means the Hamming distance between the received sequence and the path corresponding to the input sequence $\underline{a}_x$. If the last K-1 bits of $\underline{a}_y$ are denoted by $\underline{a}_{yL}$ and the last K-1 bits of $\underline{a}$ by $\underline{a}_L$, then

$$D(\underline{a}_x) - D(\underline{a}) \leqq D(\underline{a}_y) - D(\underline{a}) = D(\underline{a}_{yL}) - D(\underline{a}_L) \quad , \tag{3.2}$$

and since there are v channel symbols per branch, $\underline{a}_{yL}$ represents a path with v(K-1) channel symbols. Thus using (3.2),

$$D(\underline{a}_x) - D(\underline{a}) \leqq v(K-1).$$

Thus for the BSC, the Hamming distances of the 2^{K-1} surviving paths cannot differ by more than v(K-1). This result can also be generalized to other memoryless channels.

CHAPTER 4

SELECTING GOOD CONVOLUTIONAL CODES

In Chapter 2, time-variable convolutional codes were investigated, and it was shown that for a sufficiently large constraint length these codes can achieve any arbitrarily small word (and hence bit) probability of error. This chapter is concerned with the more practical problem of selecting good rate 1/v constraint length K fixed binary convolutional codes. First the class of codes is reduced by considering only one code in each set of equivalent codes and by excluding the codes which exhibit catastrophic error propagation [Massey (1968)]. Some of the consequences of catastrophic error propagation are also investigated, and a new necessary and sufficient condition for catastrophic error propagation is derived.

Then the codes are compared on the basis of their minimum free distance, and Heller's (1968) upper bound on this distance is given and improved. Next the linear finite-state machine approach of Viterbi (1969) is described and used to obtain upper bounds on the bit probability of error. Finally, the Viterbi decoding algorithm is used in a search for codes with a small bit probability of error and the results of this search are tabulated.

4.1 Equivalent Codes

From Chapter 2 we know that a fixed binary convolutional code which is terminated after L information bits is a group code with an Lxv(L+K-1) generator matrix

$$G = \begin{bmatrix} \underline{G}_1 \underline{G}_2 \cdots \underline{G}_K & & & \\ & \underline{G}_1 \underline{G}_2 \cdots \underline{G}_K & & \\ & & \ddots & \\ & & & \underline{G}_1 \underline{G}_2 \cdots \underline{G}_K \end{bmatrix}, \qquad (4.1)$$

zero and that $\underline{G}_K$ is not zero. For if $\underline{G}_K$ were zero, the constraint length could be redefined as one less than the present value.

Now define the reciprocal of the subgenerator $\underline{g}_i$ by

$$\tilde{\underline{g}}_i = (g_{ik} \cdots g_{i2}\, g_{i1}) \quad . \tag{4.5}$$

Then two codes whose subgenerators are reciprocals of each other have identical weight spectrums. Because for any code word $\underline{u}\,G$ of one code, the reciprocal of this code word belongs to the other code and reciprocal code words have equal weight. In fact, this reciprocal code word is formed with the reciprocal of $\underline{u}$.

Another obvious way of obtaining an equivalent code is to change the ordering of the subgenerators. For example, consider the convolutional code with subgenerators $\underline{g}_1 = (1110)$ and $\underline{g}_2 = (1101)$ written $\begin{matrix}1110\\1101\end{matrix}$. And let ~ denote equivalence. Then some equivalent codes are

$$\begin{matrix}1110\\1101\end{matrix} \sim \begin{matrix}0111\\1011\end{matrix} \sim \begin{matrix}1110\\1011\end{matrix} \sim \begin{matrix}1011\\1110\end{matrix} \tag{4.6}$$

where the reciprocal, shifting, and ordering procedures have been used.

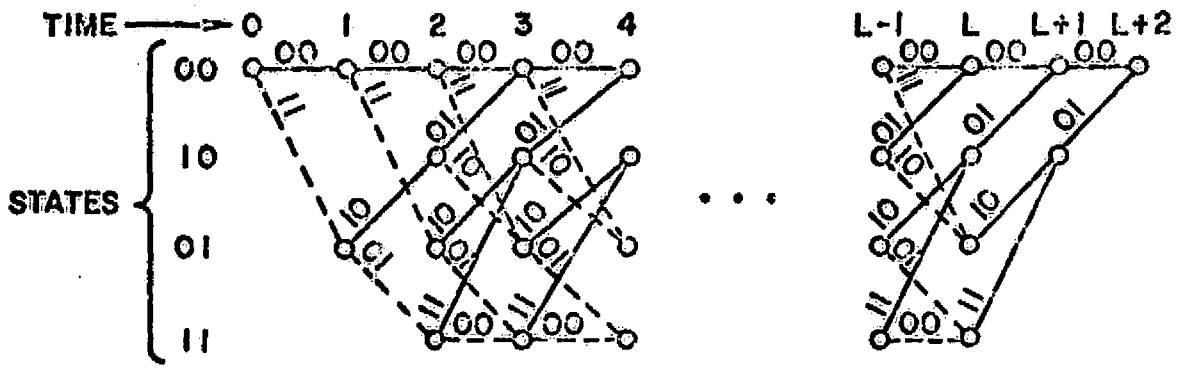
Thus in searching for the best code only one code in each equivalence class need be considered, and as noted, we may assume that the first component of each of its subgenerators is one.

4.2 Catastrophic Error Propagation

Assume that a convolutional encoder is used to transmit an infinitely long message over a binary-input binary-output channel. Then a decoder is said to exhibit catastrophic error progapation when a finite number of channel errors results in infinitely many decoding mistakes [Massey (1968)]. Massey and Sain (1968) have derived a necessary and sufficient condition for catastrophic error propagation of a

PRECEDING PAGE BLANK NOT FILMED

convolutional code of rate λ/v. For the class of rate $1/v$ convolutional codes considered here a code exhibits catastrophic error propagation if and only if the subgenerator polynomials contain a common factor. For example, the rate 1/2 constraint length 3 convolutional code described by the subgenerator polynominals $g_1(D) = D^2+1$ and $g_2(D) = D^2+D$ exhibits catastrophic error propagation since the subgenerator polynomials have a common factor of D+1. The trellis diagram of this code showing the encoder output symbols is shown in Figure 4.1.



TRELLIS DIAGRAM FOR A CODE WITH CATASTROPHIC ERROR PROPAGATION

FIGURE 4.1

Now if the all zero message is encoded and transmitted over a binary symmetric channel and the received sequence is

11 01 00 00 ... 00 11 01

then the maximum likelihood path is 1111...100. Thus six channel errors have caused an arbitrarily large number of bit decoding errors.

In general it is clear that catastrophic error propagation can occur if and only if there exists a zero weight path from some nonzero state back to itself. Using this fact the following theorem will be proved.

Theorem: If the encoder for a rate 1/v constraint length K fixed binary convolutional code is initially in any nonzero state and K-1 input symbols are shifted into the shift register, then all v(K-1) output symbols can be zero only if the code exhibits catastrophic error propagation. Conversely, if catastrophic error propagation occurs, then there exists at least one nonzero input state and one sequence of K-1 input symbols such that the corresponding v(K-1) outputs are zero.

First recall that we may assume that the first component of each subgenerator is a one and that the last component of all the subgenerators is not zero. So the two branches from each state have v-symbol output sequences which are complements of each other. Also the output symbols corresponding to the two branches entering any state cannot be identical. In particular, the shift register can never be taken from a nonzero state to the zero state by a zero weight path, because one of the two branches entering the zero state is the zero weight path from the zero state and the two entering branches cannot be identical.

Now assume that for some nonzero initial state and some input sequence of length K-1, the v(K-1) outputs are all zero. Denote the contents of the shift register at these K-1 times by the K-component vectors $\underline{A}_1, \ldots, \underline{A}_{K-1}$. Then since the shift register cannot be taken from a nonzero state to the zero state by a zero weight path, none of the $\underline{A}_i$ are all zero.

Consider the case where all the $\underline{A}_i$ are linearly independent. Then since the outputs are zero whenever the shift register contents are $\underline{A}_i$, $1 \leq i \leq K-1$, the outputs are also zero for all 2^{K-1} linear combinations of these $\underline{A}_i$. We also know that the two branches out of any state have output symbols which are complements of each other. So all the branches contain output symbols which are either all zeros or

all ones. Thus all the subgenerators are identical. And by Massey and Sain's condition this means that the code exhibits catastrophic error propagation.

If the $\underline{A}_i$ are not linearly independent, there exists at least one linear combination of the $\underline{A}_i$ which equals zero. Let J be the maximum value of i in any such linear combination. Then this linear relationship can be written as

$$\underline{A}_J = \mathcal{L}(\underline{A}_1, \ldots, \underline{A}_{J-1}) \quad . \tag{4.7}$$

Now let us examine $\underline{A}_{J+1}$. Since $\underline{A}_{J+1}$ represents the contents of a K-stage shift register at time J+1, the last K-1 components of $\underline{A}_{J+1}$ are equal to the first K-1 components of $\underline{A}_J$. Also given the last K-1 input symbols, there is at most one value for the first component such that the v output symbols are zero. Consider the vector $\mathcal{L}(\underline{A}_2, \ldots, \underline{A}_J)$. Each term of the argument has its last K-1 components equal to the first K-1 components of the corresponding terms of (4.7), and consequently, the last K-1 components of the linear combination are equal to the first K-1 components of $\underline{A}_J$ of (4.7). Also since this is a linear combination of $\underline{A}_1, \ldots, \underline{A}_{J-1}$, it produces zero output symbols. So if the first Jv output symbols are zero, then there is an input symbol such that the next v outputs are also zero. And this input symbol is just the first component of the vector $\underline{A}_{J+1}$, where

$$\underline{A}_{J+1} = \mathcal{L}(\underline{A}_2, \ldots, \underline{A}_J) \quad . \tag{4.8}$$

By repeating this argument with

$$\underline{A}_{J+k} = \mathcal{L}(A_{1+k}, \ldots, A_{J-1+k}) \quad , \qquad k = 0, 1, 2, \ldots \tag{4.9}$$

we have that there exists an arbitrarily long input sequence which produces an all zero output. After at most 2^{K-1} input symbols have been shifted into the shift register, some state in the path (sequence

of states) specified by the last K-1 components of $\underline{A}_1, \ldots, \underline{A}_{J-1}$, $\underline{A}_J, \ldots, \underline{A}_{J+k}, \ldots$ where $\underline{A}_{J+k}$ is given by (4.9), must be repeated. So there is a zero weight path from some nonzero state back to itself. Thus the code exhibits catastrophic error propagation.

The converse follows easily. For, if catastrophic error propagation occurs, there exists a zero weight path from some nonzero state back to itself and by repeating this path an arbitrarily long zero weight path may be formed. Thus the theorem is proved.

4.3 Upper Bounds on the Minimum Free Distance

Before proceeding any further in the search for good convolutional codes, some criterion of goodness must be chosen. The best criterion would be to choose the code with the smallest probability of error. However, this is difficult, if not impossible, to do. Another common criterion is to choose the code with the largest minimum distance (between code words) over a constraint length. But since most of the paths compared by the Viterbi decoding algorithm are unmerged for more than K branches, this is a poor criterion. A better choice is to compare the codes on the basis of their unrestricted minimum distance between code words. This unrestricted minimum distance is called the minimum free distance and is the only distance used here.

Now let us find an upper bound on this minimum distance. As noted previously, a terminated binary convolutional code with L information bits is a group, or linear, code with an $L \times v(L+K-1)$ dimensional generator matrix. So instead of comparing the codes on the basis of their minimum distance between code words, we may equivalently compare the minimum weights of their nonzero code words. Also it is well known [Peterson (1961)] that if all 2^L code words are arranged as rows of a matrix, then any column which is not all zero contains

half ones and half zeros. Thus the total weight of the 2^L code words is bounded by

$$\text{total weight} \leqq 2^{L-1} v(L+K-1) \quad . \tag{4.10}$$

Since there are 2^L-1 nonzero code words and their minimum weight must be less than or equal to their average weight, the minimum distance satisfies

$$d_{min} \leqq \frac{2^L}{2^L-1} \frac{v(L+K-1)}{2} \quad . \tag{4.11}$$

This bound is valid for any L and it is also valid for nonzero sequences of length $\ell < L$. Therefore (4.11) specifies a sequence of bounds, and the tightest bound can be obtained by minimizing on ℓ. Thus for arbitrarily large L

$$d_{min} \leqq \min_{1 \leqq \ell} \left[\frac{2^\ell}{2^\ell-1} \frac{v(\ell+K-1)}{2} \right] = d_H \quad , \tag{4.12}$$

where [x] denotes the largest integer less than or equal to x. This bound was derived by Heller (1968) and is denoted d_H. Heller has also shown that the minimum in (4.12) occurs at a value of ℓ that varies as log K and that for large K, (4.12) reduces to

$$d_{min} \lesssim \frac{Kv}{2} \quad . \tag{4.13}$$

If d_H is odd, then the bound of (4.12) can sometimes be improved by using the fact that the weight of any binary group code with odd minimum weight is increased by one when a parity-check digit is added.* For assume (4.12) is optimized with an input sequence of length $\ell = h$ and that d_H is odd. Then if a parity-check digit is added to the code with h input symbols, the resulting code is a group code with an $h \times [v(h+K-1)+1]$ dimensional generator matrix. The minimum distance of this extended code can be bounded with a Plotkin bound [Peterson (1961)] similar to (4.12) yielding

---

*See also: Layland, J. and R. McEliece, "An Upper Bound on the Free Distance of a Tree Code," Jet Propulsion Laboratory, Space Programs Summary, 37-62, Vol. III.

$$d_{\text{ext. min}} \leqq \left[\frac{2^h}{2^h - 1} \quad \frac{v(h+K-1)+1}{2} \right] = d_E \quad , \tag{4.14}$$

and so if $d_E < d_H + 1$, no code can achieve $d_{min} = d_H$. Thus if d_H is odd and $d_E < d_H + 1$, then

$$d_{min} \leqq d_H - 1 \quad . \tag{4.15}$$

Another similar procedure for improving (4.12) when d_H is odd is based on the fact that in a group code either all the code words have even weight or half the words have even weight and half have odd weight [Peterson (1961)]. So if (4.12) is optimized with ℓ=h and d_H is odd, then the total weight of the 2^h code words is bounded by

$$\frac{2^h}{2} d_H + \left(\frac{2^h}{2} - 1 \right)(d_H + 1) \leqq \text{total weight} \quad . \tag{4.16}$$

And using (4.10) with an input sequence of length h yields

$$\frac{2^h}{2} d_H + \left(\frac{2^h}{2} - 1 \right)(d_H + 1) \leqq \frac{2^h}{2} v(h+K-1) \tag{4.17}$$

or

$$d_H \leqq \frac{2^{h-1}}{2^h - 1} \left\{ v(h+K-1) - 1 + 2^{1-h} \right\} \quad . \tag{4.18}$$

Thus if d_H is odd and (4.18) is not satisfied, then d_H may be decreased by one, as in (4.15).

In Table 4.1 Heller's bound and this improved bound are tabulated for rates 1/2 and 1/3 and constraint lengths up to 20. The results of a computer search are also shown. So from this table it can be seen that this improved upper bound is very tight at small constraint lengths. It is also believed to be equally accurate at large constraint lengths.

TABLE 4.1

UPPER BOUNDS ON THE MINIMUM FREE DISTANCE

Constraint Length	Rate = 1/2			Rate = 1/3		
	Heller's Upper Bound	Improved Upper Bound	d_{min} Achieved	Heller's Upper Bound	Improved Upper Bound	d_{min} Achieved
3	5	5	5	8	8	8
4	6	6	6	10	10	10
5	8	8	8*	12	12	12
6	9	8	8	13	13	13
7	10	10	10	15	15	14†
8	11	11	10†	17	16	16
9	12	12	12	18	18	
10	13	13		20	20	
11	14	14		22	22	
12	16	16		24	24	
13	17	16		25	25	
14	18	18		27	26	
15	19	18		28	28	
16	20	20		30	30	
17	21	20		32	32	
18	22	22		33	33	
19	23	23		35	34	
20	24	24		36	36	

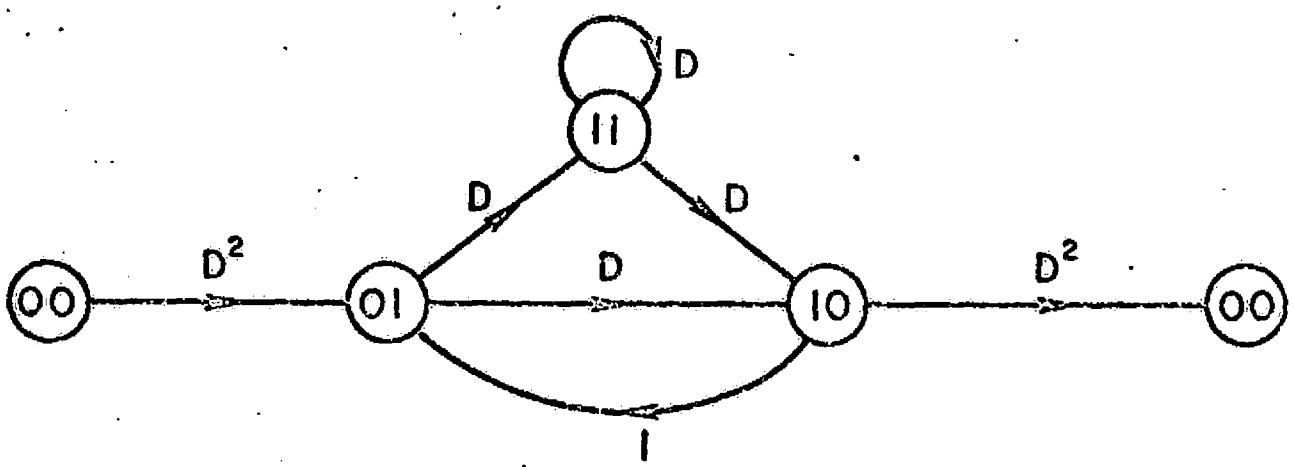
*The best noncatastrophic code has $d_{min} = 7$.

†There may be a catastrophic error propagating code with a larger d_{min}.

4.4 The State Diagram Approach

Suppose that we are given a binary-input output symmetric channel. Then since a terminated convolutional code is a group code, we may assume that the all zero message is transmitted. Now if we use a Viterbi decoder, an error occurs whenever the log-likelihood function of one of the retained paths which remerges with the all zero state at some time j is greater than that of the all zeros path up to time j. For small constraint lengths the Hamming weight of all such paths can be determined by the following procedure [Viterbi (1969)].

Consider a state diagram with the all zero state split open. Label the branches with D^i, where i is the Hamming weight of that branch. For example, the K=3, R=1/2 convolutional code of Figure 3.1 with subgenerator polynomials $g_1(D) = D^2+D+1$ and $g_2(D) = D^2+1$ would have a modified state diagram as in Figure 4.2.



MODIFIED STATE DIAGRAM FOR THE CODE OF FIGURE 3.1

FIGURE 4.2

The over-all transfer function of this directed graph can then be obtained.

$$T(D) = \frac{D^5}{1-2D} = D^5 + 2D^6 + 4D^7 + \ldots + 2^k D^{k+5} + \ldots \qquad (4.19)$$

Thus there is one incorrect path of weight 5, 2 of weight 6, etc.

Using (4.19), we may then find an upper bound on the probability of error at any step in the Viterbi decoding algorithm. Assume that antipodal PSK modulation is used on an additive white Gaussian noise channel with one sided spectral density N_o. And let the positive and negative pulses both have energy E_b. Then the probability of error in comparing an incorrect path which differs in k symbols from the correct path is given by [Wozencraft and Jacobs (1965)]

$$P = Q\left(\sqrt{\frac{2kE_b}{vN_o}}\right) \quad , \qquad (4.20)$$

where

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-\frac{\alpha^2}{2}} d\alpha \quad , \qquad (4.21)$$

and $v = \frac{1}{R}$ is the number of channel symbols per information bit. In Appendix B the following useful inequality is proved.

$$Q(\sqrt{x+y}) \leqq e^{-y/2}\ Q(\sqrt{x}) \quad , \quad x \geqq 0, \quad y \geqq 0 \qquad (4.22)$$

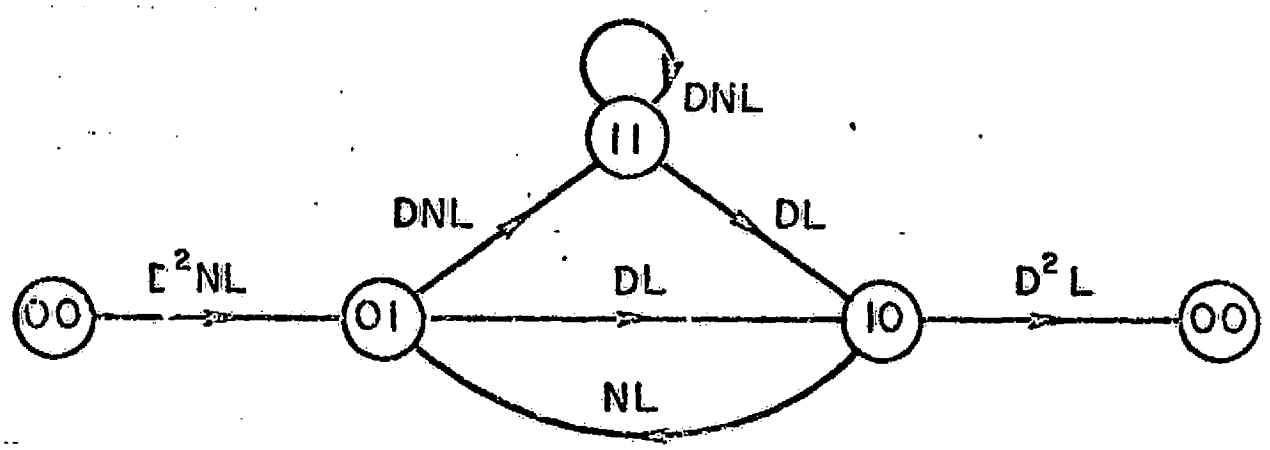
Using (4.19), (4.20), and (4.22), the probability of error at any step can be upper bounded with a union bound yielding

$$P_s \leq Q\left(\sqrt{5\frac{E_b}{N_o}}\right) + 2Q\left(\sqrt{6\frac{E_b}{N_o}}\right) + 4Q\left(\sqrt{7\frac{E_b}{N_o}}\right) + \ldots$$

$$\leq Q\left(\sqrt{5\frac{E_b}{N_o}}\right)\left\{1 + 2e^{-\frac{E_b}{2N_o}} + 4e^{-2\frac{E_b}{2N_o}} + \ldots\right\}$$

$$= \frac{\left(Q\sqrt{5\frac{E_b}{N_o}}\right)}{1-2e^{-\frac{E_b}{2N_o}}} \qquad (4.23)$$

To determine the bit probability of error we need to know the number of bit errors resulting from an incorrect path decision. These may be computed by adding a factor of N to each branch of the modified state diagram which corresponds to a "1" input bit. We can also find the lengths of these paths by adding a factor of L to each branch. With these changes Figure 4.2 becomes



MODIFICATION OF FIGURE 4.2

FIGURE 4.3

Now the transfer function is

$$T(D,N,L) = \frac{D^5 N L^3}{1-DNL-DNL^2}$$

$$= D^5NL^3 + D^6N^2(L^4+L^5) + D^7N^3(L^5+2L^6+L^7) + \ldots \qquad (4.24)$$

There is one path of weight 5, and it is three branches long and results in one bit error. There are two paths of weight 6, one of length 4 and one of length 5, and both result in two bit errors, etc. Differentiation of (4.24) with respect to N followed by setting N and L equal to one weights the incorrect paths by their number of bit errors. Then using (4.20), (4.22), and (4.24) and a union bound yields the following bound on the bit probability of error.

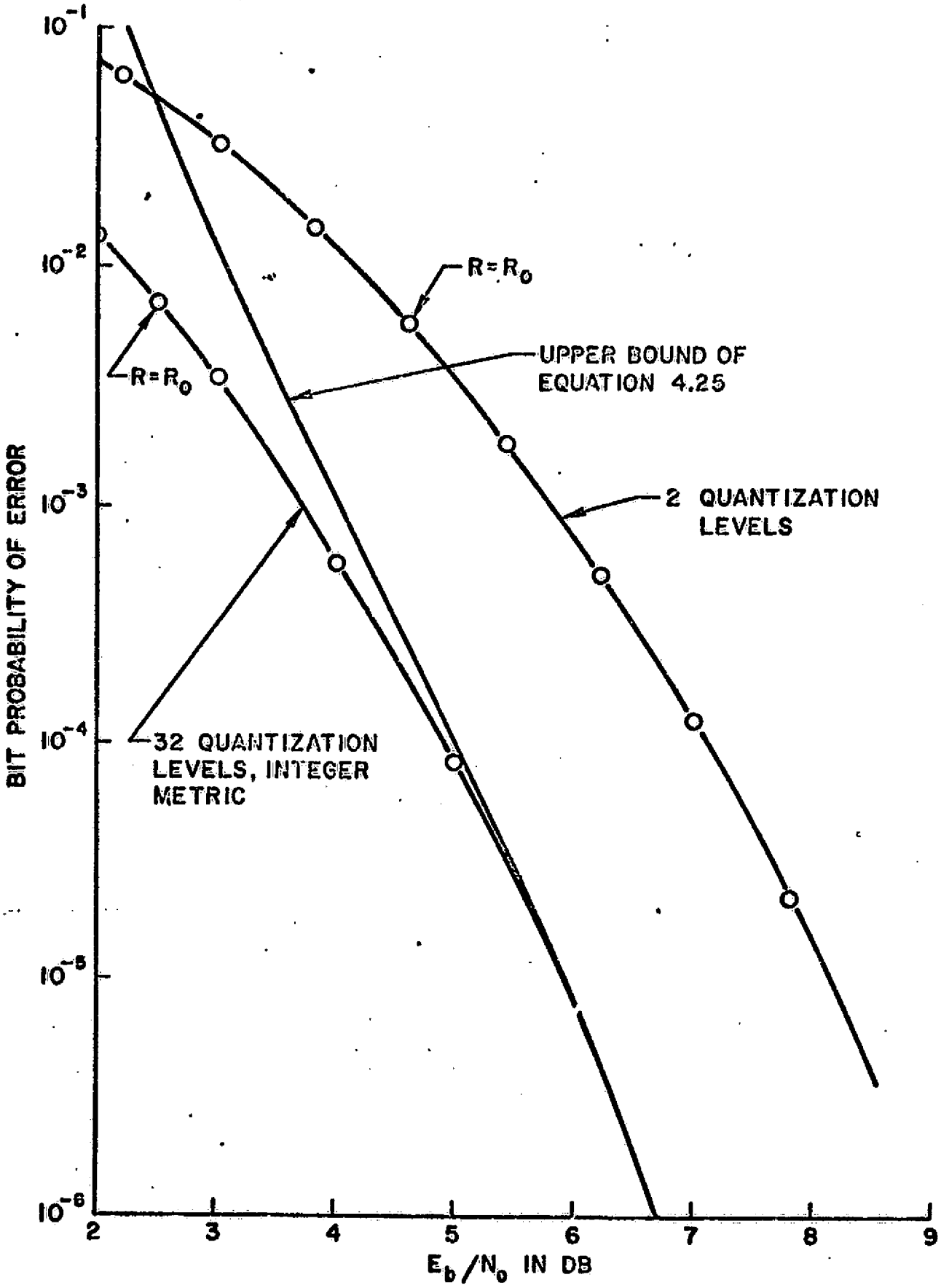
$$P_{E\,bit} \leq Q\left(\sqrt{5\frac{E_b}{N_o}}\right) + 2\cdot 2\,Q\left(\sqrt{6\frac{E_b}{N_o}}\right) + 3\cdot 4\,Q\left(\sqrt{7\frac{E_b}{N_o}}\right) + \ldots$$

$$\leq Q\left(\sqrt{5\frac{E_b}{N_o}}\right)\left\{1 + 2\cdot 2\,e^{-\frac{E_b}{2N_o}} + 3\cdot 4e^{-2\frac{E_b}{2N_o}} + \ldots\right\}$$

$$= \frac{Q\left(\sqrt{5\frac{E_b}{N_o}}\right)}{\left(1-2e^{-E_b/2N_o}\right)^2} \qquad (4.25)$$

Since (4.23) and (4.25) are based on a union bound, they are only useful for reasonably high energy-to-noise ratios. In Figure 4.4 the bound of (4.25) is compared with the results of a simulation of this system with 32 output quantization levels. It was found that very little improvement can be obtained by using more than 32 quantization levels. The results for two quantization levels (i.e., a BSC) are also shown.

This state diagram approach is also helpful in understanding catastrophic error propagating codes. Previously, a code was shown to exhibit catastrophic error propagation if and only if there was a zero weight path from some nonzero state back to itself. In terms of the modified state diagram this zero weight path is just a closed loop with zero weight.



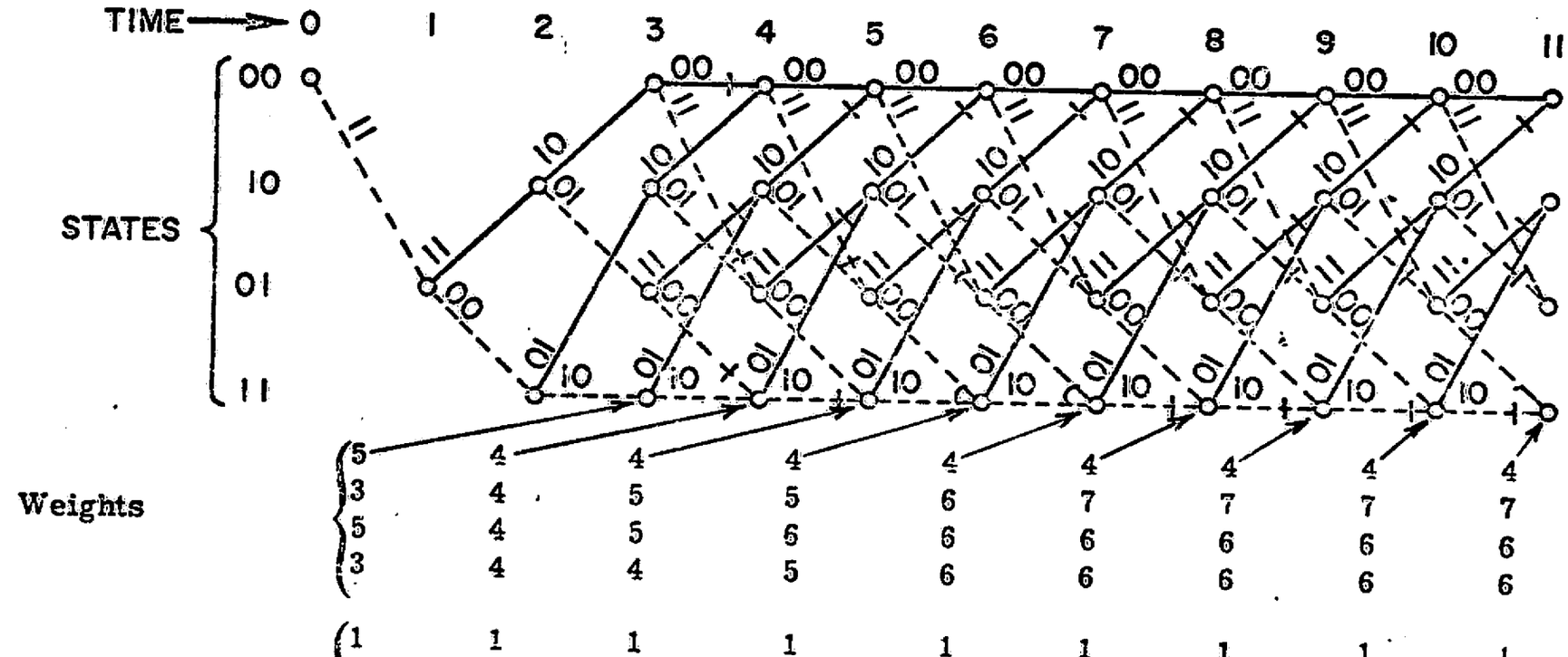
BIT PROBABILITY OF ERROR VS. ENERGY-TO-NOISE RATIO FOR A CONVOLUTIONAL CODE WITH SUBGENERATORS $\begin{matrix}111\\101\end{matrix}$

FIGURE 4.4

4.5 The Search for Good Convolutional Codes

Although the state diagram approach is helpful in finding good codes for small constraint lengths ($K \leq 4$), it is extremely difficult to find the transfer function for an arbitrary code of higher constraint length. So first let us just find the minimum weight of any path which diverges from the all zero path at time zero and remerges at any later time. Any such path will be called an adversary, and clearly, the minimum weight of these adversaries is just the minimum free distance of the code.

Suppose that the all zero information sequence is encoded and transmitted across a noiseless channel and that a Viterbi decoder (or equivalently a maximum-likelihood decoder) is used to decode the message. But restrict the first decoded bit to be a one. Then at any time $j \geq K$, this Viterbi decoder will retain 2^{K-1} paths. And each path will be the minimum weight path which diverges from the all zero state at time zero and terminates in a particular state at time j. Thus for large enough j the minimum free distance of the code is equal to the weight of the path which terminates at the all zero state. We may also keep track of the number of such minimum weight paths which terminate in each state at each time and the number of bit errors that these paths represent. For example, consider the rate 1/2 constraint length 3 code $\begin{matrix}111\\110\end{matrix}$. The trellis diagram for this code is shown in Figure 4.5. As in Chapter 3, the dotted lines represent "1" input bits and the solid lines represent "0" input bits. The results of the comparisons are indicated by a slash or an arc. The slash indicates that branch may be cut, and the arc means the two paths in that comparison had equal weights; so both paths are retained. By looking at the all zero state at time 11, we see that the minimum free distance of this code is 4 and that there is only one minimum weight adversary and it represents two bit errors.



Weights	5	4	4	4	4	4	4	4	4
	3	4	5	5	6	7	7	7	7
	5	4	5	6	6	6	6	6	6
	3	4	4	5	6	6	6	6	6
Number of paths	1	1	1	1	1	1	1	1	1
	1	1	1	1	2	4	2	1	1
	1	1	1	2	2	1	1	1	1
	1	1	1	2	4	2	1	1	1
Number of bit errors	1	2	2	2	2	2	2	2	2
	2	3	4	4	10	20	9	4	4
	2	3	4	8	8	3	3	3	3
	3	4	4	10	20	9	4	4	4

TRELLIS FOR THE $\frac{111}{110}$ CODE SHOWING THE NUMBER OF PATHS AND THE NUMBER OF BIT ERRORS THESE PATHS REPRESENT AT EACH TIME

FIGURE 4.5

In this example, the all zero state was observed at time 11, but the results would have been the same if the all zero state was observed at any time greater than or equal to 4. It is only necessary that all of the minimum weight adversaries be remerged with the all zero state. This can be guaranteed if the weight of the path which terminates at the all zero state is less than or equal to the weights of the paths which terminate at the nonzero states. For previously, it was shown that we may assume, without loss of generality, that any branch entering the all zero state has nonzero weight, and thus the total weight of any path which remerges with the all zero state at a later time is greater than the minimum weight. However, if the code exhibits catastrophic error propagation, this condition may never be satisfied.

But since there are 2^{K-1} states, there must be at least one nonzero repeated state in any adversary of length $2^{K-1}+1$ branches. In terms of the modified state diagram this means there is at least one closed loop in the path. So there is another path, formed by omitting these closed loops, which has weight less than or equal to (strictly less than if the code does not exhibit catastrophic error propagation) the weight of the original path. Thus every code has at least one minimum weight adversary with length less than or equal to 2^{K-1} branches. And if the code does not exhibit catastrophic error propagation, every minimum weight adversary has length less than or equal to 2^{K-1} branches. So it is never necessary to consider more than 2^{K-1} branches.

Another stopping rule is based on the theorem at the beginning of this chapter. Let d_o be the improved upper bound on the minimum free distance given previously. And note that since we may assume the first component of every subgenerator is a one, the first branch of every path which begins with a "1" input bit has weight v. Now if

the code does not exhibit catastrophic error propagation and if any path retained by the Viterbi decoder does not remerge with the all zero path, then the first branch of this path has weight v, and from the theorem each successive sequence of K-1 branches has a weight of at least one. Thus this path must have a weight of at least d_o after $1+(K-1)(d_o-v)$ branches. If a retained path does remerge with the all zero state, it clearly has weight greater than d_o. So the condition for the first stopping rule is satisfied. Also if after $1+(K-1)(d_o-v)$ branches one of the retained paths which terminates at a nonzero state has weight less than that of the path terminating in the all zero state, then the code exhibits catastrophic error propagation.

In a complete search of all codes of a particular rate and constraint length, many codes can be eliminated quickly without using this procedure. Thus the required computer time can be reduced by using this procedure only after verifying that the paths corresponding to input sequences of 10...0 and 110...0 have weight greater than some lower bound. Alsc codes whose subgenerator polynomials contain a common factor of D+1 (i.e., the subgenerators all contain an even number of ones) can be quickly eliminated.

Using these techniques, a complete search was made of all rate 1/2 and 1/3 codes of constraint lengths up to 9 and 8, respectively. And all the different, noncatastrophic codes with the maximum, minimum free distance for that rate and constraint length were tabulated. Then all those codes whose adversaries contained more than the minimum number of bit errors were eliminated. If more than one code remained, only those codes with the smallest number of bit errors among their adversaries at the minimum distance plus one were retained. If more than one code still remained, the adversaries at the minimum distance plus two etc... were examined until only one code

remained. If this surviving code were used on a binary symmetric channel with a low crossover probability or with an antipodal PSK modulator on an additive white Gaussian noise channel with symmetric output quantization at high energy-to-noise ratios, then it would have a bit probability of error less than or equal to that of any other code of that rate and constraint length. In this sense this code is the best code for that rate and constraint length. These results are tabulated in Tables 4.2 to 4.14.

TABLE 4.2

BEST RATE 1/2 CONSTRAINT LENGTH 3 CODE

Code $\begin{matrix}111\\101\end{matrix}$

Minimum free distance = 5

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
5	1	1
6	2	4
7	4	12
8	8	32
9	16	80
10	32	192
11	64	448
12	128	1024

TABLE 4.3

BEST RATE 1/2 CONSTRAINT LENGTH 4 CODE

Code 1111
1101

Minimum free distance = 6

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
6	1	2
7	3	7
8	5	18
9	11	49
10	25	130
11	55	333
12	121	836
13	267	2069

TABLE 4.4

BEST RATE 1/2 CONSTRAINT LENGTH 5 CODE

Code 11101
10011

Minimum free distance = 7

Weignt	Number of Adversaries	Number of Bit Errors in the Adversaries
7	2	4
8	3	12
9	4	20
10	16	72
11	37	225
12	68	500
13	176	1324
14	432	3680

TABLE 4.5

BEST RATE 1/2 CONSTRAINT LENGTH 6 CODE

Code 111101
101011

Minimum free distance = 8

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
8	1	2
9	8	36
10	7	32
11	12	62
12	48	332
13	95	701
14	281	2342
15	605	5503

TABLE 4.6

BEST RATE 1/2 CONSTRAINT LENGTH 7 CODE

Code 1111001
1011011

Minimum free distance = 10

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
10	11	36
11	0	0
12	38	211
13	0	0
14	193	1404
15	0	0
16	1331	11633
17	0	0

TABLE 4.7

BEST RATE 1/2 CONSTRAINT LENGTH 8 CODE

Code 11111001
10100111

Minimum free distance = 10

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
10	1	2
11	6	22
12	12	60
13	26	148
14	52	340
15	132	1008
16	317	2642
17	730	6748

TABLE 4.8

BEST RATE 1/2 CONSTRAINT LENGTH 9 CODE

Code 111101011
101110001

Minimum free distance = 12

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
12	11	33
13	0	0
14	50	281
15	0	0
16	286	2179
17	0	0
18	1630	15035
19	0	0

TABLE 4.9

BEST RATE 1/3 CONSTRAINT LENGTH 3 CODE

111
Code 111
101

Minimum free distance = 8

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
8	2	3
9	0	0
10	5	15
11	0	0
12	13	58
13	0	0
14	34	201
15	0	0

TABLE 4.10

BEST RATE 1/3 CONSTRAINT LENGTH 4 CODE

1111
Code 1101
1011

Minimum free distance = 10

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
10	3	6
11	0	0
12	2	6
13	0	0
14	15	58
15	0	0
16	24	118
17	0	0

TABLE 4.11

BEST RATE 1/3 CONSTRAINT LENGTH 5 CODE

Code 11111
11011
10101

Minimum free distance = 12

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
12	5	12
13	0	0
14	3	12
15	0	0
16	13	56
17	0	0
18	62	320
19	0	0

TABLE 4.12

BEST RATE 1/3 CONSTRAINT LENGTH 6 CODE

Code 111101
101011
100111

Minimum free distance = 13

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
13	1	1
14	3	8
15	6	26
16	4	20
17	5	19
18	12	62
19	14	86
20	30	204

TABLE 4.13

BEST RATE 1/3 CONSTRAINT LENGTH 7 CODE

1111001
Code 1100101
1011011

Minimum free distance = 14

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
14	1	1
15	0	0
16	7	20
17	0	0
18	11	53
19	0	0
20	35	184
21	0	0

TABLE 4.14

BEST RATE 1/3 CONSTRAINT LENGTH 8 CODE

11110111
Code 11011001
10010101

Minimum free distance = 16

Weight	Number of Adversaries	Number of Bit Errors in the Adversaries
16	1	1
17	0	0
18	8	24
19	0	0
20	24	113
2.	0	0
22	51	287
23	0	0

CHAPTER 5

COMPUTER SIMULATION

Although the theoretical results of the previous chapters are helpful in evaluating the performance of digital communication systems which use convolutional codes, these results are primarily for very large or very small constraint lengths or large energy-to-noise ratios. This chapter reports the results of a digital computer simulation of a practical convolutional coding system and discusses some of the engineering tradeoffs which must be considered in realizing such a system.

5.1 System Description

The system investigated consists of a fixed binary rate 1/v convolutional encoder, a modulator, a channel, a demodulator, and a Viterbi decoder. The encoder maps the input data, which is assumed to be a sequence of equally probable and mutually independent binary symbols, into a binary code sequence. Then for each symbol of the code sequence the modulator delivers one of two antipodal elementary waveforms of duration T_b/v seconds for transmission over the channel. So every T_b seconds one bit of information is transmitted.

Now assume that in the absence of noise the received signal is just an attenuated version of the transmitted sequence of waveforms. In particular, let thre received elementary waveforms be $\pm \sqrt{2P}$ cos $(\omega_o t + \theta)$, where P is the average received power, ω_o is the carrier frequency, and θ is the constant known phase. The channel is also assumed to contribute zero mean additive white Gaussian noise with a one-sided power spectral density of N_o watts/cps. This total received signal then enters a demodulator which is composed of a filter matched to the elementary waveform, a sample-and-hold circuit, and

an amplitude quantizer. By an appropriate choice of the gain of this filter, the output of the sample-and-hold circuit at time kT_b/v can be written as

$$u_k = x_k\sqrt{\frac{2PT_b}{vN_o}} + n_k = x_k\sqrt{\frac{2E_b}{vN_o}} + n_k, \qquad (5.1)$$

where x_k equals + 1 if the k^{th} code symbol is 0 and equals - 1 otherwise, E_b is the energy per information bit, and the $\{n_k\}$ are statistically independent zero mean unit variance Gaussian random variables. So with (5.1) as an input, a quantizer must be chosen which minimizes the probability of error for that energy-to-noise ratio. In general, this can be done by choosing an arbitrarily large number of quantization levels, but increasing the number of quantization levels also increases the memory requirements of the system. The procedure used in selecting a reasonable quantizer for the computer simulation is discussed later in this chapter.

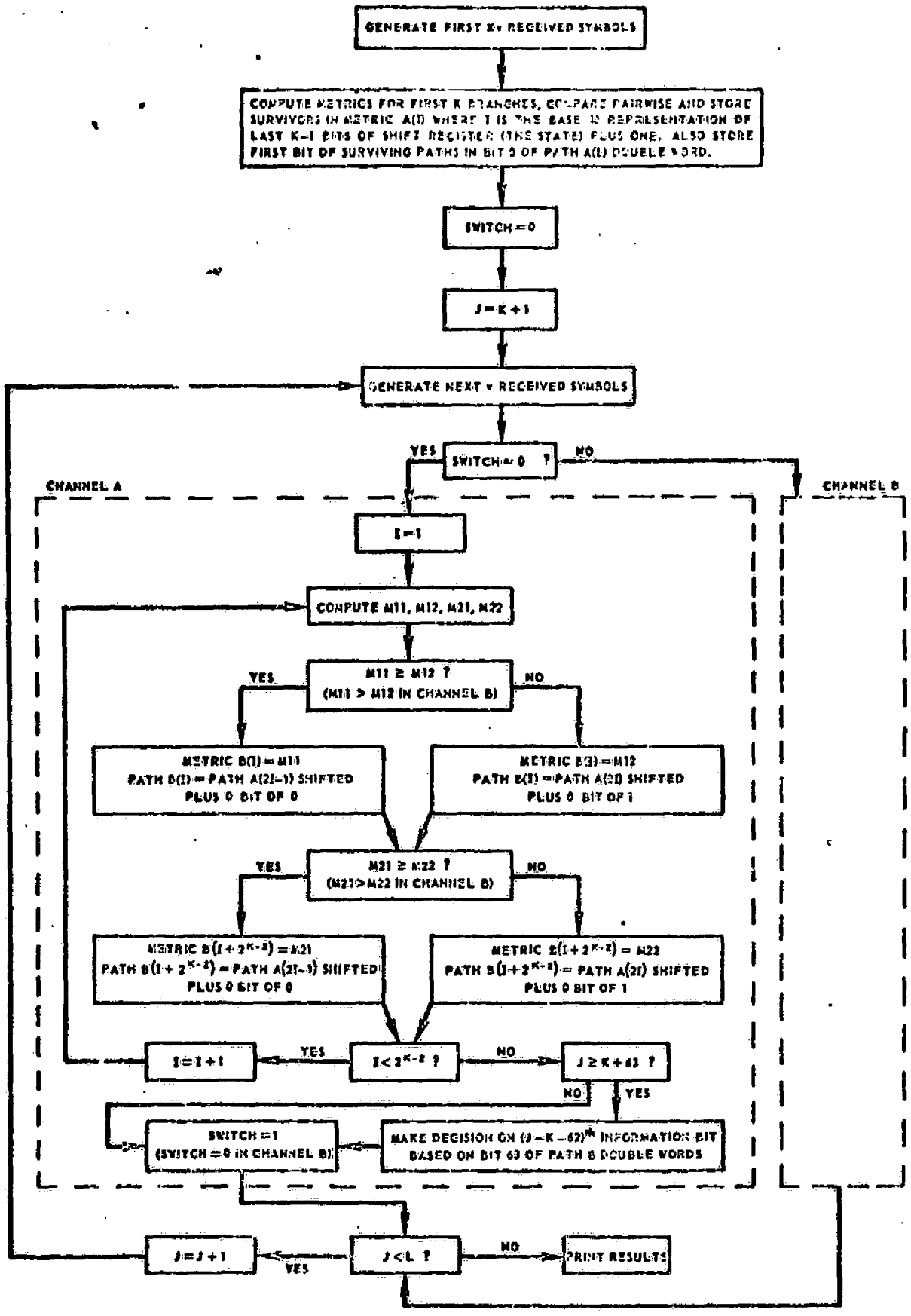
Finally, based on the quantizer outputs the Viterbi decoder produces an estimate of the data sequence. However, it is again necessary to make some approximations. First the memory of each of the 2^{K-1} retained paths must be truncated at some reasonable number, as discussed in Chapter 3. And secondly the number of significant figures to be retained in the log-likelihood functions must be determined. In fact, we may use some other means of comparison. For example, the quantized outputs could be assigned consecutive integer values instead of the usual log-likelihood values. This set of numbers which is assigned to the outputs in order to compare the various paths is called the metric of the decoder. In the above example an integer metric was used and previously log-likelihood metrics were used. Again the metric should be chosen to give a small probability of error with a reasonable complexity.

In the actual simulation it is helpful to consider the modulator, channel, and demodulator as a binary-input multiple output channel. Then all the details of the modulator, channel, and demodulator are contained in the transition probabilities of this new channel. To determine the output symbol of this new channel given the input symbol, the unit interval is segmented according to the transition probabilities. That is, the unit interval is divided into a number of segments equal to the number of possible outputs for the given input and each interval has a length equal to the corresponding transition probability. Then using the power residue method [I. B. M. Staff (1959)], a pseudo-random number is generated which is uniformily distributed over (0, 1). And the output symbol is chosen to correspond to the segment containing the pseudo-random number.

A block diagram of the simulation program is given in Figure 5.1.

5.2 Quantization

Clearly, transforming the output amplitude of the matched filter into a discrete number is an irreversible operation [Wozencraft and Jacobs (1965)] and, in general, increases the probability of error. As noted above, we would expect this increase to be small for a fine quantization, but a coarse quantization is also desirable because it reduces the memory requirements. So some engineering compromise must be reached, and this means some calculable measure of performance must be chosen. One good measure of performance is the exponent in the random ecding bounds of Chapter 2 [Wozencraft and Jacobs (1965), Jacobs (1967)]. In particular, since a typical operating rate would be near the computational cutoff rate (R_o), we will try to maximize the exponent at R_o. From Chapter 2 it can be seen that the upper and lower bounds coincide at rates greater than or equal to R_o and at R_o both exponents are equal to R_o. So the difference between



Mij=TOTAL METRIC GIVEN NEXT BIT IS j-1 AND STATE IS 2i-2+j

BLOCK DIAGRAM OF COMPUTER SIMULATION

FIGURE 5.1

the optimum (unquantized) R_o and the true R_o will be used as a measure of the degradation due to quantization.

For practical reasons let us restrict the quantizer to have uniformly spaced intervals, and since antipodal modulation is used, let these intervals be symmetric about zero. Then when the matched filter output is normalized as in (5.1), the optimum interval widths may be calculated. These interval widths vary with changing energy-to-noise ratios, but in the usual operating range ($1 < \frac{E_b}{v N_o} < 2.5$) they are essentially constant. In fact, experimentally it was found that the probability of error is very insensitive to changes in the quantization intervals. The optimum (i.e., based on the R_o measure discussed above) interval widths for several numbers of quantization levels are given in Table 5.1.

TABLE 5.1

OPTIMUM QUANTIZATION INTERVALS

Number of Quantization Levels	Interval Width
8	.58
16	.33
32	.18

With the exception of the 8 quantization levels, these are the interval widths used in the computer simulations. For 8 quantization levels an interval width of 0.5 was used to facilitate comparisons with the data of other researchers [Heller (1969)].

Another way of looking at this performance measure is as follows. Suppose we are given two systems which operate at different energy-to-noise ratios and have different quantizers but are otherwise identical. And assume that the energy-to-noise ratios are such that

both systems operate at rate R_o. Then based on the bounds of Chapter 2, we would expect both systems to have about the same probability of error for reasonably high constraint lengths. The optimum R_o can be calculated from (2.2) yielding

$$R_{o\ opt} = \frac{-\ln \frac{1}{2}\left(1+e^{-\frac{E_b}{vN_o}}\right)}{\ln 2} \text{ bits/symbol} \quad . \qquad (5.2)$$

Manipulating (5.2), we see that without quantization

$$\frac{E_b}{N_o} = -v \ln \left(2^{\frac{v-1}{v}} - 1\right) \qquad (5.3)$$

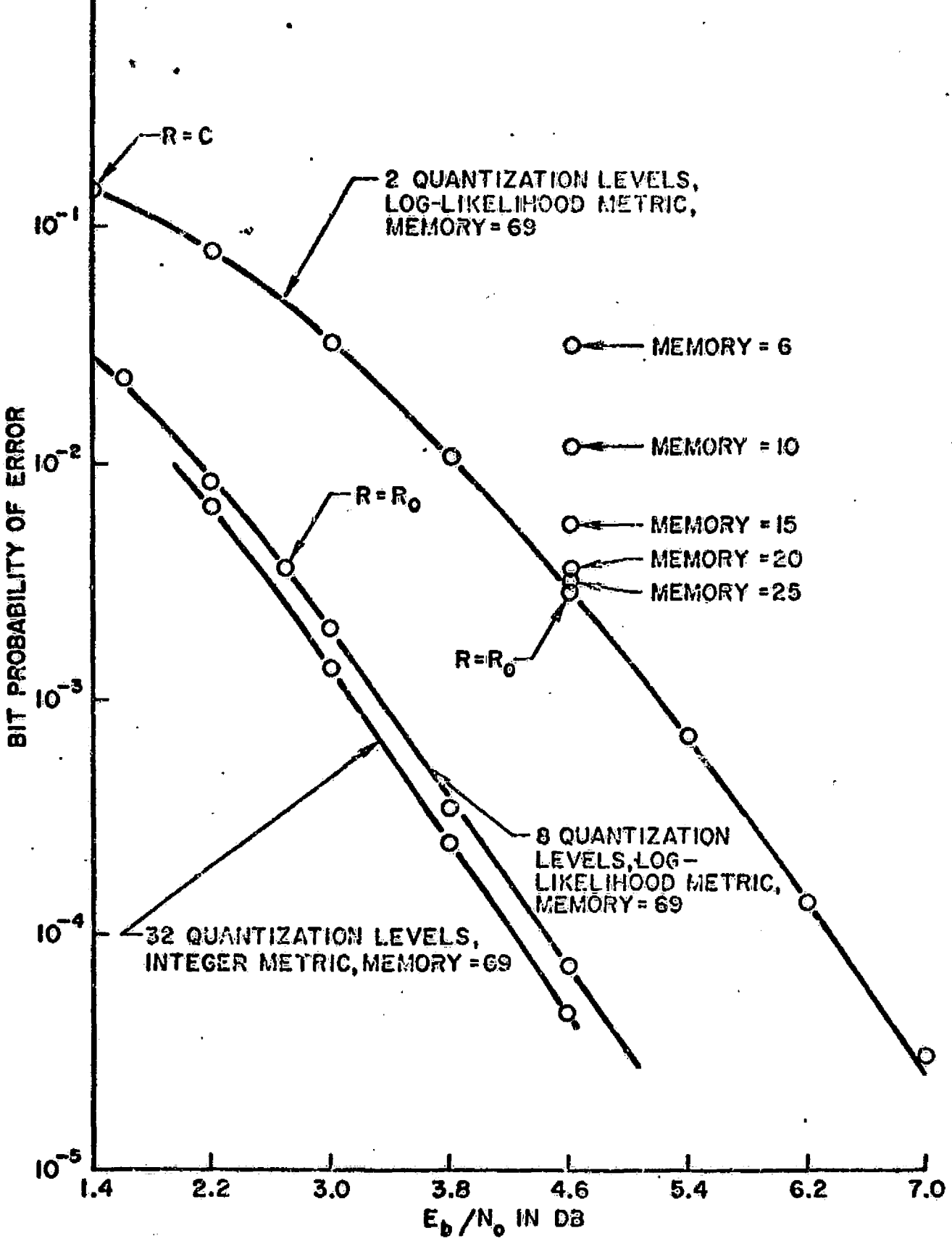
is required to operate at a rate $R = \frac{1}{v} = R_{o\ opt}$. For a rate 1/2 this means a 2.46 db energy-to-noise ratio is required.

Figure 5.2 shows the results of a computer simulation of a rate 1/2 code which illustrates these ideas. As expected, the probabilities of error of the systems with different quantizers are about the same if they are operated at rate R_o. Also we see that an energy-to-noise ratio of 4.6 db is required to operate at rate R_o with only 2 quantization levels. So we have sacrificed about 2.1 db in the energy-to-noise ratio by adding this quantizer. On the other hand, the 8 level quantizer can operate at rate R_o with ar energy-to-noise ratio of 2.7 db, which represents a loss of only 0.24 db in the energy-to-noise ratio. In many cases this is an acceptable loss because even an infinite increase in the number of quantization levels would only give a 0.24 db improvement in the energy-to-noise ratio.

Other simulation results are shown in Figures 4.4 and 6.3.

5.3 Decoder Approximations

As described previously, a Viterbi decoder retains 2^{K-1} paths at each step; so after Lv symbols have been received, $L\,2^{L-1}$ bits of



BIT PROBABILITY OF ERROR VS. ENERGY-TO-NOISE RATIO FOR A CONVOLUTIONAL CODE WITH SUBGENERATORS $\begin{matrix}11101\\10011\end{matrix}$

FIGURE 5.2

storage are required to describe the paths. For large L this storage requirement clearly becomes excessive, and we are forced to limit the storage available for each path. Assume L_m bits of storage are available for each retained path. Then before another branch can be added to the paths, a hard decision must be made on the information bit L_m branches back. If the paths are not all remerged at that point, the symbol corresponding to the path with the largest metric is chosen. Such a decoder is said to have a memory of L_m. The effects of various memories are illustrated in Figure 5.2. And, in general, it was found experimentally that a memory of five constraint lengths is usually sufficient.

The following argument due to Forney (1967) also supports this choice for the memory of the decoder. Suppose the decoder chooses the first information symbol to be the first symbol of the most likely path out to the i^{th} branch. And assume that all the paths which diverge at the first branch and remerge by the i^{th} branch have smaller log-likelihood functions than that of the correct path. Then the probability of an error in decoding the first information symbol (P_1) is just the probability that the log-likelihood function of some path which diverges from the correct path at the first branch and remains unmerged to the i^{th} branch is greater than the log-likelihood function of the maximum-likelihood path, which agrees with the first symbol of the correct path. This probability can be upper bounded by the probability that one of the totally unmerged paths has a larger log-likelihood function than the correct path. Now select a code at random as in Chapter 2. Then upper bounding the number of totally unmerged paths by 2^i, the expected value of the probability that one of the totally unmerged paths has a larger log-likelihood function than the correct path can be upper bounded by the ensemble average probability of error of a randomly

chosen block code of length iv and rate $R = \frac{\ln 2}{v}$ nats/channel symbol. Thus from (2.4)

$$E(P_1) \leqq e^{-ivE(R)} \quad , \tag{5.4}$$

where E(R) is given by (2.5). Now choose i such that

$$\frac{i}{K} = \frac{E_c(R)}{E(R)} \quad , \tag{5.5}$$

where $E_c(R)$ is the convolutional coding exponent given by (2.36). Then using (5.5) in (5.4) yields

$$E(P_1) \leqq e^{-KvE_c(R)} \quad . \tag{5.6}$$

But (5.6) is asymptotically equal to the corresponding convolutional coding word, and hence bit, error probability bound (2.34). So asymptotically the expected value of the error probability (P_1) is not increased by imposing a final decision at the i^{th} branch if (5.5) is satisfied. For example, for a very noisy channel at rate R_o the ratio of Equation (5.5) is 5.8. Thus asymptotically a memory of 5.8 constraint lengths would be sufficient.

To complete the decoder specifications we must choose a metric for the comparisons. In most of this dissertation a log-likelihood metric is used, but experimentally it was found that the integer metr c performs equally well. And in general, in contrast to the sequential decoders, the Viterbi decoder is relatively insensitive to changes in the metric. So in a practical decoder the system complexity and the memory requirements can be minimized by using an integer metric.

CHAPTER 6

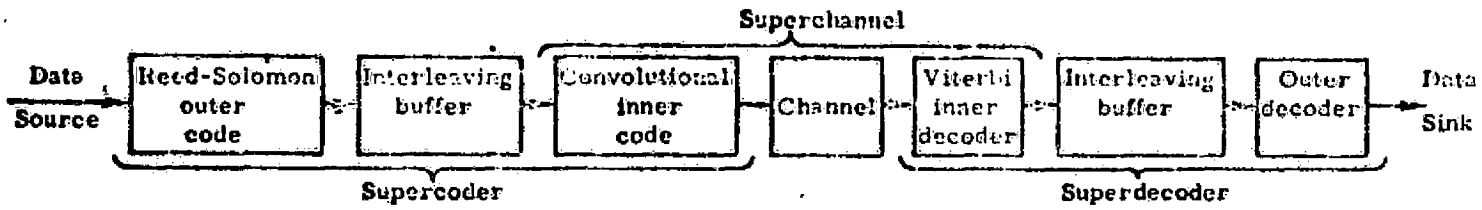
CONCATENATION

In this chapter a concatenated coding system with a convolutional inner code and a Reed-Solomon outer code is described which is capable of operating at very low error probabilities at rates above the computational cutoff rate (R_o) of sequential decoding. The convolutional inner code, decoded with a Viterbi decoder, is used to obtain a moderate probability of error, and then a Reed-Solomon outer code with interleaving achieves the desired probability of error.

6.1 System Description

Previously it was shown that the probability of error of convolutional codes can be made arbitrarily small by making the constraint length sufficiently large. However, increasing the constraint length also increases the complexity of the decoder, and in some high-performance applications the extreme decoder complexity may make the system impractical to build. Recently Forney (1966) has described a concatenated coding system which can be used advantageously in such a situation, and a similar system is considered here. In particular, a concatenated coding system with a Viterbi-decoded constraint length K convolutional inner code and a 2^K-symbol Reed-Solomon outer code, as shown in Figure 6.1, is investigated.

Based on Forney's (1966) results, the convolutional inner code should be chosen to achieve a moderate probability of error at slightly greater than the required rate. So since this is a high-performance system, a rate slightly less than 1/3 bit/channel symbol is assumed acceptable and a rate 1/3 convolutional code was chosen. In the actual simulation a constraint length of 8 was also chosen so that the Viterbi decoder would have a reasonable complexity. Then the inner code should be the optimum (smallest probability of error) code of this



CONCATENATED CODING SYSTEM

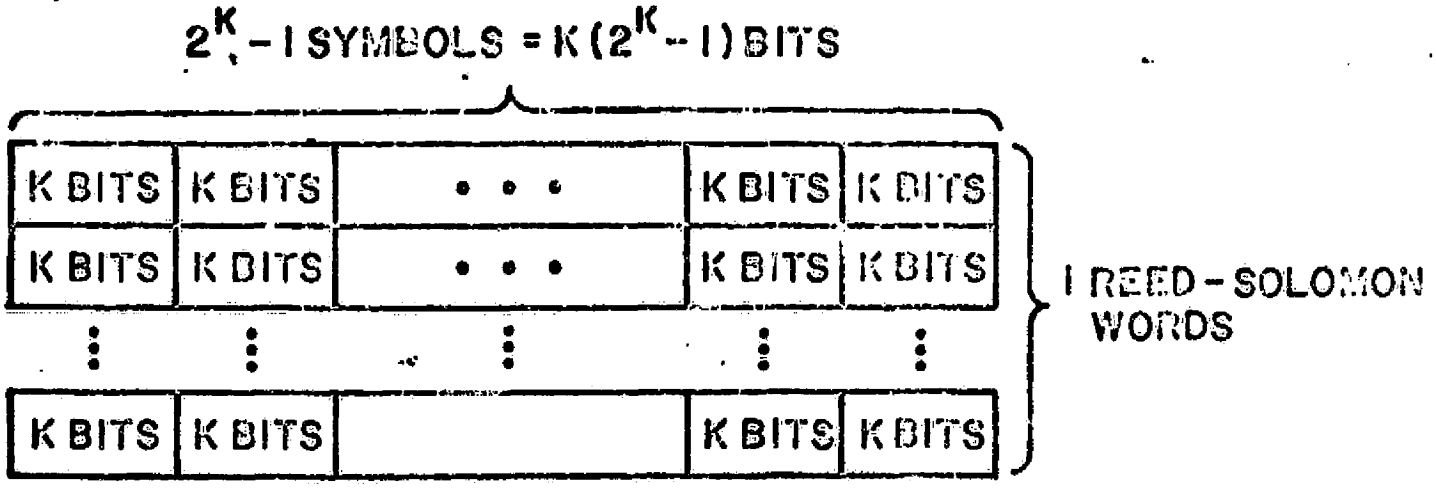
FIGURE 6.1

rate and constraint length and this code should be optimum for relatively small energy-to-noise ratios. Unfortunately, no constructive procedure is known for selecting such a code and the codes found in Chapter 4 were optimum at high energy-to-noise ratios. However experimentally it was found that the codes of Chapter 4 also perform quite well at smaller energy-to-noise ratios. So in the simulation the best rate 1/3 constraint length 8 code from Chapter 4 was used as the inner code.

Now even though the superchannel has a binary input and output, the outer code may have a large alphabet size if we represent each letter of the outer code, by a sequence of superchannel letters. Then for an algebraic outer code the outer decoder operation depends only on the number of sequences of superchannel outputs that contain errors. Such a decoder is especially useful on superchannels where the errors are clustered as for a convolutional inner code. For example for inner code energy-to-noise ratios of the order of 1.25 db the constraint length 8 system which was simulated had 2.5 bit errors per incorrect Reed-Solomon symbol. So for a convolutional inner code, an

algebraic outer code with an alphabet size large enough such that each symbol represents a sequence at least as long as those of the most likely bit error clusters is desirable. And since Reed-Solomon codes belong to this class and are optimum in the sense that no group code with the same alphabet size, block length, and number of check digits can have a larger minimum distance [Gallager (1968)], a 2^K-symbol Reed-Solomon outer code was chosen.

To specify the outer decoder the number of correctable superchannel symbol errors must be established, and for a constraint length of 8 this means the probabilities of having i ($0 \leqq i \leqq 255$) Reed-Solomon errors in the block of 255 symbols must be found. These probabilities are difficult, if not impossible, to determine analytically, but they were determined by simulation, and it was found that in order to counteract the extremely long runs of errors in the superchannel, it was necessary to insert an interleaving buffer between the inner and outer coding operations. A possible alternative to this interleaving buffer, which warrants more investigation, is to use a Reed-Solomon code with a larger alphabet size. That is, choose the Reed-Solomon code such that each symbol represents a sequence of more than K superchannel bits. Here, however, assume that an interleaving buffer, such as in Figure 6.2, is used. The Reed-Solomon encoder first generates i code words and stores them in the rows of the encoding buffer, as in Figure 6.2. Then the Reed-Solomon symbols are read out of the buffer by columns. That is, the first K bits of the first word etc., until the first K bits of the i^{th} word are read. Then the second K bits of the first word are read etc., until all $iK(2^K-1)$ bits are read. For the encoding operation to proceed continuously, two of these buffers are needed. The Reed-Solomon encoder would be storing words in one buffer while the symbols are being read from the other. So $2iK(2^K-1)$ bits of storage are required. The decoding

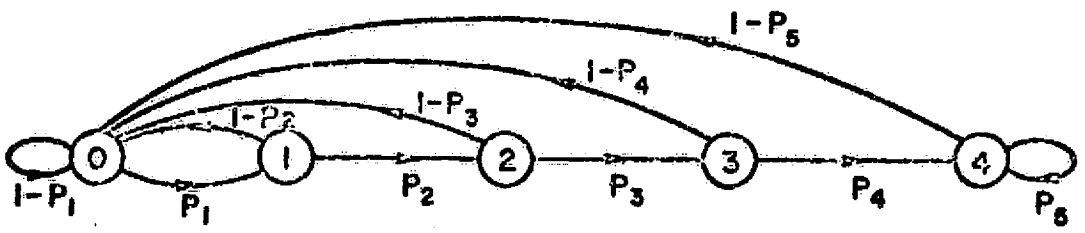


INTERLEAVING BUFFER

FIGURE 6.2

buffer is similar except that the Reed-Solomon symbols are stored by columns and read by rows.

Now i is chosen such that the probability of an error in any Reed-Solomon symbol is independent of the other symbols in that block. To help estimate this i the statistical model of Figure 6.3 was used.



RUN LENGTH STATISTICAL MODEL

FIGURE 6.3

With the exception of state 4, the states in Figure 6.3 represent the number of consecutive Reed-Solomon symbol errors without interleaving. State 4 represents four or more consecutive Reed-Solomon symbol errors. For example, starting in state 0 the probability of j, $j \geq 4$, consecutive symbol errors is $P_1P_2P_3P_4P_5^{j-4}(1-P_5)$. For a constraint length of 8, the transition probabilities (P_i and $1 - P_i$) were determined by simulation for several different inner code energy-to-noise ratios, and the probabilities of having j, $j \geq 0$, consecutive symbol errors were computed from the model. These were then compared with the corresponding probabilities from the simulation (e.g., Table 6.1), verifying that Figure 6.3 is a reasonable model for the system. Thus using this model the probability of having a large number of consecutive symbol errors can be easily computed, whereas by simulation an enormous amount of computer time would be required to accurately estimate the probabilities of these rare events. Now let us choose i such that the probability of i consecutive symbol errors is less than the desired bit probability of error. This makes it unlikely that two Reed-Solomon symbols in the same block will be from the same bit error cluster, or since one correct symbol (K bits) guarantees that the path is remerged with the correct path, it is unlikely that two symbols in the same block will be from the same unmerged span. Thus the symbol errors are relatively independent of each other. Choosing i in this manner, it was found that an i of 16 was sufficient for error probabilities of 10^{-7} and inner code energy-to-noise ratios of the order of 1.25 db. Thus the encoding and decoding buffers must each have 65,280 bits of storage.

6.2 Experimental Results

This system, with a memory 72 integer metric Viterbi decoder and an 8-level quantizer, was simulated on an SDS Sigma-7 computer

TABLE 6.1

COMPARISON SHOWING THE ACCURACY OF THE RUN-LENGTH MODEL FOR AN INNER CODE ENERGY-TO-NOISE RATIO OF 1.5 db

Measured transition probabilities:

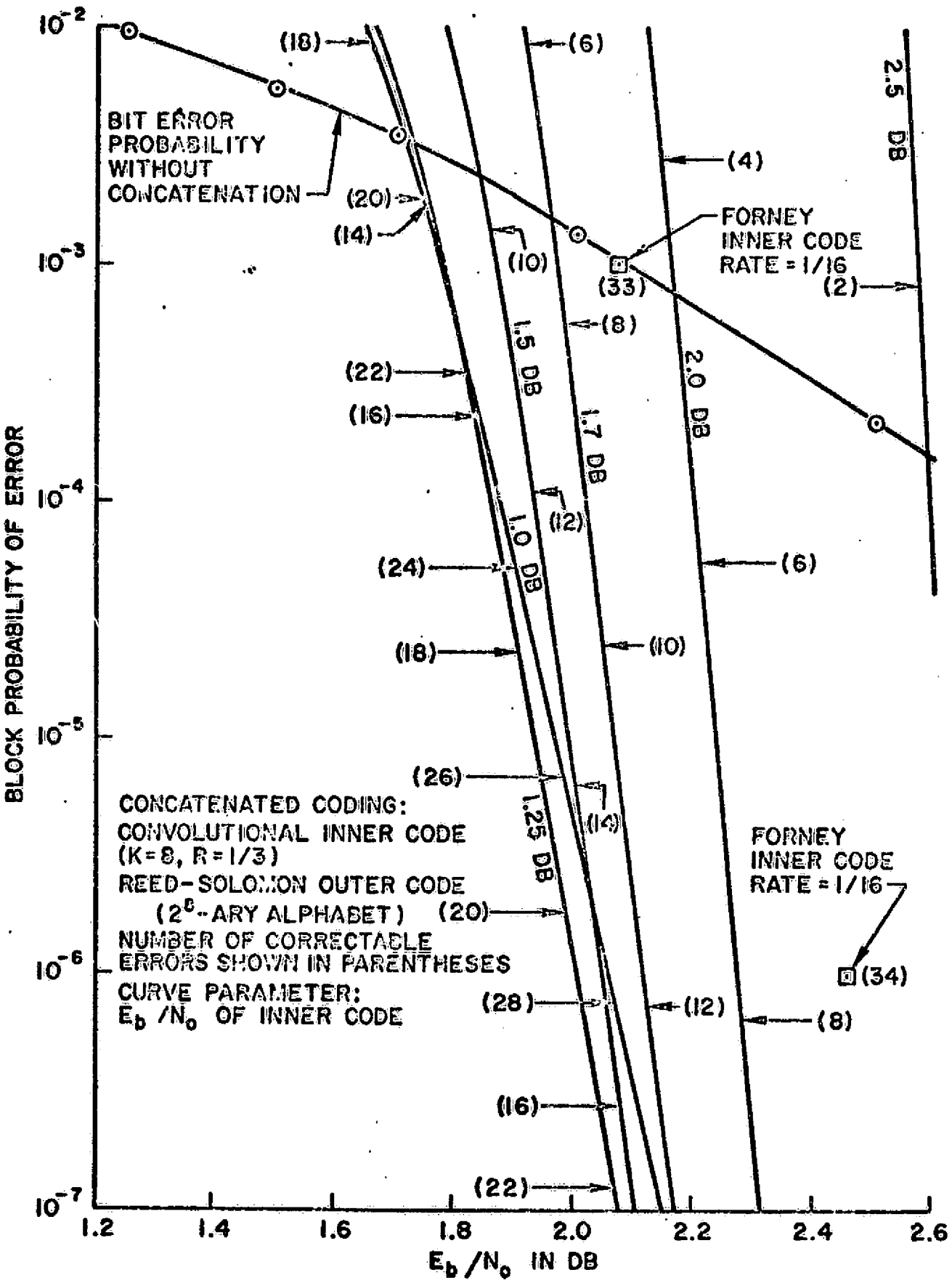
$P_1 = 0.005858$
$P_2 = 0.818427$
$P_3 = 0.483444$
$P_4 = 0.479453$
$P_5 = 0.345794$

	Probability of I consecutive Reed-Solomon errors	
I	Computed from model	Simulation results*
0	9.94×10^{-1}	9.94×10^{-1}
1	1.06×10^{-3}	1.06×10^{-3}
2	2.47×10^{-3}	2.47×10^{-3}
3	1.20×10^{-3}	1.20×10^{-3}
4	7.27×10^{-4}	7.27×10^{-4}
5	2.51×10^{-4}	2.84×10^{-4}
6	8.69×10^{-5}	9.54×10^{-5}
7	3.00×10^{-5}	1.58×10^{-5} (one sample)
8	1.03×10^{-5}	1.58×10^{-5} (one sample)
9	3.59×10^{-6}	
10	1.24×10^{-6}	
11	4.29×10^{-7}	
12	1.48×10^{-7}	
13	5.13×10^{-8}	
14	1.77×10^{-8}	
15	6.14×10^{-9}	
16	2.12×10^{-9}	

*Number of samples = 63,195

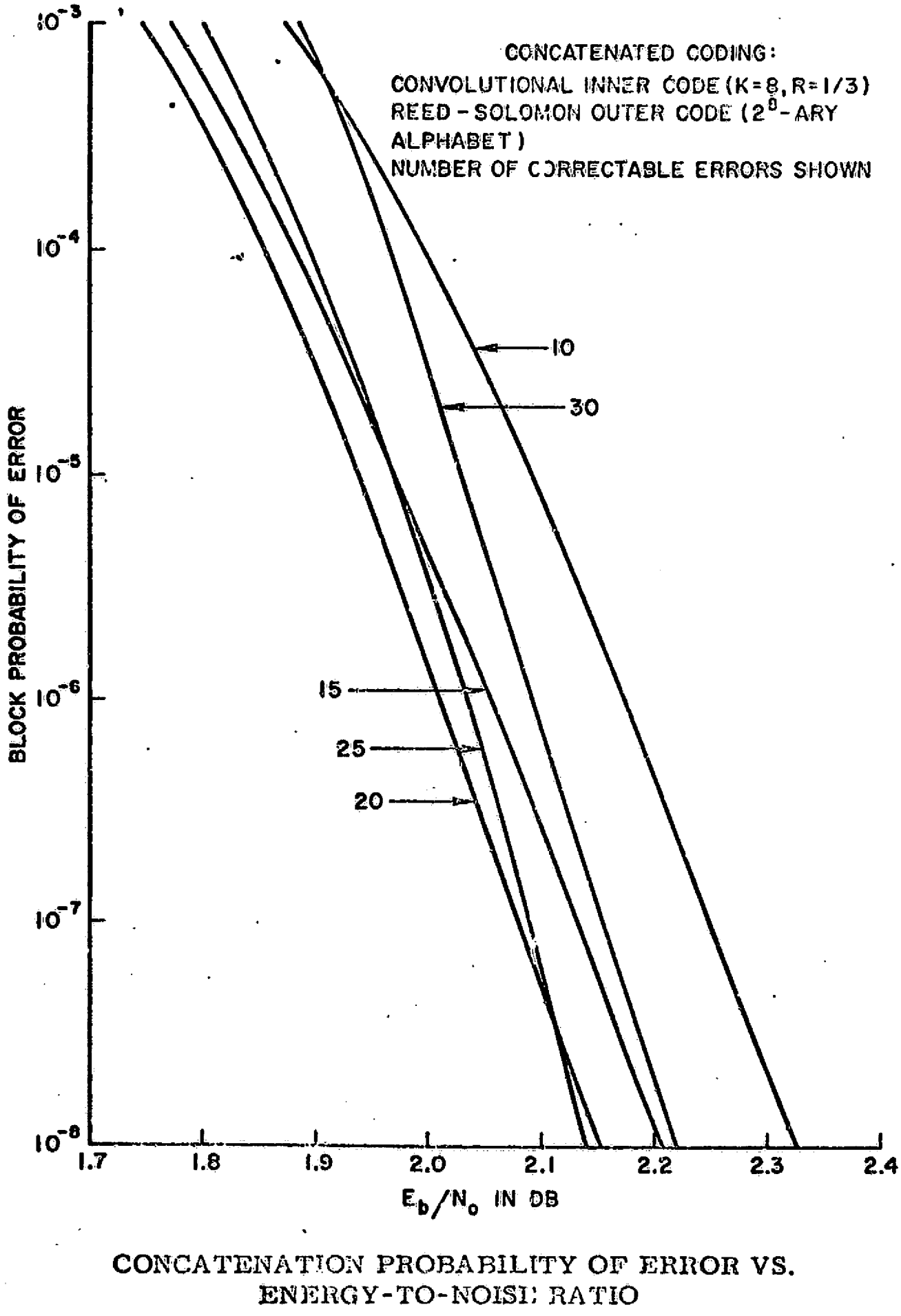
and the results are shown in Figure 6.4 and 6.5. The curves of Figure 6.4 have the energy-to-noise ratio E_b/N_o of the inner code as a parameter, and they show that for block probabilities of error (or coding failure) of the order of 10^{-6} an inner code energy-to-noise ratio of about 1.25 db is optimum. The resulting E_b/N_o adjusted for the outer coding is $(1.25-10 \log_{10} R_{RS})$ db for outer code rates of R_{RS} bits/superchannel symbol. Some of Forney's (1966) results for a biorthogonal inner code and a Reed-Solomon outer code are also shown in this figure. They show that in order to obtain results comparable to those of this system a Reed-Solomon decoder which corrects more errors, and is thus more complex, is needed and much smaller overall rates are required.

Figure 6.5 shows that with an error-correction capability of 20 errors, corresponding to an outer code rate of $\frac{215}{255}$, a block probability of error of 3×10^{-5} can be achieved at an energy-to-noise ratio of 1.9 db. The optimum (unquantized) R_o for this system can be calculated from (5.2), and from (5.3), with $1/3 \left(\frac{215}{255}\right)$ substituted for $1/v$, we see that an energy-to-noise ratio of 1.93 db is required to operate at rate $R_{o\,opt}$. The energy-to-noise ratio required to operate at the unquantized R_o is a few tenths of a db higher than this. Also the bit probability of error may be as much as an order of magnitude less than the block probability of error. For example, if the Reed-Solomon code corrects k errors, then it is possible that only k+1 bit errors in the block could cause k+1 symbol errors and thus an error or a decoding failure, which we are counting as an error. On the average there are about 2.5 bit errors per incorrect symbol for the best constraint length 8 system. So a block decoding error just means, on the average, 2.5 (k+1) or more bit errors in the block of 2040 bits. Thus this system can operate at very small bit error probabilities at rates above R_o.



CONCATENATION PROBABILITY OF ERROR FOR VARIOUS INNER CODE ENERGY-TO-NOISE RATIOS

FIGURE 6.4



CONCATENATION PROBABILITY OF ERROR VS. ENERGY-TO-NOISE RATIO

FIGURE 6.5

APPENDIX B*

Coupled Decoding of a Convolutional-Block Concatenated System

by

G. F. Zeoli

*extracted from Ph.D. Dissertation of G. F. Zeoli "Coupled Decoding of Block-Convolutional Concatenated Codes", 1971. Chapter numbers correspond to dissertation chapters.

CHAPTER 2

CONCATENATED COMMUNICATION SYSTEM

The communication system to be investigated is shown in Figure 2.1. The information source provides I parallel binary channels. These binary sequences may originate from I separate sources or may be the result of suitable multiplexing of a different number of actual sources (see Chapter 8).

The outer encoder is algebraic, operating over $GF(2^H)$. As shown more clearly in Figure 2.2, the outer encoder divides each of the I input binary sequences into H-bit sections, each of which constitutes a symbol which is an element of $GF(2^H)$. The outer algebraic encoder block encodes I input symbols into M = I + P output symbols, yielding M binary streams which then enter the inner encoder.

The inner encoder consists of M parallel convolutional encoders, each of constraint length K_1 and rate 1/v. The output of each inner encoder modulates an antipodal signal generator, the output of which is transmitted over an additive white Gaussian noise channel. The receiver is a matched filter whose output in the absence of noise would be plus or minus a prescribed constant, but due to noise it is a continuous variable having the two constants as conditional means. A 3-bit quantizer (8 levels) provides a discrete output to a Viterbi decoder which operates with a decoding constraint length $K_2 \leq K_1$.

FIGURE 2.1 BLOCK DIAGRAM OF CONCATENATED COMMUNICATION SYSTEM

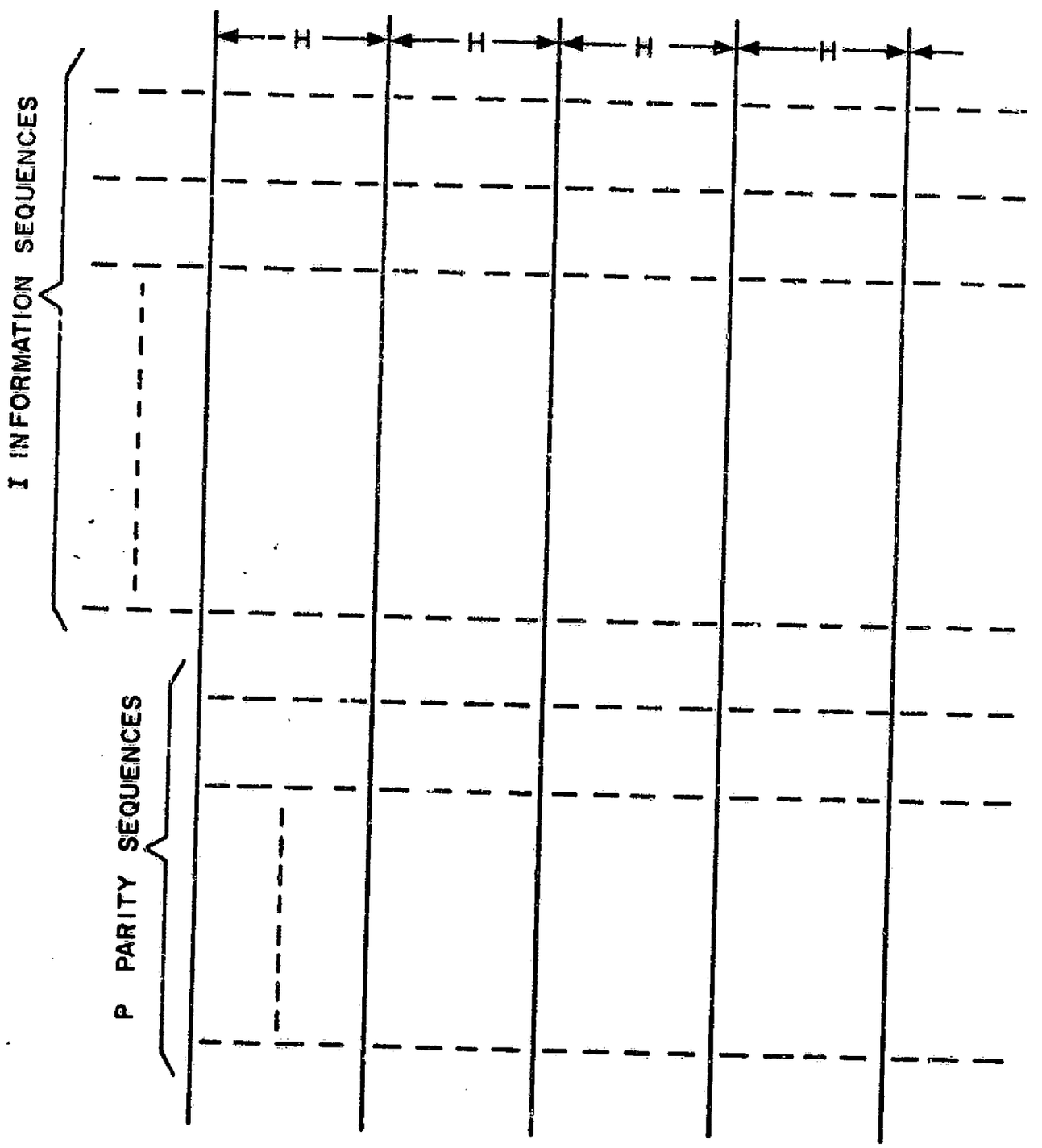


FIGURE 2.2 EXPANSION OF NUMBER OF DATA STREAMS FROM I TO M = I + P IN OUTER ENCODER

2.1 Threshold Erasure Decoding Procedure

The operation of the Viterbi decoder in each of the M channels is as follows: It begins decoding and continues until N symbols have been decoded by each of the 2^{K_2-1} states of the decoder, at which time the decoding ceases. (This decoding cycle involves Nv channel symbols.) The state having the largest metric is selected and its metric is compared to a threshold. If the metric exceeds the threshold, the initial H bits of the N-bit decoded sequence are accepted; otherwise, they are erased. The remaining N-H decoded bits are discarded and not utilized. Each of the accepted H-bit sequences is an estimate of the corresponding H-bit symbol (element of $GF(2^H)$) produced by the outer algebraic encoder. The outer decoder is thus presented with M H-bit symbol estimates, some of which are erasures, to decode algebraically. Provided that the number of erasures plus twice the number of remaining undetected errors (i.e., whose metrics passed the threshold test) does not exceed one less than the minimum distance of the outer code, the outer decoder is assured of being able to correct the errors and correctly fill in the erasures. Otherwise errors may occur.

The inner Viterbi decoder will now begin decoding again at the first receiver output symbol that is the result of the entrance into the convolutional encoder of the first binary input symbol of the next H-bit input sequence. That is, the Viterbi decoder decodes N input symbols at a time, involving Nv receiver output symbols, but only utilizes the initial H of them each cycle of operation. The decoder shifts along H input symbols each cycle. However, before the Viterbi decoder in the m^{th} inner channel, $1 \leq m \leq M$, can begin the next cycle,

every state sequence (of which it has 2^{K_2-1}) must be set equal to the sequence so far decoded for that channel, including the H symbols of the last cycle. If the m^{th} channel suffered an erasure or an error during the last cycle, the most recent H bits are provided by the outer decoder, which presumably has corrected the error or erasure. The M inner decoders now again decode N symbols, stop, erase or accept the first H symbols, etc., the cycle repeating itself.

2.2 Fixed Erasure and GMD Decoding Procedures

The decoding procedure just described compares the maximum state metric of each of the M Viterbi convolutional decoders with a threshold, and based on that comparison, either accepts the first H decoded symbols or erases them. This procedure will be referred to as the Threshold Erasure procedure.

An alternate procedure would be to always erase the first H symbols of a fixed number, N_φ, of the M channels which have the lowest maximum state metrics. The number N_φ should be chosen less than D-1, where D is the minimum distance of the outer algebraic code, so that some error correction capability remains in the outer algebraic decoder. This procedure will be referred to as the Fixed Erasure procedure.

A third alternative procedure is suggested by the generalized minimum distance (GMD) decoding concept formulated by Forney [1966]. A GMD decoder is utilized as the outer decoder in place of algebraic decoding. This GMD decoder will be explained in greater detail in the next chapter, but briefly the idea is the following: The maximum state metric of the Viterbi decoder in each of the M channels provides a

measure of the reliability of the corresponding initial H-bit sequence. One might consider a decoder similar to a maximum likelihood decoder that considered all 2^{HI} possible M-symbol algebraic codewords, and in some suitable fashion picked that codeword which "most reliably" compares with the output of the inner decoder. This procedure will be made more precise in the next chapter, and will be referred to as the GMD procedure.

CHAPTER 3
DETAILED SYSTEM DESCRIPTION

A brief general description of the communication system to be investigated was given in Chapter 2. Here that description is made more precise preparatory to an evaluation of error performance to be undertaken in the following chapter. The general block diagram of the system is shown in Figure 2.1.

3.1 Outer Encoder

The outer encoder accepts I binary data streams from an information source, as shown in Figures 2.1 and 2.2. The encoder divides the streams into H-bit words, each such word constituting a single symbol belonging to $GF(2^H)$.

The outer code is taken to be a BCH code. The properties of these codes will not be described here; the reader is referred to Forney [1966], Gallager [1968], Berlekamp [1968], and the references listed there. Whenever the code length (i.e., the total number of communication channels M) is less than 2^H, a Reed-Solomon (RS) code is the optimum code in the sense of minimum distance. The RS codes are a subset of the BCH codes with the property that for a given code length M and number of information symbols I, they exhibit the maximum minimum distance of any linear code. Again, the reader is referred to the above references for a description of the RS codes.

The BCH and RS codes are characterized for our purposes by a minimum distance D. For I information symbols and M total symbols,

implying P = M - I parity check symbols, the minimum distance is given by

$$D = P + 1 \qquad \text{for RS codes}$$

$$D < P + 1 \qquad \text{for BCH codes which are not RS} \qquad (3.1)$$

A partial list of binary BCH codes and their minimum distances is given in Peterson [1961].

The outer encoder therefore accepts I input symbols belonging to $GF(2^H)$ and generates P additional parity check symbols, also belonging to $GF(2^H)$, thus producing M = I + P binary data streams at its output. An example of a specific RS encoder is given in Forney [1966].

3.2 Inner Encoder

The inner encoder is of the convolutional type. The general form is shown in Figure 3.1. One of the M binary streams emanating from the outer encoder enters the convolutional encoder of Figure 3.1 from the left, bit-by-bit. After any given input bit has entered the shift register, v modulo 2 sums are formed from the tapped outputs shown in the figure. Whether a given shift register position is connected to a given modulo 2 summer is determined by the "convolutional code". A great deal of research has gone on and is going on to discover procedures for constructing good convolutional codes, good in the sense of low error probability. See, for example, Odenwalder [1970].

The convolutional encoder produces v binary output symbols for every input symbol, one emanating from each modulo 2 summer. The rate

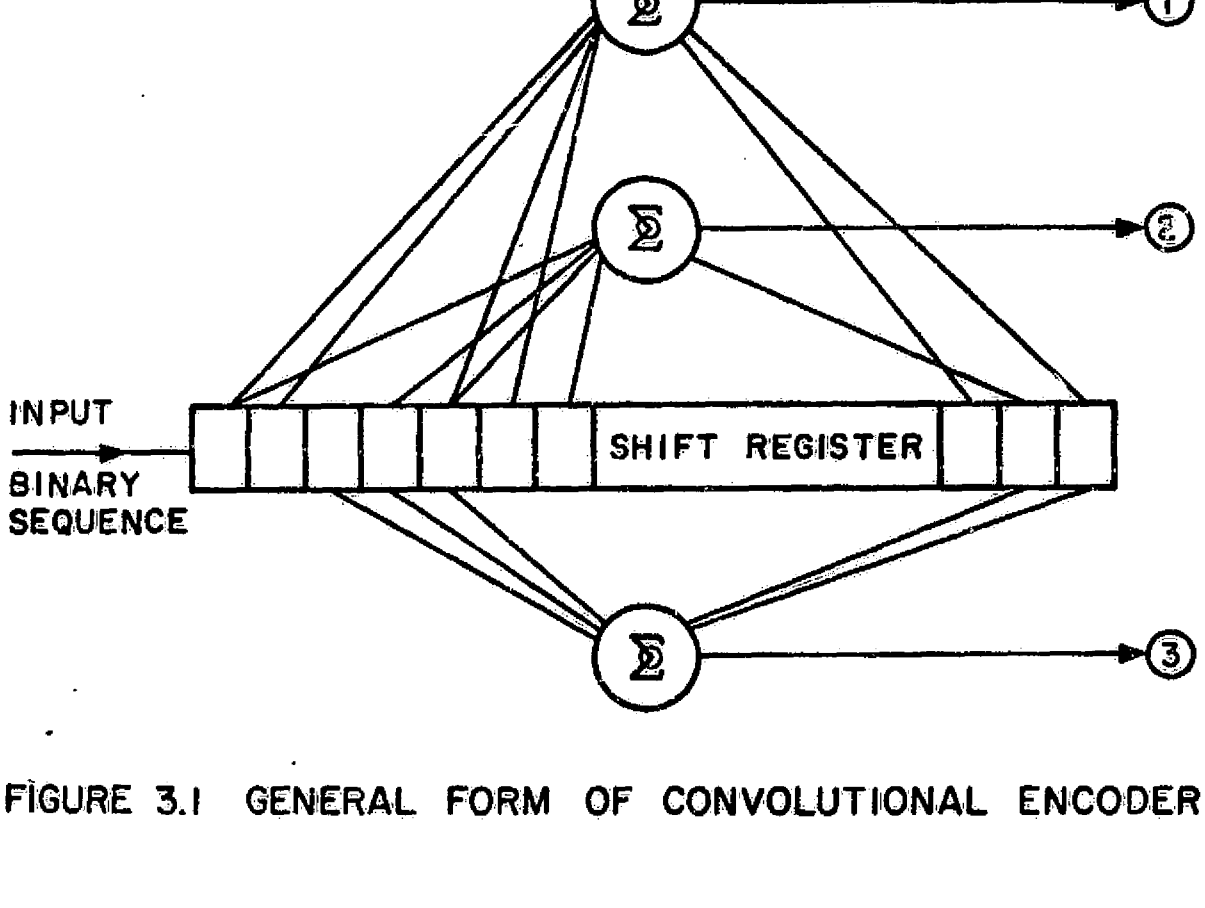


FIGURE 3.1 GENERAL FORM OF CONVOLUTIONAL ENCODER

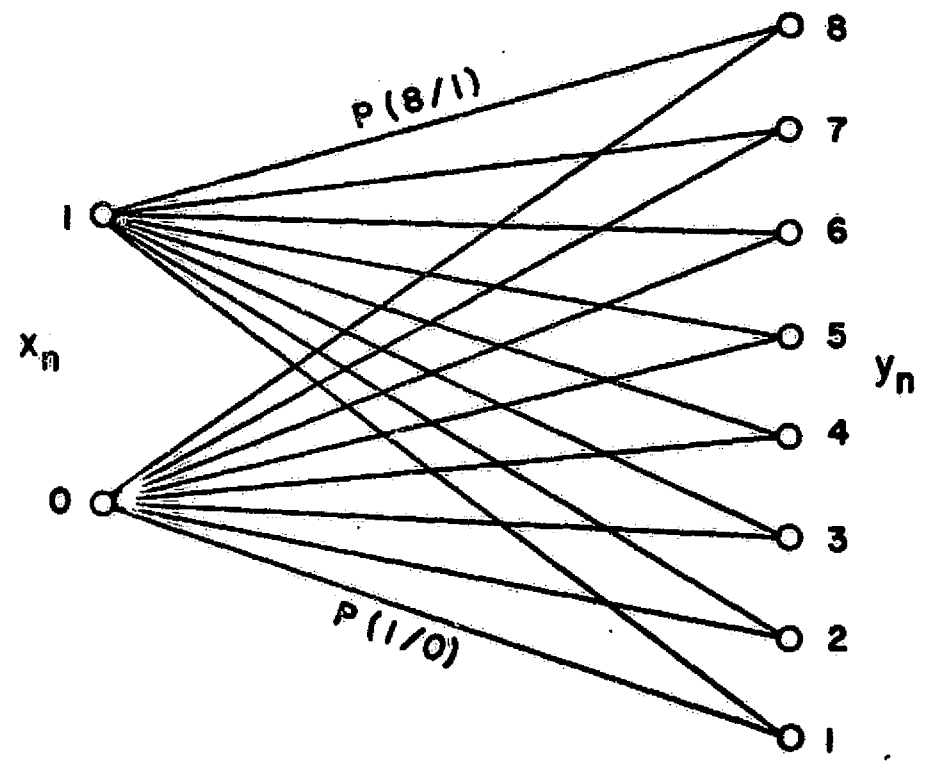


FIGURE 3.2 DISCRETE INPUT - DISCRETE OUTPUT EQUIVALENT CHANNEL

of the convolutional encoder is 1/v. The output of the overall inner encoder is thus M binary streams at v times the rate of its input.

3.3 AWGN Channel

The transmission channel in each of the M channels is assumed to be an additive white Gaussian noise channel. The transmitter is modulated by the binary stream produced by the inner convolutional encoder. If T is the time between successive bits from the inner encoder, the transmitter choses one of two possible equal-energy signals of duration T. The receiver is assumed to be synchronous, knowing when each signal begins.

The optimal likelihood detector for white Gaussian noise interference is a matched filter. The most practical signal is a constant amplitude sine wave where the binary modulation chooses one of two possible phases. The optimum choice of phases to minimize error probability is 0 and π; that is, one signal is the negative of the other, commonly called antipodal.

Thus the transmitter transmits

$$S_n(t) = \begin{cases} A \cos \omega(t-nT) & o < t - nT < T \quad \text{if } x_n = 1 \\ -A \cos \omega(t-nT) & o < t - nT < T \quad \text{if } x_n = 0 \end{cases} \tag{3.2}$$

where x_n is the n^{th} input symbol. The receiver observes

$$y_n(t) = S_n(t) + n(t) \qquad o < t - nT < T \tag{3.3}$$

where n(t) is a sample function of white Gaussian noise. The matched or correlation receiver forms

$$z_n = \int_{nT}^{(n+1)T} y_n(t) \cos \omega(t-nT)dt$$

$$= \pm \frac{AT}{2} + \int_{nT}^{(n+1)T} n(t) \cos \omega(t-nT)dt \qquad (3.4)$$

where the double frequency term in the signal portion of the integral is assumed zero ($\omega T = 2\pi n$) or negligible.

Therefore z_n is plus or minus a constant, depending on whether x_n was 1 or 0, plus a Gaussian random variable. Letting

$$\eta_n = \int_{nT}^{(n+1)T} n(t) \cos \omega(t-nT)dt \qquad (3.5)$$

it is easily shown that

$$E(\eta_n) = 0 \qquad (3.6)$$

$$E(\eta_n^2) = \sigma_{\eta_n}^2 = \frac{N_o T}{4} \qquad (3.7)$$

where $N_o/2$ is the two-sided power spectral density of the zero-mean Gaussian white noise. The probability density of the receiver output can now be written as

$$P(z_n|x_n) = \sqrt{\frac{2}{\pi N_o T}} \exp\left\{-2\left[z_n-(2x_n-1)\frac{AT}{2}\right]^2/N_oT\right\} \qquad (3.8)$$

where x_n = 1 or 0.

It is impractical to store and process the precise output of the correlation detector. Therefore the continuous receiver output is quantized by a 3-bit (8-level) quantizer, with a resulting loss of about 0.2 dB (Jacobs [1967]; see also Chapter 7). Others have determined (Heller [1969], Odenwalder [1970]) that for this 8-level quantizer the quantization interval should be chosen approximately equal to one-half the magnitude of the expected output,

$$Q = \frac{AT}{4} \qquad (3.9)$$

The loss due to quantization is not very sensitive to the choice of Q.

Thus the discrete input-discrete output channel equivalent shown in Figure 3.2 will have transition probabilities

$$P(1|0) = \int_{-\infty}^{-3Q} P(z_n|0)dz_n = \frac{1}{2}\left[\operatorname{erf}\left(-\frac{1}{2}\sqrt{\frac{U_s}{N_o}}\right)+1\right] \qquad (3.10)$$

$$P(m|0) = \int_{(m-5)Q}^{(m-4)Q} P(z_n|0)dz_n = \frac{1}{2}\left[\operatorname{erf}\left(\left(\frac{m-2}{2}\right)\sqrt{\frac{U_s}{N_o}}\right) - \operatorname{erf}\left(\left(\frac{m-3}{2}\right)\sqrt{\frac{U_s}{N_o}}\right)\right]$$

$$m = 2, 3, \ldots 7 \qquad (3.11)$$

$$P(8|0) = \int_{3Q}^{\infty} P(z_n|0)dz_n = \frac{1}{2}\left[1 - \frac{1}{2}\operatorname{erf}\left(\frac{5}{2}\sqrt{\frac{U_s}{N_o}}\right)\right] \qquad (3.12)$$

and

$$P(j|1) = P(9-j|0) \qquad (3.13)$$

where

$$U_s = \frac{A^2 T}{2} = \text{signal energy per AWGN channel bit} \qquad (3.14)$$

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-y^2}\,dy \qquad (3.15)$$

3.4 Inner Decoder

The inner decoder is a Viterbi decoder, described in detail in Viterbi [1967]. Actually a separate Viterbi decoder of constraint length $K_2 \leq K_1$ (recall that K_1 is the convolutional encoding constraint length) operates in each of the M parallel channels. In a practical

implementation, a single decoder would be time-shared among the M channels (see Chapter 8).

The initial step of one of these decoders of constraint length K_2 is to consider all 2^{K_2} possible binary input sequences of length K_2 which could enter the encoder bit-by-bit until they are completely contained in the shift register. Each such sequence generates a K_2v-bit code sequence which is compared to the received sequence using some metric, usually the likelihood function, or more conveniently, the log likelihood function. The received sequence is a sequence of K_2v 8-level symbols (hereafter referred to as octal) from the receiver quantizer. Designating as y_i the v octal symbols corresponding to the i^{th} encoder input symbol, the metric for the m^{th} input sequence, $m = 1, 2, \ldots 2^{K_2}$, is

$$\mu_m = \sum_{i=1}^{K_2} \log P(y_i|x_{mi}) \qquad (3.16)$$

where $x_m = \{x_{m1}, x_{m2}, \ldots, x_{mK_2}\}$ is the m^{th} input sequence.

The possible input sequences are considered in pairs which differ from one another only in the first symbol x_{m1}. The sequence in each pair having the higher metric is retained along with its metric; its "adversary" is discarded. Thus after the first step is completed there remain 2^{K_2-1} candidate sequences with their metrics. Each of these is referred to as a state, numbered by the binary number $x_{m2}, x_{m3}, \ldots$ x_{mK_2} that remains, which is unique to each "survivor".

The second step is to take each of the surviving candidate input sequences, consider the next input symbol to be either a one or zero, compute the corresponding $P(y_{K_2+1}|x_{mK_2+1})$, add it to the stored metric for that survivor, thus creating again 2^{K_2} adversaries. Now regroup them in pairs which again are identical in the last K_2-1 input symbols, and choose as survivors those members of each pair having the higher metrics. There are 2^{K_2-1} survivors, and the cycle repeats.

In the communication system described here, this decoder goes through N such steps and then stops, selecting from its 2^{K_2-1} states that one having the highest metric. In the Threshold Erasure procedure, this highest metric is compared to a threshold, and if the threshold is exceeded, the initial H bits of the corresponding state sequence are accepted and passed on to the outer decoder; otherwise the outer decoder receives an erasure.

In the Fixed Erasure procedure, the M highest state metrics from the M parallel channels are compared after N bits are decoded, and the $N_\varphi \leq D-1$ channels having the smallest metrics have their first H bits erased. The initial H-bit sequences of the remaining $M-N_\varphi$ channels are accepted and passed on to the outer decoder.

In the GMD procedure, the initial H-bit portion of the state sequence having the highest metric is accepted for each of the M channels, regardless of the metric value, and passed on to the outer decoder. Also passed along to the outer decoder is the corresponding maximum state metric which provides a measure of the reliability of the H-bit sequence as an estimate of the true channel input sequence.

3.5 Outer Decoder

The outer decoder may be either an algebraic or a GMD decoder. The outer encoder, recall, was taken to be a linear encoder, using symbols which are elements of $GF(2^H)$, such as a BCH encoder. The weight of a codeword is defined to be the number of non-zero symbols in the codeword. The distance between any two codewords is the weight of their difference, or simply the number of symbol positions in which they differ. A linear code has the property that any linear combination of codewords is another codeword. Thus to evaluate the distance properties of a code one need only evaluate the weights of its codewords. The distance defined here is commonly called the Hamming distance. The minimum distance of a linear code is the minimum weight of any of its codewords.

All linear codes of length M possess an information subset of size I which uniquely determines the codewords. The remaining M - I symbols are check symbols. It is the fact that, of the q^M possible received sequences (with symbols elements of GF(q)), only q^I are possible codewords that provides the "distance" between codewords, which in turn provides the error correcting capability. In the present instance, the I input channels provide the set of I information symbols, and the outer encoder generates P = M - I check symbols.

The minimum distance between codewords of a linear code is limited by

$$D \leq M - I + 1 \tag{3.17}$$

A code that achieves the maximum possible distance for the given M and I is called a maximum code. Reed-Solomon codes are maximum, whereas BCH codes in general are not. A Reed-Solomon code exists whenever the code length M is less than the size of the code alphabet. For a code with symbols which are elements of GF(q),

$$M < q . \tag{3.18}$$

For the communication system considered here, where the outer symbols are elements of $GF(2^H)$, we require that

$$M < 2^H \tag{3.19}$$

in order to be able to use a Reed-Solomon code. The minimum distance is then

$$D = M - I + 1 = P + 1 \tag{3.20}$$

where P is the number of check or parity symbols.

In general the received outer codeword estimate (output of inner decoder) that is presented for outer decoding may contain, in addition to correctly received symbols, N_e symbols in error and N_φ erased symbols. The erased symbols arise when the demodulator preceding the decoder is sufficiently uncertain as to which symbol it has received. The decoding algorithm then simply computes that codeword closest to the received word in the Hamming metric. If N_φ symbols have been

erased and so are ignored in determining distance, the minimum distance between codewords in the remaining symbols is at least $D - N_\varphi$. Thus, if the number of symbol errors satisfies

$$N_e < \frac{1}{2}(D - N_\varphi) \tag{3.21}$$

then decoding to the closest codeword is assured of yielding the correct word. Correct decoding is assured providing

$$2N_e + N_\varphi < D \, . \tag{3.22}$$

For RS codes with $D = P + 1$, this becomes

$$2N_e + N_\varphi \leq P \, . \tag{3.23}$$

Thus it appears that in a sense errors are twice as serious as erasures, and this leads us to expect that errors-and-erasures decoding will give superior performance to errors-only decoding.

The other type of outer decoder considered is a GMD decoder. If the inner decoder, which is essentially the demodulator for the outer decoder, provides only symbol estimates or erasures to the outer decoder, it is providing highly quantized reliability information. The GMD decoder attempts to utilize a greater amount of reliability information which the inner decoder can provide by the likelihood information contained in its maximum state metric.

Let μ_m be the maximum state metric in the m^{th} channel of the inner decoder, m = 1, 2, ... M, associated with the decoding of one M-symbol outer codeword that is to be presented to the outer decoder. There are M code symbols together with M corresponding metrics available to the outer decoder.

One might consider as an outer decoder the following: The outer symbols are elements of $GF(2^H)$. The outer code has I information symbols, so there are 2^{HI} possible outer codewords of length M. Consider comparing the received codeword, as provided by the inner decoder, with each possible transmitted codeword.

Define

$$S(\hat{x}_m, x_{km}) = \begin{cases} 1 & \hat{x}_m = x_{km} \\ -1 & \hat{x}_m \neq x_{km} \end{cases} \qquad (3.24)$$

where $\hat{x}_m$ is the symbol estimate of the m^{th} channel, m = 1, 2, ... M, and x_{km} is the corresponding symbol of the k^{th} possible codeword, k = 1, 2, ... 2^{HI}. Form

$$J_k' = \sum_{m=1}^{M} S(\hat{x}_m, x_{km})\, \mu_m \qquad (3.25)$$

as a measure of how well (reliably) the received codeword compares with the k^{th} [illegible] transmission, and decode the j^{th} codeword, where j is the k which maximizes J_k'.

This decoder is exhaustive, in that it will always yield a decoded word, assuming a rule for deciding ties. However, it suffers the same drawback of optimum maximum likelihood decoders: the number of possible codewords, and therefore the complexity of a real-time decoder grows exponentially with code length M for a given rate (i.e., exponentially with I).

One approach to reducing complexity is to perform what Forney calls conditional decoding. The reliability metric μ_m is transformed by a nonlinear function $\alpha(\mu_m)$ such that

$$0 \leq \alpha(\mu_m) \leq 1 \ . \tag{3.26}$$

The result is that the new code comparison metric

$$J_k = \sum_{m=1}^{M} S(\hat{x}_m, x_{km})\, \alpha(\mu_m) \tag{3.27}$$

is such that if J_k exceeds some threshold J_o, one can be sure that no other codeword can also do so, so that the search can be terminated. That is, we have the following theorem (Forney [1966]):

Theorem

There is at most one codeword from the linear code of length M and minimum distance D for which

$$J_k = \sum_{m=1}^{M} S(\hat{x}_m, x_{km})\, \alpha(\mu_m) > M - D \qquad (3.28)$$

where $S(\hat{x}_m, x_{km})$ is defined in (3.24) and $\alpha(\mu_m)$ satisfies (3.26).

Proof

Let (3.28) be satisfied by the k^{th} codeword

$$\underline{x}_k = \{x_{k1}\ x_{k2} \cdots\cdots x_{kM}\}$$

and let $\underline{x}_{k'}$ be any codeword at a distance D or more from $\underline{x}_k$. Let the sets of locations S, T and U be defined by

$$\begin{aligned}
x_{km} &= x_{k'm} && m \in S\ ; \\
x_{km} &\neq x_{k'm} \text{ and } \hat{x}_m = x_{km}, && m \in T\ ; \\
x_{km} &\neq x_{k'm} \text{ and } \hat{x}_m \neq x_{km}, && m \in U\ . \qquad (3.29)
\end{aligned}$$

Now, using (3.24) and (3.29),

$$J_k = \sum_{m \in S} S(\hat{x}_m, x_{km})\, \alpha(\mu_m) + \sum_{m \in T} \alpha(\mu_m) - \sum_{m \in U} \alpha(\mu_m)$$

$$= A_S + A_T - A_U \qquad (3.30)$$

and

$$J_{k'} = A_S - A_T + \sum_{m \in U} S(\hat{x}_m, x_{k'm})\, \alpha'(\mu_m)$$

$$= A_S - A_T + B_U \tag{3.31}$$

Since $$S(\hat{x}_m, x_{k'm}) \leq 1, \tag{3.32}$$

$$B_U \leq A_U . \tag{3.33}$$

Since $S(\hat{x}_m, x_{km})\, \alpha(\mu_m) \leq 1$, and the number of elements in the set S is at most M - D, then it follows that

$$A_S \leq M - D \tag{3.34}$$

It was initially postulated that $J_k > M - D$, so combining (3.28), (3.30) and (3.34)

$$J_k = A_S + A_T - A_U > M - D$$

$$A_T - A_U > M - D - A_S > 0 \tag{3.35}$$

and hence, using (3.31), (3.33) and (3.35),

$$J_{k'} = A_S - A_T + P_U \leq A_S - A_T + A_U$$

$$< A_S \leq M - D$$

$$\therefore J_{k'} < M - D \qquad (3.36)$$

as was to be proved.

While there is at most one $\underline{x}_k$ such that $J_k > M - D$, there may be no $\underline{x}_k$ that satisfies (3.28). In this event, the decoder fails to decode at all. Thus the characterization "conditional decoding": decoding occurs on the condition that there exists a codeword $\underline{x}_k$ satisfying (3.28).

There still remains the problem of finding the codeword $\underline{x}_k$, if one exists, that satisfies (3.28). Forney [1966] shows a technique for finding it by successively erasing the least reliable symbols in $\hat{\underline{x}}$ and doing an errors-and-erasures decoding. He further shows that no more than $(D+1)/2$ such erasing and decoding steps are required to find $\underline{x}_k$, if it exists.

The nonlinear function $\alpha(\mu)$ that will be used here is the piecewise-linear function shown in Figure 3.3, where the breakpoints are to be optimized. If $\alpha(\mu) = 1$ for all μ, the decoding is errors-only. If $\mu_\ell = \mu_u$, the decoding corresponds to Threshold Erasure errors-and-erasures decoding. In general the optimum $\alpha(\mu)$ will be

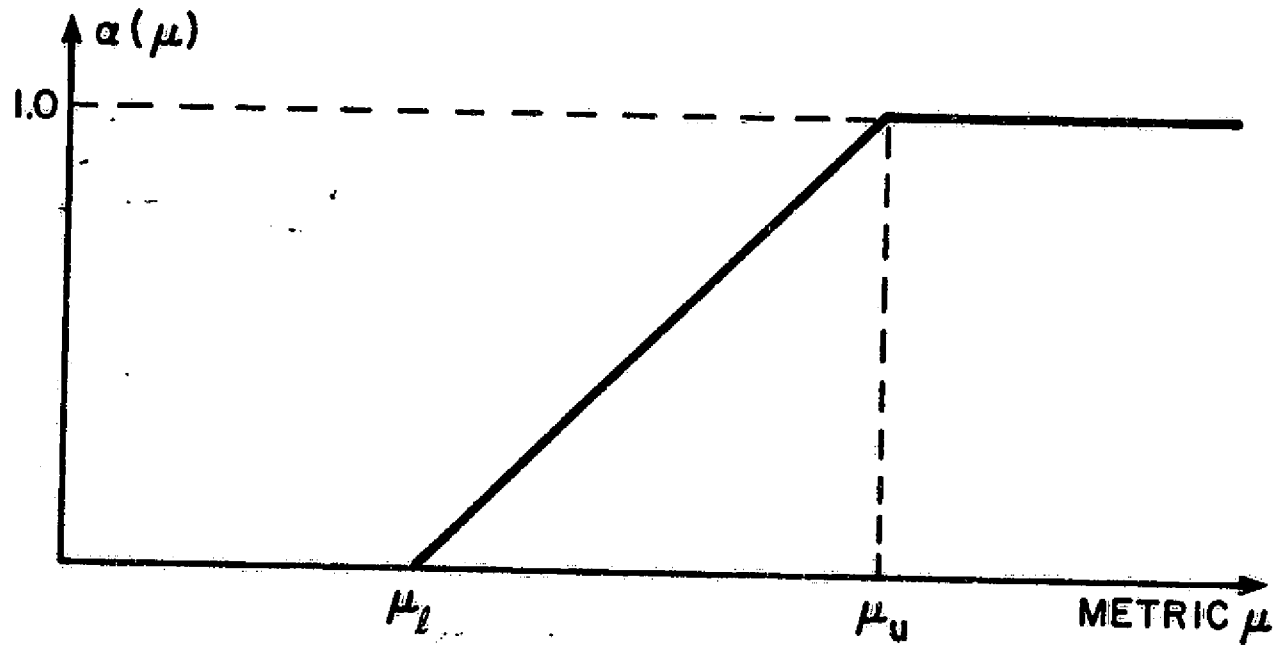


FIGURE 3.3 PIECEWISE LINEAR FUNCTION $\alpha(\mu)$

different from either of these limiting cases, providing superior performance to both.

3.6 Decoder Feedback

At the beginning of each decoding cycle, the inner Viterbi decoder in each of the M inner channels must have all of its 2^{K_2-1} state sequence registers set equal to the sequence so far decoded for that channel, including the H symbols from the last cycle. If an erasure or error occurred, this is made manifest by a difference existing between the first H bits of the maximum metric state (which may be an erasure) and the corresponding H-bit word outputted by the outer decoder. The latter prevails and is fed back to the inner decoder.

CHAPTER 4
ERROR PERFORMANCE

This chapter will develop expressions for the probability of decoding error for the concatenated communication system shown in Figure 2.1. Recall that in each of the M parallel channels the inner encoder-channel-decoder employs a rate 1/v convolutional code and a Viterbi-type decoder. The decoder begins decoding, decodes N symbols (goes through N steps as described in Chapter III), stops, examines its maximum state metric, and then, depending upon which outer decoding procedure is used, passes on to the outer decoder either the first H decoded symbols of the maximum metric state, or an H-symbol erasure, or the first H symbols together with the maximum state metric. After the outer decoder has corrected any errors or erasures and fed these corrections back to the inner decoder, the inner decoder is then ready to step along Hv channel symbols from where it had last started, and to begin again, repeating the cycle.

The performance of any of the outer decoders, and therefore of the entire system, will depend upon whether decoding errors are made among the first H symbols of the M parallel channels, and upon the statistical properties of the maximum state metric in each channel.

The M parallel inner channels are independent. The performance of any of the possible outer decoders, and therefore of the entire system, will depend upon the following properties of each of the inner channels: (1) the probability of a decoding error occurring within the first H decoded symbols of the maximum-metric state (call this event E); (2) the

distribution of the maximum state metric, given a decoding error has occurred among the first H symbols of that state sequence; (3) the unconditional distribution of the maximum state metric. We define for the inner decoder:

$p(\mu|E)$ = probability density of the maximum state metric given that an error has occurred among that state's first H decoded symbols.

$p(\mu)$ = unconditional probability density of the maximum state metric.

$$P(\mu|E) = \int_{-\infty}^{\mu} p(\xi|E)d\xi \qquad (4.1)$$

$$P(\mu) = \int_{-\infty}^{\mu} p(\xi)d\xi \qquad (4.2)$$

In the following discussion of the three outer decoders it will be seen that the performance of each of them can be written conveniently in terms of two probability functions, $P_e(\mu)$ and $P_\varphi(\mu)$, which apply to any one of the M inner decoder channels, defined as follows:

$P_e(\mu_T)$ = probability that a decoding error occurs among the first H decoded symbols, and that the maximum state metric exceeds μ_T

$$= Pr\{E\} \int_{\mu_T}^{\infty} p(\xi|E)d\xi$$

$$= \Pr\{ E \} \left[1 - P(\mu_T | E) \right] \tag{4.3}$$

$P_\varphi(\mu_T)$ = probability that the maximum state metric fails to exceed μ_T

$$= \int_{-\infty}^{\mu_T} p(\xi) d\xi = P(\mu_T) \tag{4.4}$$

4.1 Threshold Erasure Outer Decoding Procedure

In this decoding procedure the maximum state metric of each inner decoder channel is compared to a threshold μ_T, and the first H decoded symbols of the corresponding state sequence are passed on to the outer decoder if the threshold is exceeded; otherwise, an erasure is passed along. Using an errors-and-erasures decoding algorithm, the outer decoder is assured of being able to correct errors and replace erasures correctly, provided the number of channels in error N_e and the number erased N_φ satisfy (3.22):

$$2N_e + N_\varphi < D \tag{3.22}$$

The M parallel channels are independent. The number of channels putting out correct, erroneous and erased H-ary symbols is given therefore by the multinomial distribution, and the overall probability of decoding error is bounded by

$$P_E \leq \sum_R \binom{M}{N_\varphi \quad N_e} P_e^{N_e} P_\varphi^{N_\varphi} (1-P_e-P_\varphi)^{M-N_e-N_\varphi} \tag{4.5}$$

where $P_e = P_e(\mu_T)$, $P_\varphi = P_\varphi(\mu_T)$, R is the region where (3.22) is violated, and

$$\binom{M}{N_\varphi \; N_e} = \frac{M}{N_\varphi!\, N_e!\, (M-N_\varphi-N_e)!} .$$

More specifically,

$$P_E \le \sum_{N_e=0}^{N_o} \sum_{N_\varphi=D-2N_e}^{M} \binom{M}{N_\varphi \; N_e} P_e^{N_e} P_\varphi^{N_\varphi} (1-P_e-P_\varphi)^{M-N_e-N_\varphi}$$

$$+ \sum_{N_e=N_o+1}^{M} \binom{M}{N_e} P_e^{N_e} (1-P_e)^{M-N_e} \qquad (4.6)$$

where N_o is the largest integer such that $2N_o < D$.

Rather than compute exactly the probability of violating (3.22), we could compute instead a Chernoff bound. Defining x_m for the m^{th} channel, m = 1, 2, ..., M as

$$x_m = \begin{cases} 2 & m^{th} \text{ channel contains an error within the first H decoded symbols, and } \mu_m > \mu_T \\ 1 & \mu_m < \mu_T \\ 0 & \text{neither of the above} \end{cases} \qquad (4.7)$$

Then the violation of (3.22) is identical to the condition

$$\sum_{m=1}^{M} x_m \geq D \tag{4.8}$$

The x_m are independent discrete random variables, and applying the standard Chernoff bound (Gallager [1968]) we have

$$\Pr\left[\sum_{m=1}^{M} x_m \geq D\right] \leq e^{-sD}\, E\left\{e^{s\Sigma x_m}\right\}$$

$$= e^{-sD}\left[E\left\{e^{sx}\right\}\right]^M \qquad s > 0 \tag{4.9}$$

where x is one of the x_m. Using $P_e(\mu_T)$ and $P_\varphi(\mu_T)$ defined in (4.3) and (4.4), the expectation in (4.9) is found to be

$$E\left\{e^{sx}\right\} = 1 - P_\varphi(\mu_T) - P_e(\mu_T) + e^s\, P_\varphi(\mu_T) + e^{2s}\, P_e(\mu_T) \tag{4.10}$$

Therefore a bound on (4.6) would be

$$P_E \leq \min_{s>0} e^{-sD}\left[1 + (e^s-1)\, P_\varphi(\mu_T) + (e^{2s}-1)\, P_e(\mu_T)\right]^M \tag{4.11}$$

4.2 Fixed Erasure Outer Decoding Procedure

Here the maximum state metrics of the M parallel inner channels are compared, and the N_φ channels having the smallest maximum metrics have their first H decoded symbols erased, while the remaining $M - N_\varphi$ channels have their first H symbols accepted and passed on to the outer decoder. In contrast to the first procedure, the number N_φ is fixed rather than being a random variable determined by threshold crossings. It should be chosen less than D - 1 to leave some error-correcting capability in the outer algebraic decoder. The optimum value to use to minimize overall error probability will depend on the inner channel signal-to-noise ratio.

Consider the M maximum state metrics that exist at the end of one stage of the inner decoder, $\mu_1, \mu_2, \ldots \mu_M$. Reorder them by magnitude,

$$\mu_j^{(1)} \le \mu_k^{(2)} \le \ldots\ldots \le \mu_z^{(M)}$$

The p-smallest $\mu_r^{(p)} \triangleq \mu^{(p)}$ has the density function

$$p\left(\mu^{(p)}\right) d\mu^{(p)} = \binom{M}{M-p\ p-1} P\left(\mu^{(p)}\right)^{p-1} \left[1-P\left(\mu^{(p)}\right)\right]^{M-p} p\left(\mu^{(p)}\right) d\mu^{(p)} \tag{4.12}$$

where $P(\mu)$ is the unconditional maximum state metric distribution function (4.2) and $p(\mu)$ the corresponding density.

If N_φ channels are erased, the maximum number of errors that is assured of being correctable by the outer decoder is N_e, the maximum integer such that

$$2N_e < D - N_\varphi \tag{4.13}$$

An outer decoding error may occur if the number of remaining unerased channels having errors exceeds N_e.

We need the probability that any given channel, whose maximum state metric is known to at least equal $\mu^{(N_\varphi)}$, contains a decoding error among its first H decoded symbols. Again let E be the event that the first H decoded symbols contain an error, and E_2 be the event that the maximum state metric exceeds $\mu^{(N_\varphi)}$. Using Bayes rule we have

$$\Pr(E|E_2) = \frac{\Pr(E_2|E)\,\Pr(E)}{\Pr(E_2)} = \frac{P_e\left[\mu^{(N_\varphi)}\right]}{1 - P_\varphi\left[\mu^{(N_\varphi)}\right]} \tag{4.14}$$

using the P_e and P_φ defined in (4.3) and (4.4).

Therefore, for a given value of $\mu^{(N_\varphi)}$, the largest metric in the set of the N_φ smallest of the maximum state metrics of the M parallel channels, the probability of error is

$$\Pr\left\{\text{error}\,|\,\mu^{(N_\varphi)}\right\} \le \sum_{N=N_e+1}^{M-N_\varphi} \binom{M-N_\varphi}{N} \left(\frac{P_e\left[\mu^{(N_\varphi)}\right]}{1-P_\varphi\left[\mu^{(N_\varphi)}\right]}\right)^N \left(\frac{1-P_e\left[\mu^{(N_\varphi)}\right] - P_\varphi\left[\mu^{(N_\varphi)}\right]}{1 - P_\varphi\left[\mu^{(N_\varphi)}\right]}\right)^{M-N_\varphi-N} \tag{4.15}$$

The overall probability of error is thus, combining (4.15) and (4.12),

$$P_E \leq \int_{-\infty}^{\infty} p\left\{\mu_\varphi(N_\varphi)\right\} \Pr\left\{\text{error} \middle| \mu_\varphi(N_\varphi)\right\} d\mu_\varphi(N_\varphi)$$

$$= \sum_{N=N_e+1}^{M-N_\varphi} \binom{M}{N\ N_\varphi-1\ M-N_\varphi-N} \int_0^1 P_e^{N}(\xi)\left[1-P_e(\xi)-P_\varphi(\xi)\right]^{M-N_\varphi-N}$$

$$\times \left[P_\varphi(\xi)\right]^{N_\varphi-1} dP_\varphi(\xi) \qquad (4.16)$$

This error probability may also be bounded by using a Chernoff bound as was done for the Threshold Erasure decoder. Let x_m for the m^{th} channel be defined by

$$x_m = \begin{cases} 2 & \text{if } m^{th} \text{ channel has an error in the first H decoded symbols} \\ 0 & \text{otherwise} \end{cases} \qquad (4.17)$$

for $m \in S$, where S is the set of unerased channels, of which there are $M-N_\varphi$. Now, using (4.13), the probability of error can be bounded by

$$P_E \leq \Pr\left[\sum_{m\in S} x_m \geq D - N_\varphi\right] \leq \min_{s>0} e^{-s(D-N_\varphi)}\, E\left\{e^{s\Sigma x_m}\right\} \qquad (4.18)$$

The expectation in (4.18) is over the joint probability distribution of the x_m and $\mu^{(N_\varphi)}$, and can be written as

$$E\left\{e^{s\Sigma x_m}\right\} = \int_{-\infty}^{\infty} p\left(\mu^{(N_\varphi)}\right)\, d\mu^{(N_\varphi)} \sum_{\substack{x_m \\ m\in S}} e^{s\Sigma x_m}\, P\left(x_m \middle| \mu^{(N_\varphi)}\right) \qquad (4.19)$$

As far as the conditional expectation is concerned, holding $\mu^{(N_\varphi)}$ constant, the x_m are independent random variables. The conditional probability of one of the x_m's being 2 is given by (4.14). Thus (4.19) can be rewritten as

$$E\left\{e^{s\Sigma x_m}\right\} = \binom{M}{M-N_\varphi\ \ N_\varphi-1} \int_0^1 P_\varphi\left(\mu^{(N_\varphi)}\right)^{N_\varphi-1}\left[1 - P_\varphi\left(\mu^{(N_\varphi)}\right)\right]^{M-N_\varphi}$$

$$\cdot\left[1-\frac{P_e\left(\mu^{(N_\varphi)}\right)}{1-P_\varphi\left(\mu^{(N_\varphi)}\right)} + e^{2s}\frac{P_e\left(\mu^{(N_\varphi)}\right)}{1-P_\varphi\left(\mu^{(N_\varphi)}\right)}\right]^{M-N_\varphi} dP_\varphi\left(\mu^{(N_\varphi)}\right) \qquad (4.20)$$

Substituting this expression into (4.18) gives for the error bound

$$P_E \le \binom{M}{M-N_\varphi\ \ N_\varphi-1} \min_{s>o}\left\{e^{-s(D-N_\varphi)}\right.$$

$$\left.\int_0^1 \left[1-P_\varphi\left(\mu^{(N_\varphi)}\right) + (e^{2s}-1)\, P_e\left(\mu^{(N_\varphi)}\right)\right]^{M-N_\varphi} \cdot P_\varphi\left(\mu^{(N_\varphi)}\right)^{N_\varphi-1} dP_\varphi\left(\mu^{(N_\varphi)}\right)\right\} \qquad (4.21)$$

4.3 CMD Outer Decoding Procedure

When the GMD decoder is used for the outer decoder, the inner decoder passes on the first H decoded symbols from the maximum metric state sequence of each of the M parallel inner channels together with the metric. The CMD decoder transforms the metric of the m^{th} channel into $0 \leq \alpha(\mu_m) \leq 1$. Referring to the discussion of this decoder in Chapter III, there is at most one codeword $\underline{x}_k$ which satisfies the relationship (3.28)

$$J_k = \sum_{m=1}^{M} S(\hat{x}_m, x_{km})\, \alpha(\mu_m) > M - D \qquad (3.28)$$

Therefore, one may upper bound the probability of error by computing the probability that (3.28) is not satisfied by the correct codeword.

$$P_E \leq \Pr\left\{J_k \leq M - D \mid \underline{x}_k \text{ is transmitted}\right\} \qquad (4.22)$$

Here we compute a Chernoff bound on P_E. J_k is the sum of M independent random variables. Using the standard Chernoff bound,

$$\Pr\left\{J_k \leq M - D\right\} \leq e^{-s(M-D)}\; E\left\{e^{sJ_k}\right\} . \qquad (4.23)$$

$$s < 0$$

Because of the independence of each term in the sum (3.28),

$$E\left\{e^{sJ_k}\right\} = \left[E\left\{e^{s\alpha(\mu)\ S(\hat{x},x_k)}\right\}\right]^M = \left[g(s)\right]^M \qquad (4.24)$$

where the channel subscript has been omitted. Evaluating the expectation we have

$$E\left\{e^{s\alpha(\mu)\ S(\hat{x},x_k)}\right\} = g(s)$$

$$= Pr\left\{\ \bar{E}\ \right\} \int_{-\infty}^{\infty} e^{s\alpha(\mu)}\ p(\mu\,|\,\bar{E})d\mu$$

$$+ Pr\left\{\ E\ \right\} \int_{-\infty}^{\infty} e^{-s\alpha(\mu)}\ p(\mu\,|\,E)d\mu \qquad (4.25)$$

where the event E is the occurrence of a decoding error within the first H symbols of the maximum-metric state, and $\bar{E}$ is the complement of E. From the definition of $P_e(\mu_T)$ in (4.3) it is seen that

$$Pr\left\{\ E\ \right\}\ p(\mu\,|\,E) = -\frac{dP_e(\mu)}{d\mu} \qquad (4.26)$$

Similarly, from the definition of $P_\varphi(\mu_T)$ in (4.4) we have

$$P_\varphi(\mu_T) = Pr\left\{\ E\ \right\} \int_{-\infty}^{\mu_T} p(\mu\,|\,E)d\mu + Pr\left\{\ \bar{E}\ \right\} \int_{-\infty}^{\mu_T} p(\mu\,|\,\bar{E})d\mu \qquad (4.27)$$

from which we obtain

$$\Pr\left\{\bar{E}\right\} p(\mu|\bar{E}) = \frac{dP_{\varphi}(\mu)}{d\mu} + \frac{dP_{e}(\mu)}{d\mu} \tag{4.28}$$

Therefore (4.25) can be rewritten as

$$g(s) = \int_0^1 e^{s\alpha(\mu)} \left[dP_{\varphi}(\mu) + dP_{e}(\mu)\right] - \int_0^1 e^{-s\alpha(\mu)}\, dP_{e}(\mu) \tag{4.29}$$

Using the piecewise linear transformation $\alpha(\mu)$ shown in Figure 3.3, (4.29) simplifies to

$$\begin{aligned} g(s) &= P_{\varphi}(\mu_\ell) + e^{s}\left[1 - P_{\varphi}(\mu_u) - P_{e}(\mu_u)\right] \\ &\quad + e^{-s} P_{e}(\mu_u) + \int_{\mu_\ell}^{\mu_u} e^{s\left[(\mu-\mu_\ell)/(\mu_u-\mu_\ell)\right]} \left[dP_{\varphi}(\mu)+dP_{e}(\mu)\right] \\ &\quad - \int_{\mu_\ell}^{\mu_u} e^{-s\left[(\mu-\mu_\ell)/(\mu_u-\mu_\ell)\right]}\, dP_{e}(\mu) \end{aligned} \tag{4.30}$$

Returning to (4.23), the error probability bound can be written as

$$P_E \leq e^{-s(M-D)} \left[g(s)\right]^M = e^{-s(M-D)+M\mu(s)} \tag{4.31}$$

where $\mu(s) = \ln g(s)$ and $s < o$. Letting $\delta = D/M$,

$$P_E \leq \min_{s>o} e^{-M\left[s(1-\delta)-\mu(s)\right]} \tag{4.32}$$

The GMD decoder reduces to the familiar errors-only or errors-and-erasures (with a threshold eraser) algebraic decoders for the limiting cases:

$$\alpha(\mu) = 1 \quad \text{for all } \mu \qquad \text{errors-only decoder}$$

$$\alpha(\mu) = \begin{cases} 1 & \mu \geq \mu_T \\ 0 & \mu < \mu_T \end{cases} \qquad \text{errors-and-erasures decoder} \tag{4.33}$$

Notice, for instance, that in the latter case the two integrals in (4.30) vanish and the error bound (4.31) becomes

$$P_E \leq \min_{s<o} e^{-s(M-D)}\left[e^{s}\left(1-P_{\varphi}(\mu_T)-P_e(\mu_T)\right)\right.$$
$$\left. + P_{\varphi}(\mu_T) + e^{-s}\, P_e(\mu_T)\right]^M \tag{4.34}$$

which is identical to (4.11) for the Threshold Erasure decoder if s is replaced by -s.

In the former case, $\alpha(\mu) = 1$ for all μ, the moment generating function becomes

$$g(s) = e^{s} + (e^{-s}-e^{s})\, P_e(-\infty) \tag{4.35}$$

and the error bound for errors-only decoding (i.e., an error occurs if $2N_e \geq D$) is

$$P_E \leq \min_{s<0} e^{-s(M-D)}\left[e^{s} + (e^{-s}-e^{s})\, P_e(-\infty)\right]^{M} \qquad (4.36)$$

The fact that the GMD decoder includes our Threshold Erasure decoder and an errors-only decoder as limiting cases assures us that an optimization of the GMD decoder over $\alpha(\mu)$ must provide at least as good performance as either of these limiting cases.

4.4 - Summary of Error Performance Expressions

The probability of failure of the outer decoder, and therefore of the entire system, is bounded by (4.6) or (4.11) for the Threshold Erasure outer decoding procedure and by (4.16) or (4.21) for the Fixed Erasure procedure.

A bound on the failure probability for the GMD decoder is given in (4.32), with limiting cases given in (4.34) and (4.36). In all cases the expressions are constructed as functions of $P_e(\mu)$ and $P_\infty(\mu)$, given in (4.2) and (4.3). It is these probability functions that will be evaluated by the simulation to be described in Chapter 6.

CHAPTER 6
SIMULATION DESIGN

The concatenated communication system of Figure 2.1 was simulated on the IBM 360-91 at the UCLA Campus Computer Network. In the previous dicussion in Chapters 3 and 4, it was seen that the nature of the outer encoder-decoder determined the structure of the error probability equations, each of which could be written in terms of two error probability functions describing the error performance of the inner encoder-decoder. The outer encoder was always taken to be algebraic, using a BCH code. When the number of parallel transmission channels is less than the algebraic code alphabet, a Reed-Solomon code (a sub-class of BCH codes) is maximal and the best to use.

Three outer decoders have been considered: one employs a threshold to test the maximum state metric of each of the M inner decoders to decide whether to accept or erase that state's initial H decoded bits, and then does an errors-and-erasures algebraic decoding across the M 2^H-ary symbols; a second always erases a fixed number E of the M inner channel outputs, and then also does errors-and-erasures algebraic decoding; the third (GMD) decoder accepts the first H decoded symbols and the metric of the state having the maximum metric in each of the M inner channels, performs a nonlinear transformation on the metric, and then by a sequence of not more than (D+1)/2 errors-and-erasures decoding steps, arrives at a (outer) decoded word provided the decoded word from the inner decoder lies sufficiently close to a possible transmitted word.

6.1 General Parameter Selection

Since it is a straightforward matter to compute the error performance of the various outer encoder-decoder combinations once the performance of the inner encoder-channel-decoder is known, only the latter need be simulated. The performance of the M inner channels is independent, so only one was simulated. Table 6.1 lists the parameters of a typical inner channel that were chosen for the simulation.

It was desirable to choose the inner convolutional encoder constraint length considerably longer than practical decoding constraint lengths. Because of the word size of the IBM 360 computer, an encoding constraint length of 32 was chosen. The decoding constraint length wanted to be as long as currently practical. The decoding constraint length was chosen to be 8, although an early simulation at constraint length 4 was done to gain experience in simulating the system. The constraint length 4 simulation was programmed in FORTRAN IV, with a

TABLE 6.1 – GENERAL SIMULATION PARAMETERS

Encoder Constraint Length K_1	= 32
Decoder Constraint Length K_2	= 8
Convolutional Code Rate 1/v	= 1/3
Signal Energy to Noise Power Density Ratio Per Information Bit $(U_B/N_o)_I$	= 1.25 dB, 1.0 dB, 0.5 dB, 0 dB
Signal Energy to Noise Power Density Ratio Per Channel Bit $(U_S/N_o)_I$	= -3.55 dB, -3.8 dB, -4.3 dB, -4.8 dB
Antipodal Channel Signals; Expected Output	= ±2Q
Eight Level Output Quantizer, Step Size	= Q
Number of Decoded Data Symbols per Cycle	= 40

running time of approximately 0.4 sec per trial. Since the constraint length 8 simulation should take roughly 16 times longer, it was clearly necessary to employ a more efficient computer language. Therefore, the bulk of the constraint length 8 simulation was written in IBM 360 ASSEMBLER language. With this more efficient language the time per trial turned out also to be approximately 0.4 sec/trial, indicating an approximate 16 to 1 speed advantage.

Because of the linearity of the code, the decoder error performance is the same for all possible input sequences. Thus, to simplify the simulation the input data sequence was always taken to be the all-zeros sequence. The noise was generated by using the commonly available computer internal program which generates random numbers uniformly distributed over (0,1). Dividing this interval into subintervals proportional to the appropriate set of channel transition probabilities, (3.10) to (3.15), readily provides the random transitions.

The random noise generation and the processing of the maximum state metric and decoding error data was done in a FORTRAN main program, while the Viterbi decoder was simulated in an ASSEMBLER language subprogram. The entire program is included in the Appendix.

6.2 Convolutional Code Selection

A very important matter is the selection of the convolutional code generators; i.e., the particular tap connections to be used between the data shift register and the v modulo 2 summers of Figure 2.1. Many researchers have attempted to develop procedures for constructing good codes for particular types of decoders, but no one has as yet

constructed optimum codes, in the sense of minimizing the probability of decoding error for some particular type of decoder.

One severe handicap in the search for good code construction procedures is the lack of a way short of simulation to evaluate accurately how good a particular code is. Various distance measures have been defined which are more readily evaluated, and which simulation experience has shown are well correlated with the error performance of particular decoders. For example, the minimum free distance, D_{FREE}, is found to correlate well with sequential decoder error performance (Costello [1969]). D_{FREE} is the minimum weight of the code words generated by passing all possible sequences entirely through the encoder. Since sequential decoders base their decisions on the weights of long sequences of code symbols, this correlation is not surprising. To evaluate D_{FREE}, however, it is necessary to consider all possible sequences out to length $\approx K^2$, where K is the constraint length (Rosenberg [1971]).

Another distance measure which is easier to evaluate than D_{FREE} is D_{MIN}. D_{MIN} is the minimum weight codeword produced by all possible data sequences of length K, where K is the encoder constraint length, introduced into the encoder until they just fill the data register. Thus the codewords whose weights are evaluated are all Kv long, where 1/v is the convolutional code rate. This distance measure might be expected to be well correlated with the performance of a decoder that operated discontinuously on short sequences (comparable to the constraint length) of code symbols. Although the inner decoder here appears to operate in this manner, in reality, because of the feedback

of data from the outer decoder to the inner decoder, the inner decoder in each of the M channels actually decodes continuously, except for an occasional stopping and restarting to guard against catastrophic failure. (See Chapter 8).

However, we chose to use D_{MIN} as an aid to choosing good codes for the simulation because of the relative ease of its evaluation.* The codes found to have a large D_{MIN} were further compared among themselves in the actual simulation.

It was decided to use a rate 1/3 non-systematic code. Non-systematic codes were considered because of their advantage over systematic codes in D_{FREE} (although D_{MIN} is insensitive to this advantage). Odenwalder [1970] has found the best rate 1/2 and rate 1/3 non-systematic codes out to lengths 9 and 8, respectively, by exhaustive search. By best is meant "If one of these codes were used on a BSC with a low crossover probability or with an antipodal PSK modulator on an AWGN channel with symmetric output quantization at high energy-to-noise ratios, then it would have a bit probability of error less than or equal to that of any other code of that rate and constraint length."

The Odenwalder codes did not meet the present need for two reasons: (a) they are not long enough, the longest rate 1/3 code being constraint length 8, whereas we wish to use length 32; (b) the codes he found are best at high energy-to-noise ratios, whereas the region of interest here is at very low energy-to-noise ratio. Recall that the reason for

---

*I am indebted to S. Curry for the use of his computer program for the evaluation of D_{MIN}.

the latter is that the inner decoder is allowed to have a relatively high error probability in concatenation schemes, with the outer decoder then correcting these errors to produce the overall desired low error probability.

Other "good" codes were considered as a basis for extension by random selection, such as the two non-trivial generators of Costello's [1969] best rate 1/3 constraint length 32 systematic codes, and Bahl and Jelinek's* rate 1/2 complementary code of maximum constraint length 24. Neither of these gave a D_{MIN} as large as those obtained starting with Odenwalder's best constraint length 8 code.

Therefore, for lack of any better alternative, the simulation was done using the best rate 1/3 constraint length 8 code from Odenwalder, filling out the remaining 24 bits in each of the three generators by random selection. Several random selections were used, all of which exhibited large D_{MIN}. Random coding arguments would lead one to expect that randomly selected codes have a substantial likelihood of being reasonably good. This is particularly so in the present scheme where only the first H decoded symbols are of primary interest, where H was never required to exceed 10.

Upper and lower bounds exist for D_{MIN} for asymptotically large constraint length fixed convolutional encoders. Robinson [1965] and Massey [1968] have shown that for all binary fixed codes of rate $R = 1/v$ and constraint length K,

---

*Bahl, L.R. and F. Jelinek, "Free Distance Structure of Convolutional Codes and Complementary Rate 1/2 Codes," Unpublished.

$$\lim_{K\to\infty} D_{MIN} \leq \frac{K(v-1)}{2} \tag{6.1}$$

which for K = 32, v = 3, is $D_{MIN} \leq 32$.

The lower bound guarantees that at least one code can be found with minimum distance greater than or equal to the bound. Wozencraft and Reiffen [1961] and Massey [1963] have shown that there exists at least one binary fixed code of rate R = 1/v and constraint length K such that

$$\lim_{K\to\infty} D_{MIN} \geq H^{-1}\left(\frac{v-1}{v}\right) Kv \tag{6.2}$$

where $H(x) = -x \log_2(x) - (1-x) \log_2(1-x)$ is the binary entropy function. For v = 3, $H^{-1}(2/3) = .175$, so the lower bound is $D_{MIN} \geq 16.8$.

Therefore, we look for codes with D_{MIN} in the range $17 \leq D_{MIN} \leq 32$. Table 6.2 lists the codes that were used and the D_{MIN} evaluated for each. The codes are given in hexadecimal, and are of binary constraint length 32. The repetition of the first two hexadecimal symbols represents the Odenwalder code to which randomly chosen bits were added.

TABLE 6.2 – D_{MIN} FOR CONVOLUTIONAL CODE GENERATORS USED IN SIMULATION

Generators	D_{MIN}	Generators	D_{MIN}
F7743EDE		F77BE447	
D9746F2C	25	D92B5AAB	25
950C9BEE		95F40234	
F759BAC1		F7E8D487	
D9FBE3C1	24	D9C69291	24
95DD65B8		95656298	
F711F4EE		F713AEF0	
D98226A9	24	D99C6306	24
9524116C		9584746C	

CHAPTER 7
SIMULATION RESULTS

This chapter will present the results of the simulation of one inner channel, as described in Chapter 6, and will discuss the application of these results to the evaluation of overall error probability for the concatenated communication system of Figure 2.1. The bounds on probability of error developed in Chapter 4 will be evaluated for the three decoders: Threshold Erasure, Fixed Erasure and GMD. The performance achieved will be compared to that obtained previously by Odenwalder [1970] and Forney [1966].

7.1 Direct Simulation Results

The parameters of the inner superchannel that was simulated are given in Table 6.1. The principal direct results from the computer simulation are shown in Figures 7.1 through 7.4. Curves of $P_e(\mu_T)$ and $P_\varphi(\mu_T)$, (4.3) and (4.4), are plotted as a function of μ_T for H = 4, 6 and 8. Recall that the outer code alphabet size is 2^H, and in order to use Reed-Solomon outer codes, the number of parallel inner channels M must be chosen small enough that $2^H > M$.

The number of simulation trials that were made at each $(U_B/N_o)_I$ are tabulated in Table 7.1. More will be said later in this chapter about the statistical confidence to be placed in the resulting curves of Figures 7.1 through 7.4.

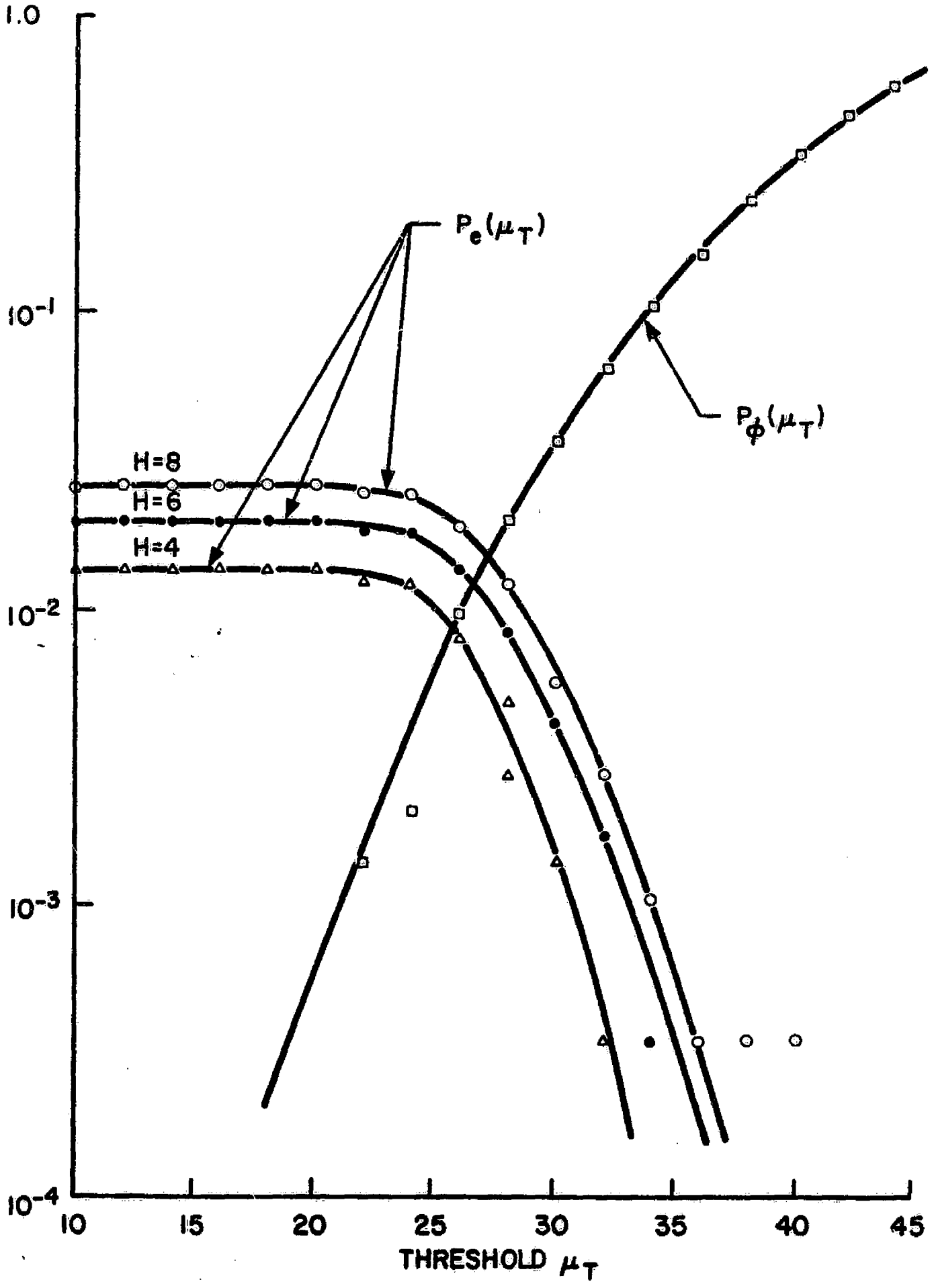


FIGURE 7.1 SIMULATION RESULTS FOR $P_e(\mu_T)$ AND $P_\phi(\mu_T)$ FOR $(U_B/N_o)_1$ =1.25db, H=4,6 AND 8

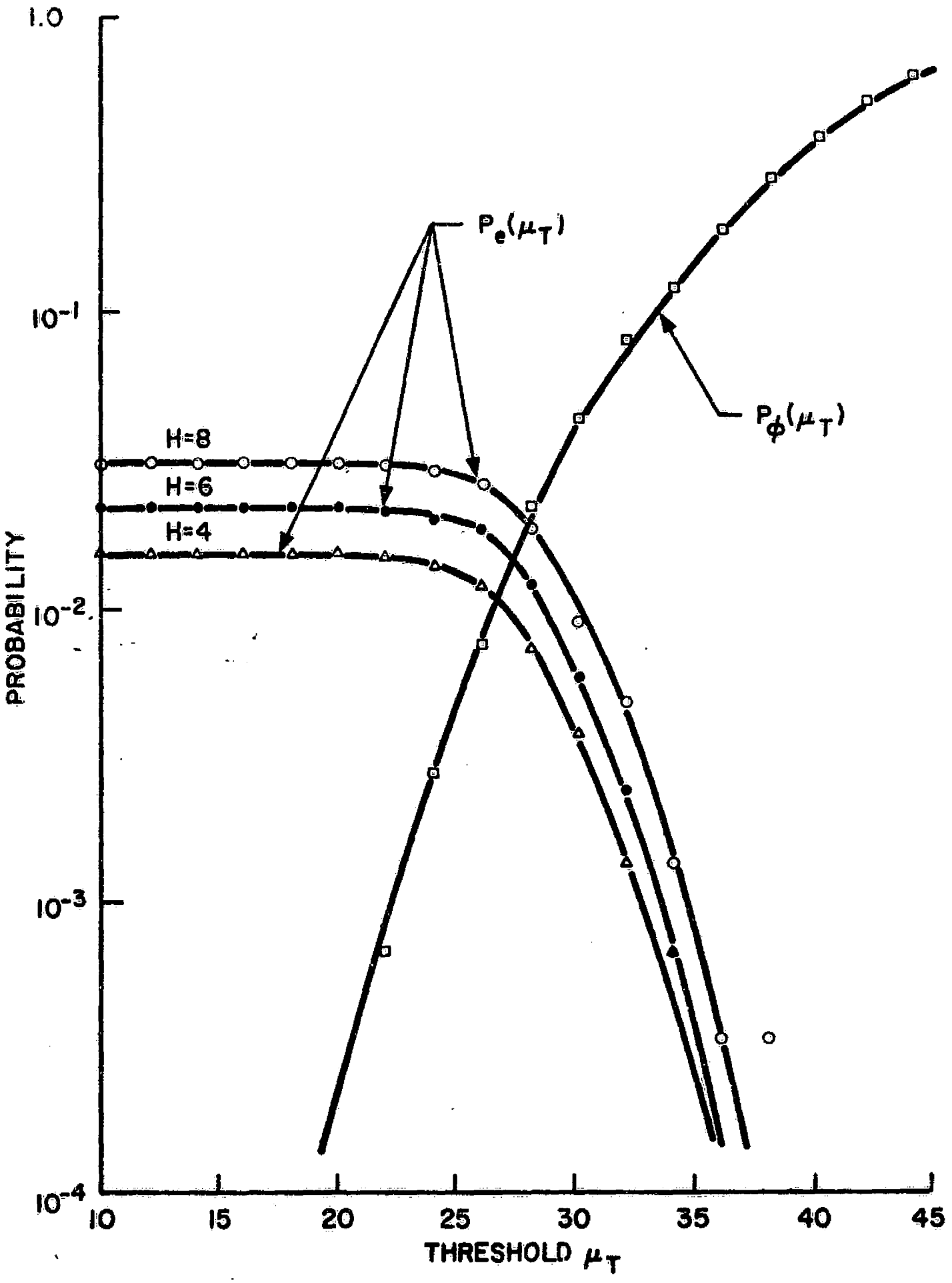


FIGURE 7.2 SIMULATION RESULTS FOR $P_e(\mu_T)$ AND $P_\phi(\mu_T)$ FOR $(U_B/N_o)_I = 1.0$ db, FOR H=4,6 AND 8

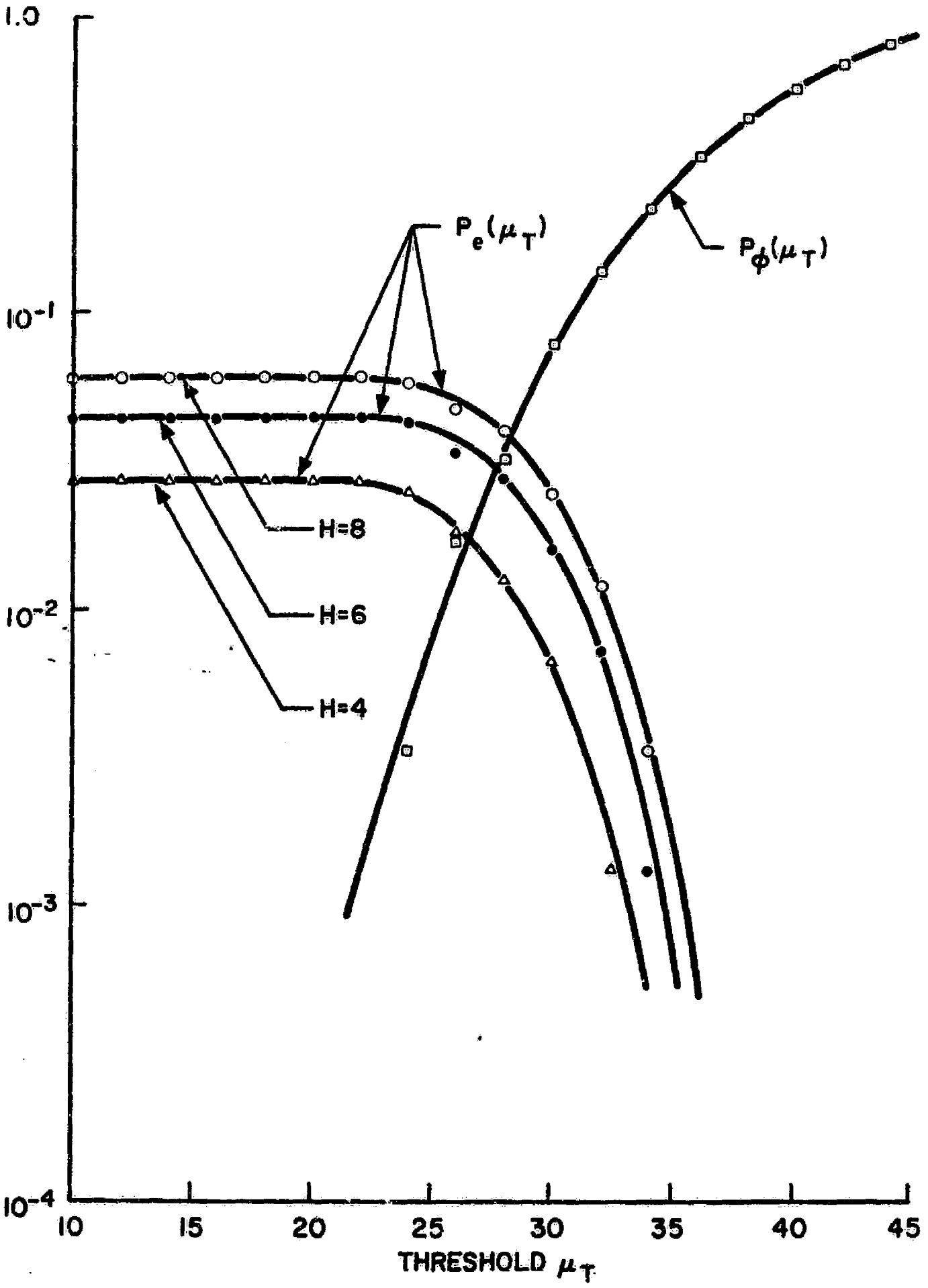


FIGURE 7.3 SIMULATION RESULTS FOR $P_e(\mu_T)$ AND $P_\phi(\mu_T)$ FOR $(U_B/N_o)_1$=0.5 db, FOR H=4,6 AND 8

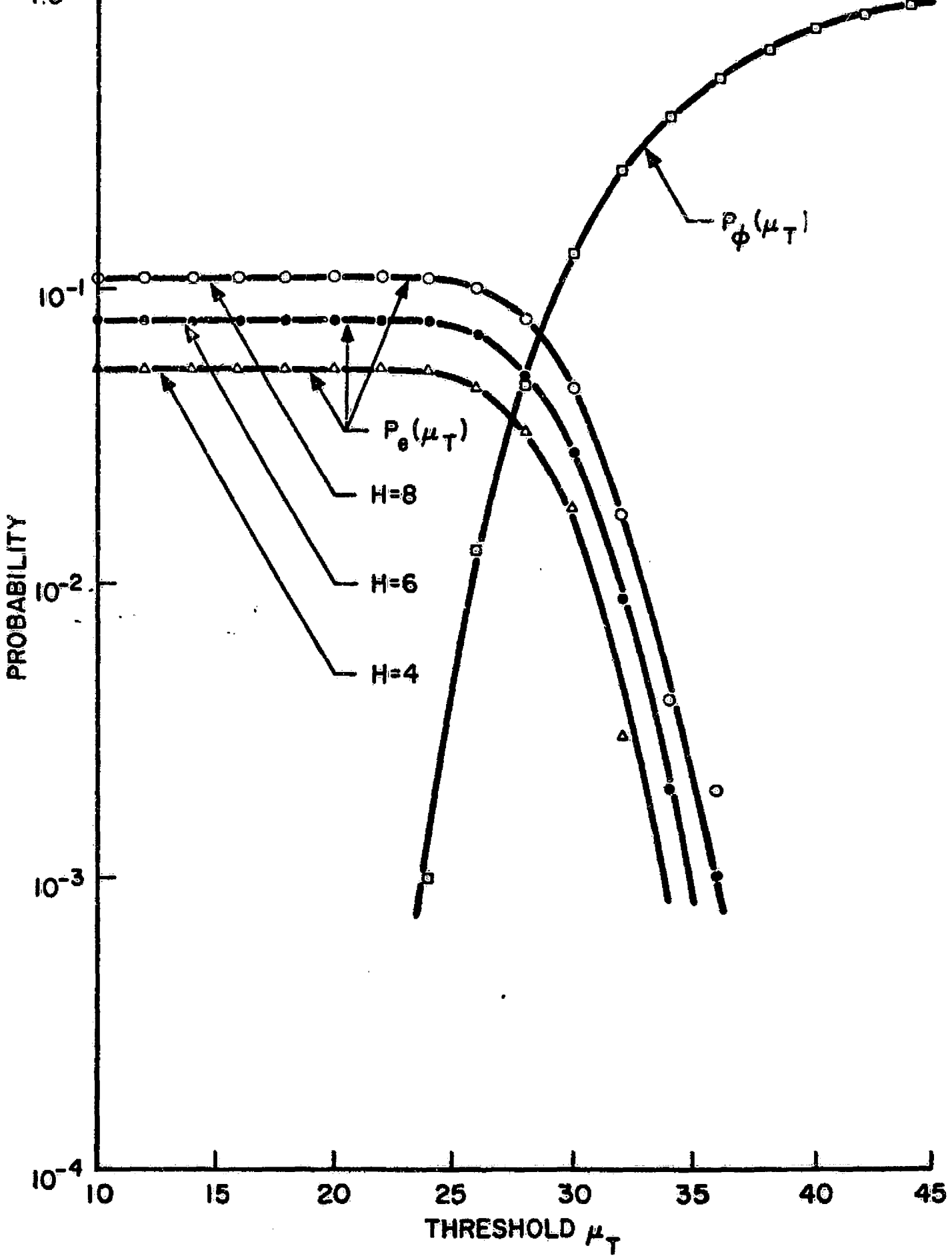


FIGURE 7.4 SIMULATION RESULTS FOR $P_e(\mu_T)$ AND $P_\phi(\mu_T)$ FOR $(U_B/N_o)_1$=0db, FOR H=4,6 AND 8

TABLE 7.1 – NUMBER OF TRIALS FOR EACH CASE SIMULATED

$(U_B/N_o)_I$ (dB)	Number of Trials
1.25	3000
1.0	7600
0.5	1500
0	1000

7.2 Application of Simulation Results to System Performance

7.2.1 Threshold Erasure Decoder

The overall probability of failure of the outer decoder was bounded by (4.6), or more weakly by (4.11), for the case of the Threshold Erasure decoder:

$$P_E \leq \sum_{N_e=0}^{N_o} \sum_{N_\varphi=D-2N_e}^{M} \binom{M}{N_\varphi\ N_e} P_e^{N_e} P_\varphi^{N_\varphi} (1-P_e-P_\varphi)^{M-N_e-N_\varphi}$$

$$+ \sum_{N_e=N_o+1}^{M} \binom{M}{N_e} P_e^{N_e} (1-P_e)^{M-N_e} \qquad (4.6)$$

where D is the minimum distance of the outer algebraic code, M is the total number of parallel inner channels (the length of the outer algebraic codewords), N_e is the number of errors among the M 2^H-ary symbols, N_φ is the number of erasures, and $M-N_e-N_\varphi$ is the number of

correctly decoded symbols produced by the inner decoder. N_o is the largest integer such that $2N_o < D$.

Although not explicitly indicated, P_e and P_v are functions of the threshold, μ_T. Thus, the error probability P_E may be minimized by proper choice of the threshold.

As described in Chapter 3, for Reed-Solomon codes,

$$D = P + 1 \tag{7.1}$$

where there are I incoming information channels, P parity channels are generated by the algebraic outer encoder, and M = P + I. For non-RS BCH codes, $D < P + 1$; see Peterson [1961] for a tabulation of binary BCH codes and their minimum distances for lengths up through 255. The non-RS BCH codes yield performance inferior to that of the RS codes, and so their use will not be considered further.

If $(U_B/N_o)_I$ is the ratio of signal energy to noise power density per information bit for the inner superchannel (see Figure 1.2), then the overall signal energy to noise density ratio is

$$\left(\frac{U_B}{N_o}\right)_o = \left(\frac{U_B}{N_o}\right)_I \left(\frac{M}{I}\right) = \left(\frac{U_B}{N_o}\right)_I \left(1 + \frac{P}{I}\right) \tag{7.2}$$

The overall signal energy to noise power density ratio will be the independent parameter against which all the overall performance results

will be plotted and compared. The corresponding overall information rate is given by

$$R = \frac{1}{v}\left(\frac{I}{M}\right) \tag{7.3}$$

where $1/v$ is the rate of the convolutional code, here taken to be $1/3$.

Figure 7.5 shows the nature of the optimization of system performance over the range of threshold μ_T, for the case of H = 8, M = 255, RS outer code, $(U_B/N_o)_I$ = 1.0 dB, for various numbers of parity channels, P, and therefore for various $(U_B/N_o)_o$ from (7.2). Also shown on each curve is the overall rate R in bits per use.

Plotting the minima of such curves as those in Figure 7.5 for the four values of $(U_B/N_o)_I$ gives the overall error probability performance curves of Figure 7.6. The abscissa is overall signal energy to noise power density ratio; the ordinate is the logarithm of the probability of failure of the outer decoder. The curves are, of course, the loci of sets of discrete points corresponding to specific numbers of parity channels P in (7.2). At selected points the maximum number of correctable errors is shown parenthetically, where $(N_e)_{max}$ is the largest integer equal to or less than (D-1)/2. For Reed-Solomon codes, (D-1)/2 = P/2. As will be discussed later, the outer decoder complexity is related to $(N_e)_{max}$.

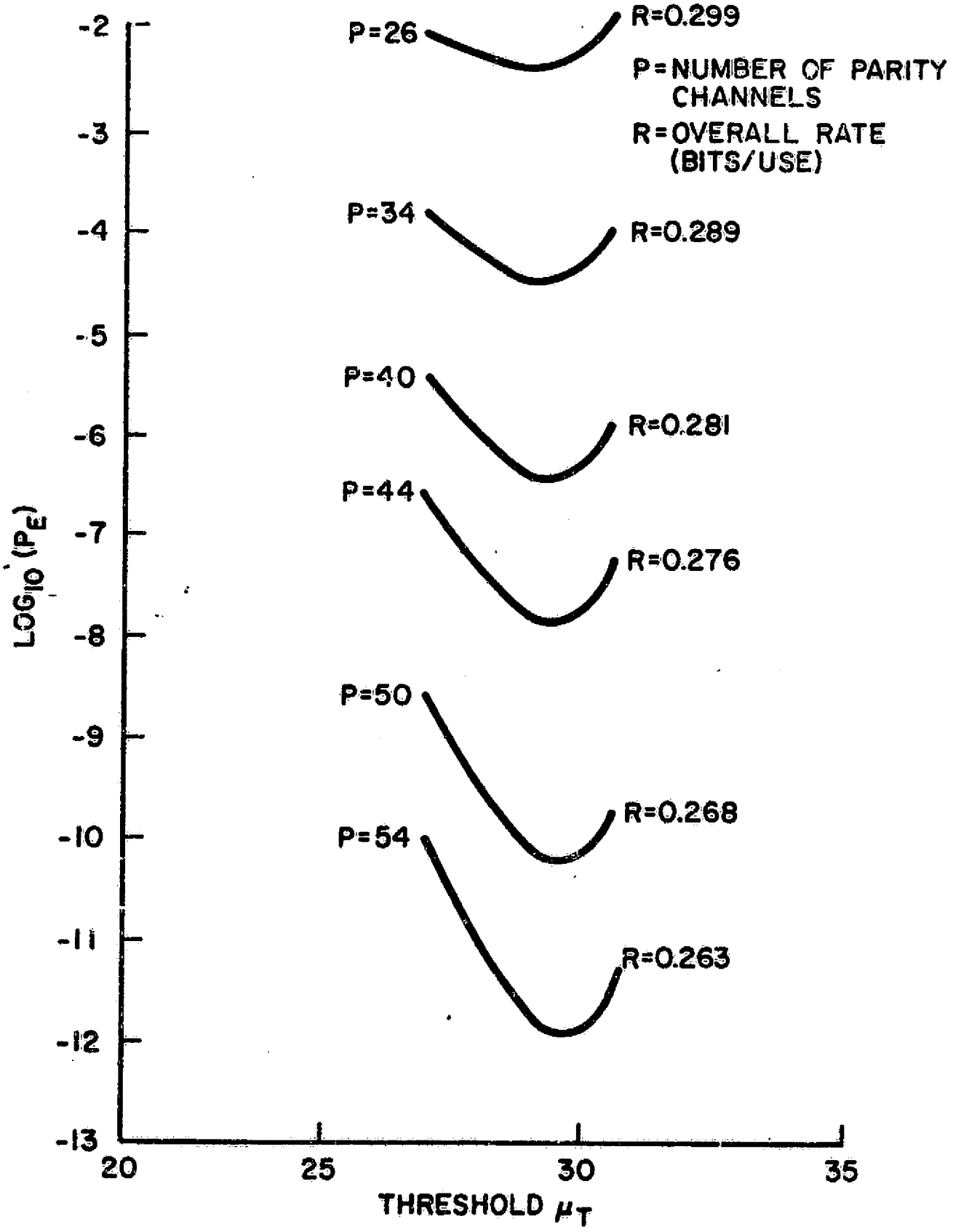


FIGURE 7.5 OPTIMIZATION OF THRESHOLD ERASURE DECODER FOR $M=255$, $(U_B/N_o)_1 = 1.0DB$, $H=8$

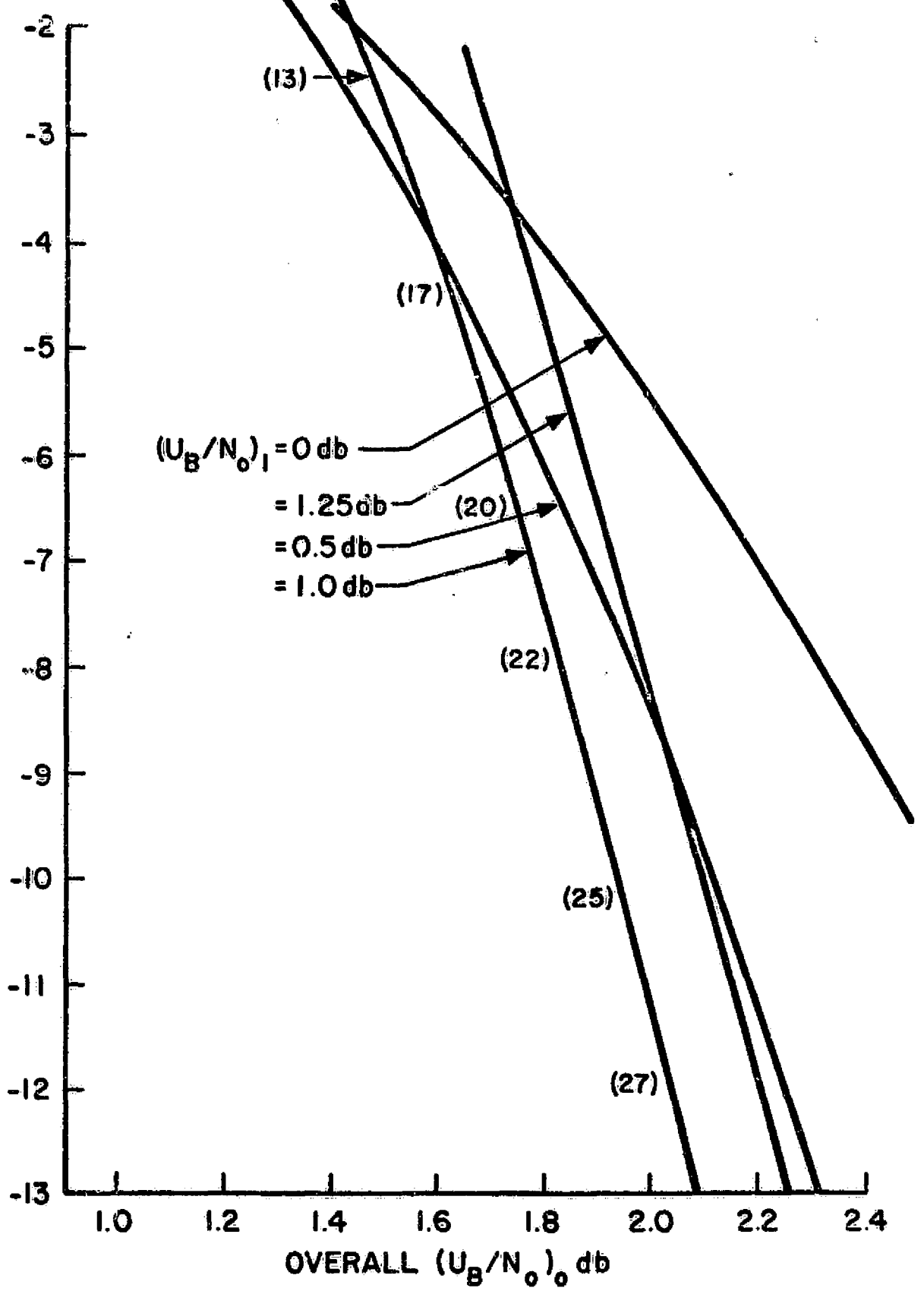


FIGURE 7.6 OVERALL ERROR PROBABILITY OF THRESHOLD ERASURE DECODER FOR H=8, M=255

7.2.2 Genie-Augmented Decoder

For each of the M = I + P inner channels the Threshold Erasure decoder considers that state among the $2^{K_2-1} = 128$ states which has the maximum metric. If the maximum metric exceeds the threshold, the first H decoded data symbols are accepted as a 2^H-ary symbol for the outer decoder; otherwise an H-bit erasure symbol is provided to the decoder. If the maximum state metric passes the threshold test and yet the first H symbols contain a decoding error, the outer decoder has an error which must be corrected. This it can do provided the number of errors N_e and the number of erasures N_φ satisfy (3.22):

$$N_\varphi + 2N_e < D \tag{3.22}$$

Given the inner convolutional encoder-decoder and the outer algebraic encoder-decoder, the best the system could operate would be that realized if the threshold detector were replaced by a magic genie who would automatically erase the first H symbols whenever a decoding error is contained in the first H symbols of the maximum-metric-state sequence. Thus, every error would become an erasure and correct decoding would result if $N_e < D$. In this case

$$P_\varphi = P_e(-\infty) = Pr\{\text{decoding error in first H symbols}\} \tag{7.4}$$

$$P_e = 0$$

Using these values in (4.6) gives the results shown in Figure 7.7 together with the Threshold Erasure decoder results shown dotted.

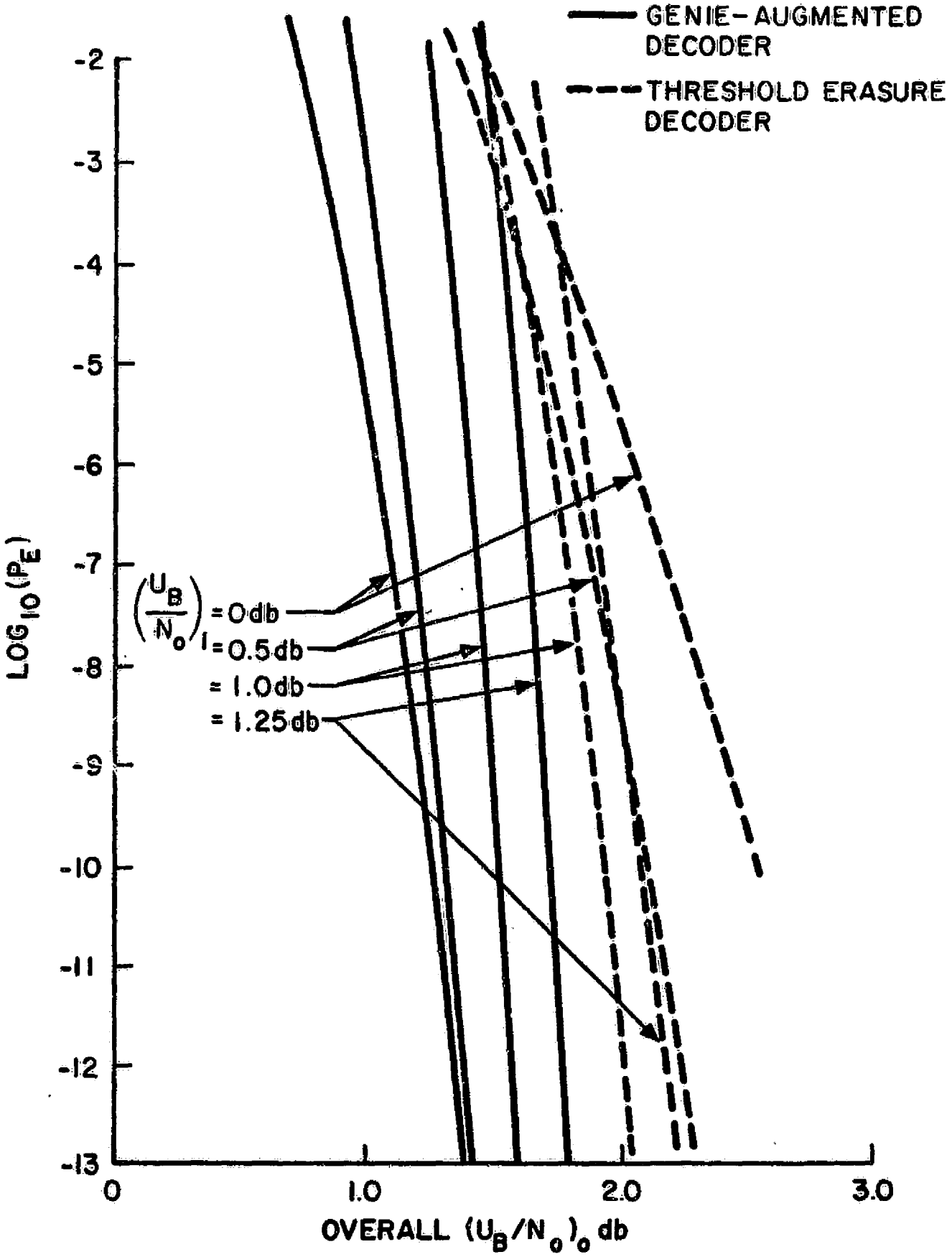


FIGURE 7.7 COMPARISON OF OVERALL ERROR PROBABILITY OF GENIE-ANGMENTED AND THRESHOLD ERASURE DECODERS FOR H=8, M=255

At high values of $(U_B/N_o)_I$, where the distributions of the maximum metric with and without error in the first H symbols tend to separate from one another (see Figure 5.2), the threshold does a good job of detecting errors. At $(U_B/N_o)_I$ = 1.25 dB it is seen that about 0.2 to 0.3 dB is lost due to the threshold detector. At $(U_B/N_o)_I$ = 1.0 dB, the loss ranges from 0.25 to 0.4 dB, and at $(U_B/N_o)_I$ = 0.5 dB from 0.65 to 0.8 dB. This decoder loss was one of the motivations for investigating the Fixed Erasure and GMD decoders.

7.2.3 Performance at Rates Above R_{COMP}

Before leaving the Threshold Erasure decoder it is interesting to relate the overall error probability and rate performance to the R_{COMP} of sequential decoding and to channel capacity C. Channel capacity for each inner channel including the output quantizer is

$$C = \sum_{j=1}^{8} P(j|o) \log \left(\frac{2P(j|o)}{P(j|o)+P(j|1)} \right) \qquad (7.5)$$

where the $P(j|k)$ are the transition probabilities (3.10) to (3.15), listed in Table 7.2 for the four cases of the simulation. The computational cutoff rate for sequential decoding, above which the expected number of decoding computations per decoded symbol becomes unbounded, is

$$R_o = R_{COMP} = E_o(1) = \log 2 - \log \left[1 + 2 \sum_{j=1}^{4} \sqrt{P(j|o)\, P(j|1)} \right] \qquad (7.6)$$

The capacity and R_{COMP} are listed in Table 7.2 for each case, in bits per channel use.

TABLE 7.2 – TRANSITION PROBABILITIES, CAPACITY AND R_{COMP} FOR EACH CASE SIMULATED

	$(U_B/N_o)_I$			
	1.25 dB	1.0 dB	0.5 dB	0 dB
P(1|0) = P(8|1)	0.318693	0.323402	0.332712	0.341533
P(2|0) = P(7|1)	0.181307	0.176598	0.167289	0.158467
P(3|0) = P(6|1)	0.181307	0.176598	0.167289	0.158467
P(4|0) = P(5|1)	0.145779	0.143677	0.139157	0.134445
P(5|0) = P(4|1)	0.094264	0.095100	0.096292	0.096772
P(6|0) = P(3|1)	0.048945	0.051211	0.055425	0.059095
P(7|0) = P(2|1)	0.020506	0.022433	0.026535	0.030616
P(8|0) = P(1|1)	0.009199	0.010981	0.015301	0.020605
Capacity C*	0.440	0.423	0.388	0.355
R_{COMP}*	0.275	0.261	0.234	0.210

*C and R_{COMP} given in bits/use.

Figures 7.8 and 7.9 are replots of the error performance of the Threshold Erasure decoder as a function of the effective rate (7.3) normalized to the channel capacity, with R_{COMP}/C indicated. For probabilities of outer decoder failure of about 10^{-6}, it is seen that rates about 8 percent above R_{COMP} can be achieved.

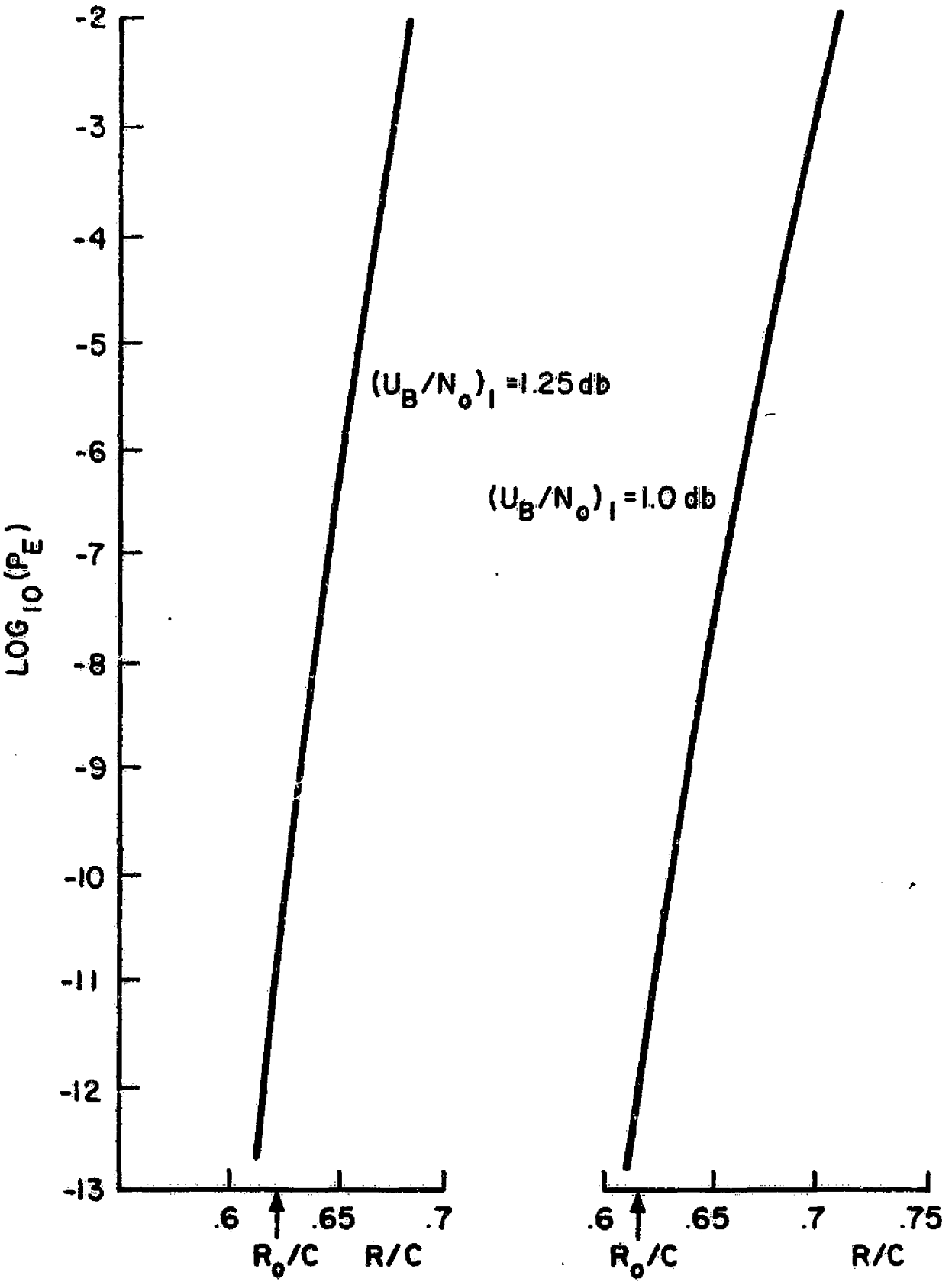


FIGURE 7.8 THRESHOLD ERASURE DECODER ERROR PROBABILITY RELATED TO CAPACITY C AND $R_o = R_{COMP}$ FOR H=8, M=255

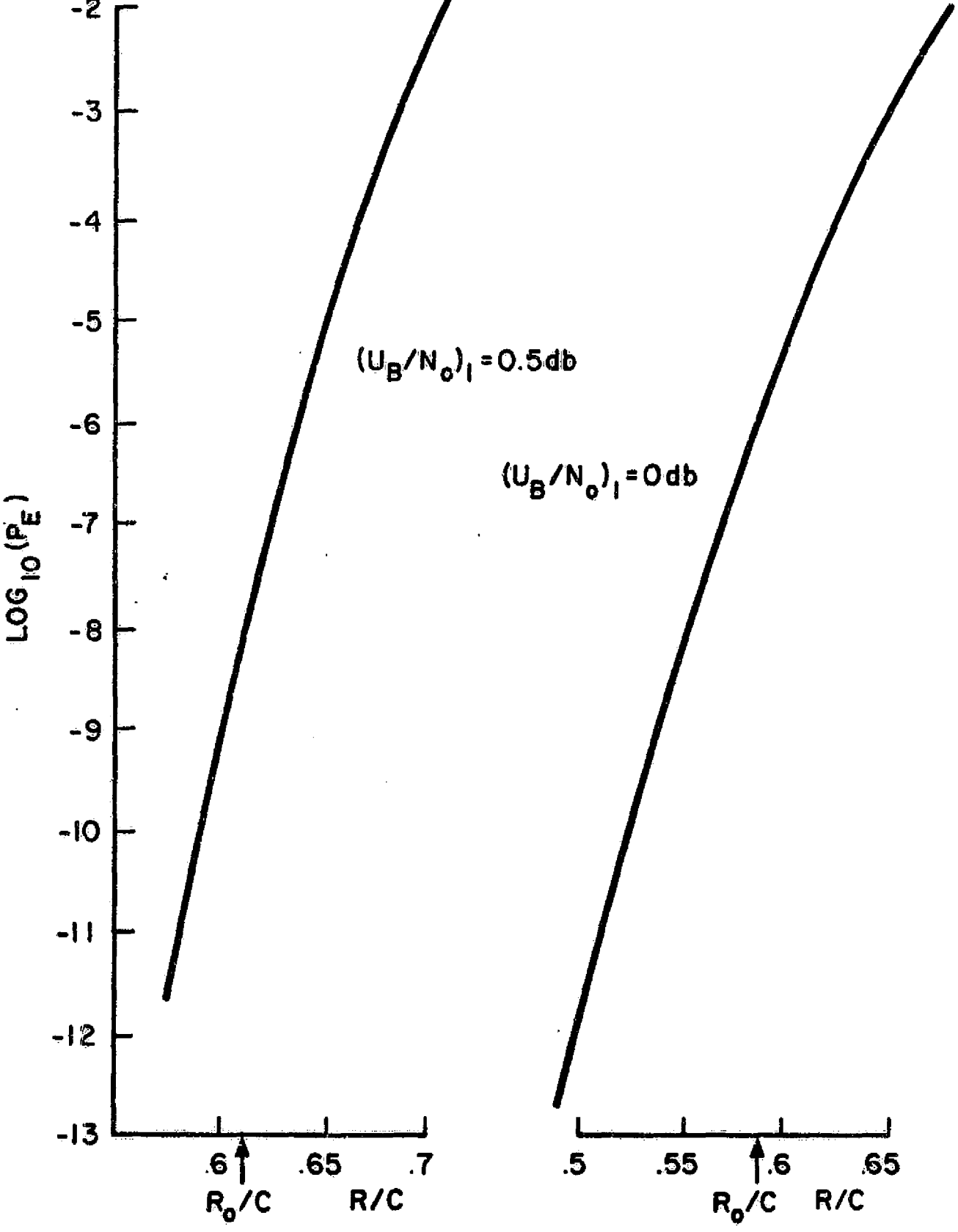


FIGURE 7.9 THRESHOLD ERASURE DECODER ERROR PROBABILITY RELATED TO CAPACITY C AND $R_o = R_{COMP}$ FOR H=8, M=255

7.2.4 Fixed Erasure Decoder

Recall that the Fixed Erasure decoder erases the 2^H-ary symbols of the N_φ inner channels whose maximum state metrics are lowest, passing on the remaining $M - N_\varphi$ 2^H-ary symbols to the outer algebraic decoder. The bound on probability of outer decoder failure is from (4.16)

$$P_E \leq \sum_{N=N_e+1}^{M-N_\varphi} \binom{M}{N \; N_\varphi-1 \; M-N_\varphi-N} \cdot \int_0^1 P_e{}^N(\xi) \left[1-P_e(\xi)-P_\varphi(\xi)\right]^{M-N_\varphi-N} P_\varphi(\xi)^{N_\varphi-1} \, dP_\varphi(\xi) \quad (4.16)$$

where N_e is the maximum number of correctable errors; i.e., the largest integer such that $2N_e < D - N_\varphi$. For a given $(U_B/N_o)_I$, that is for given functions $P_e(\mu_T)$ and $P_\varphi(\mu_T)$, the probability of error is a function of N_φ, with respect to which it can be minimized. Figure 7.10 shows how the bound on P_E varies with N_φ for several values for P, the number of parity channels, with M = 255, H = 8, $(U_B/N_o)_I$ = 1.0 dB, and using a Reed-Solomon outer code with D = P + 1. The point N_φ = o corresponds to errors-only decoding. The improvement in error performance resulting from using an optimum Fixed Erasure errors-and-erasures decoder is evident.

Figure 7.11 shows a plot of the error probability for the optimum N_φ using the data in Figures 7.1 through 7.4 for H = 8. Also shown as dotted curves are the Threshold Erasure decoder results from Figure 7.6

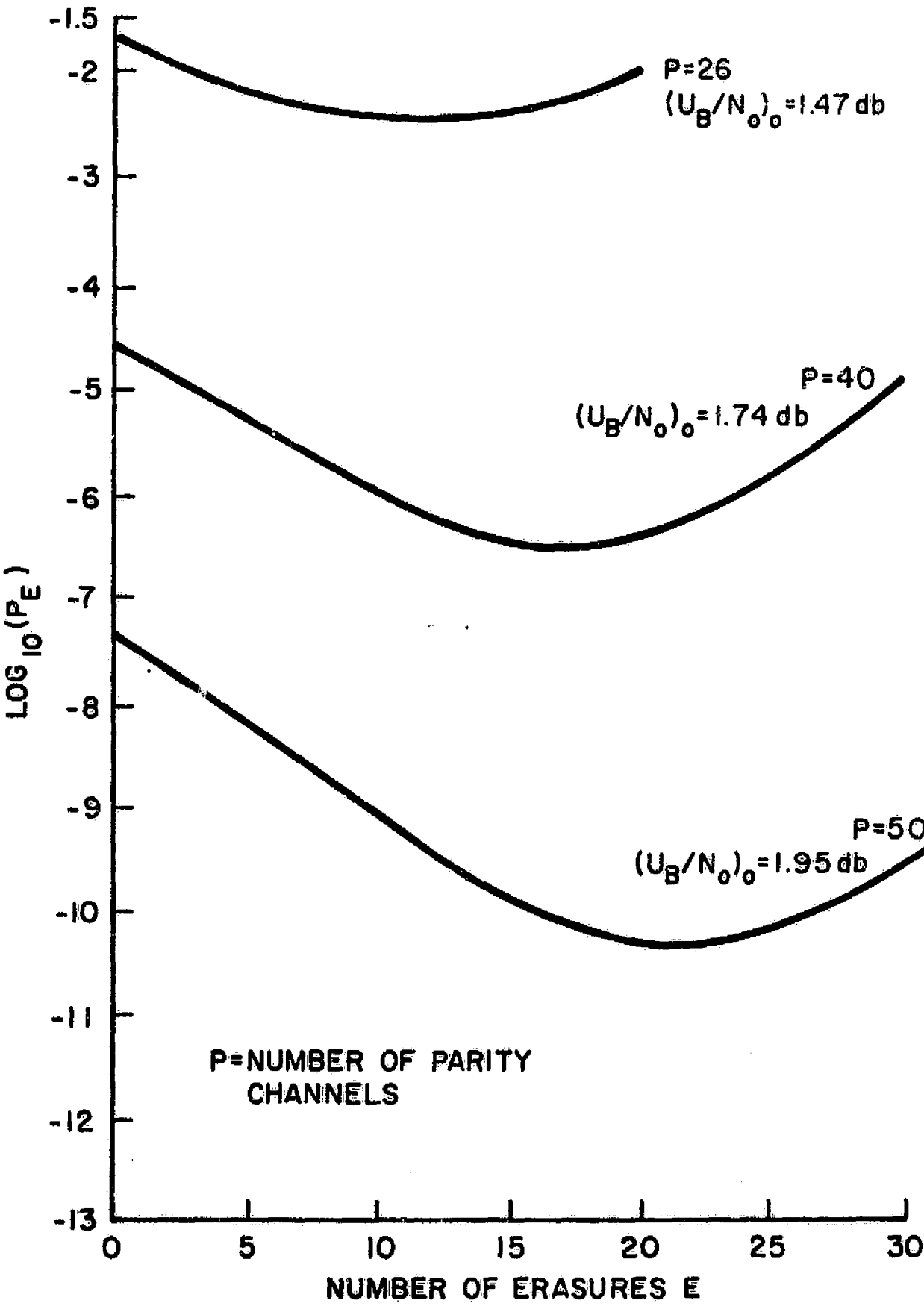


FIGURE 7.10 OPTIMIZATION OF FIXED ERASURE DECODER FOR $M=255$, $(U_B/N_0)_I=1.0DB$, $H=8$

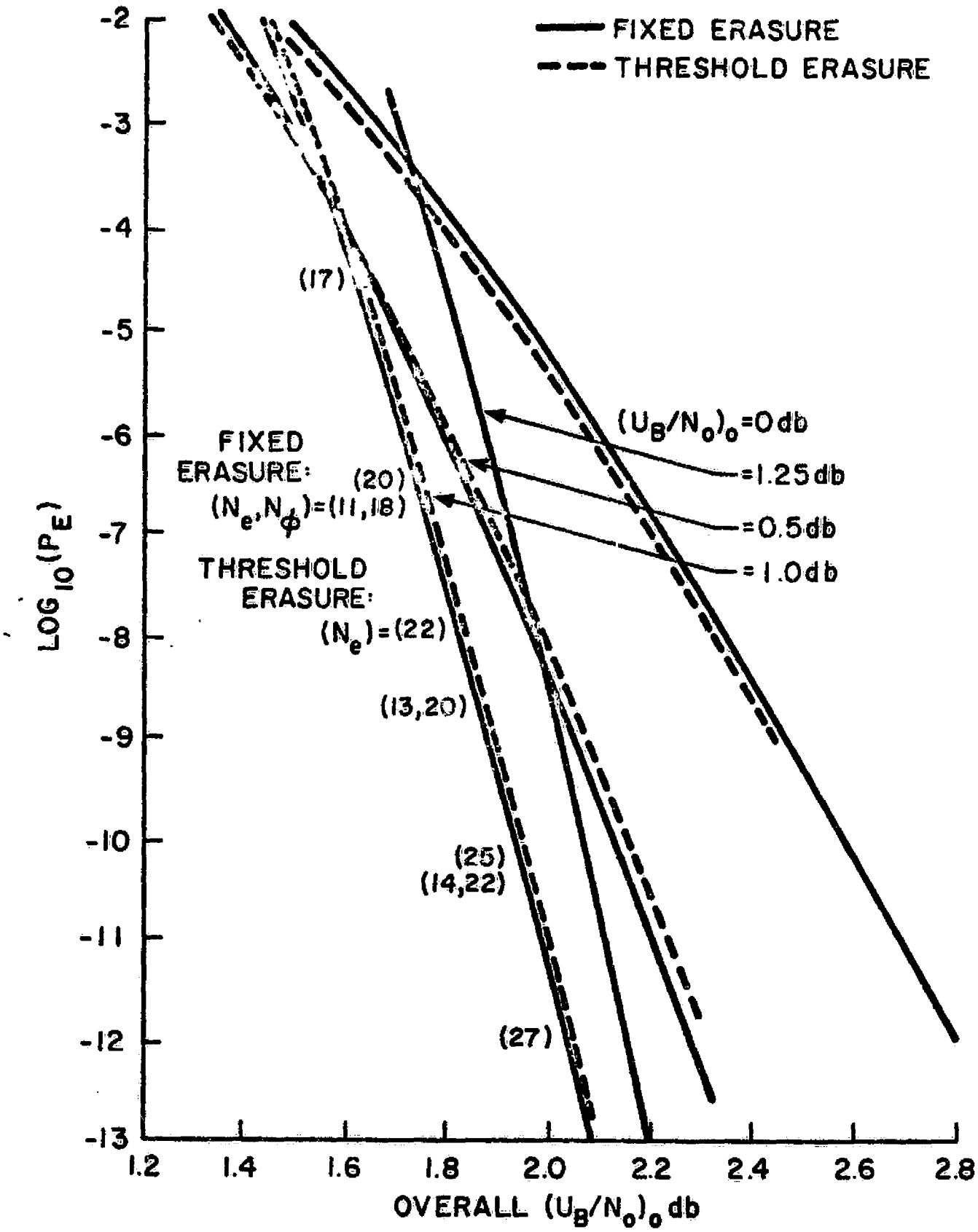


FIGURE 7.11 COMPARISON OF ERROR PROBABILITY OF FIXED AND THRESHOLD ERASURE DECODERS FOR H=8, M=255

for comparison. The two decoders, when optimized with respect to the threshold and the number of erasures, respectively, yield virtually identical error performance.

The maximum error correction capability required of the outer decoder is shown parenthetically, together with the optimum number of erasures, at a selected number of points along the curves. Also shown for comparison is the maximum error correction capability required for the Threshold Erasure decoder.

7.2.5 GMD Decoder

In the case of the GMD decoder each of the M inner channels provides its maximum-metric-state metric and initial 2^H-ary symbol to the outer GMD decoder. The GMD decoder transforms each metric according to the nonlinear function $\alpha(\mu)$, in the present case the piecewise linear function plotted in Figure 3.3. Then, by a technique described in detail in Forney [1966], in at most (D+1)/2 successive errors-and-erasures decoding cycles a decoded word will be found, provided a possible transmitted word lies close enough to the received word using the measure J defined in (3.28).

The GMD decoder is assured of success if inequality (3.28) is satisfied for the correct codeword $\underline{x}$. Similarly, the Threshold Erasure and Fixed Erasure decoders are assured of success if inequality (3.22) is satisfied. For the latter two decoders, error bounds (4.6) and (4.16) are exact evaluations of the probability that inequality (3.22) is violated, whereas for the GMD decoder, the best we have is a Chernoff bound (4.32) that inequality (3.28) fails to be satisfied. It

is the exact bounds (4.6) and (4.16) that were used to obtain the curves of Figures 7.6, 7.8, 7.9 and 7.11.

The probability of GMD decoder failure to decode, or of decoding the wrong word, is bounded by the Chernoff bound (4.32):

$$P_E \leq \min_{s<o} e^{-M[s(1-\delta)-\mu(s)]} \tag{4.32}$$

where $\delta = D/M$ and $\mu(s) = \ell n\, g(s)$, defined by (4.24). Computing $\mu(s)$ from (4.29) using the simulation data for the case $(U_B/N_o)_I = 1.0$ dB, and optimizing over $s < o$ and the parameters μ_u and μ_ℓ of the nonlinear function $\alpha(\mu)$ of Figure 3.3, one obtains the results shown in Figure 7.12. The Threshold Erasure decoder results are again shown dotted for comparison, although here the comparison is only qualitative for the reasons mentioned above. As discussed in Chapter IV, the GMD decoder includes the Threshold Erasure decoder as a limiting case, so the actual optimized GMD performance must be better.

Figure 7.13 shows the optimum $\alpha(\mu)$ obtained for three values of overall $(U_B/N_o)_o$, both for the GMD decoder and for the limiting case of Threshold Erasure decoding, using the Chernoff bound (4.32) (or equivalently (4.11)) for the Threshold Erasure case. The optimum threshold derived using the Chernoff bound is almost exactly the same as derived using the more exact error expression (4.6). The error probability bounds for the GMD decoder are seen to be slightly better than for the Threshold Erasure decoder.

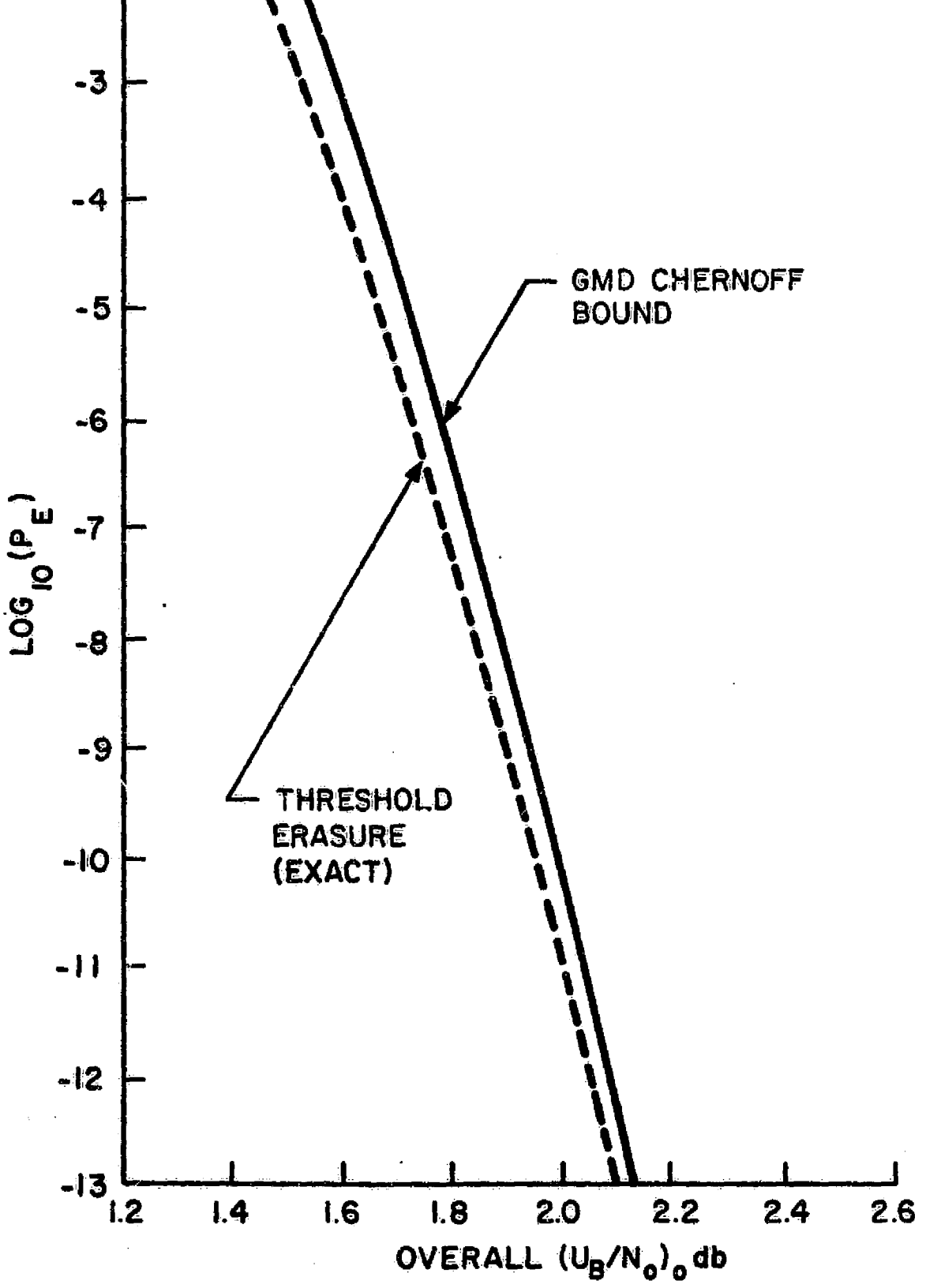
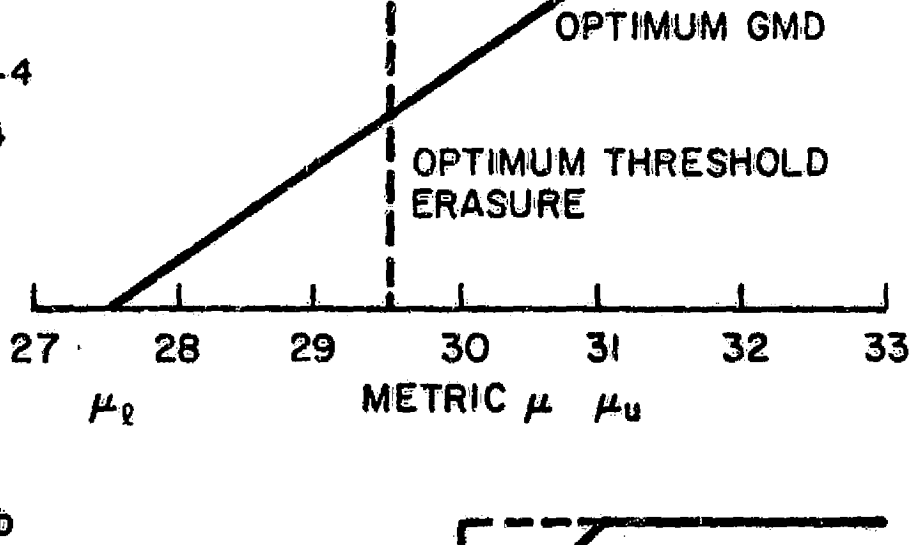
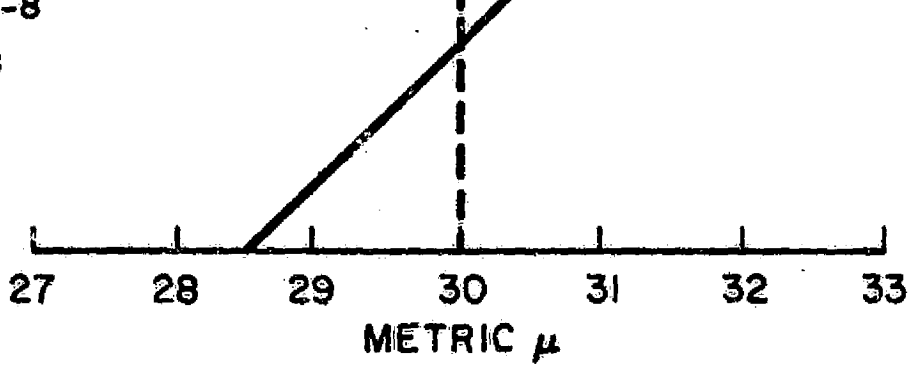


FIGURE 7.12 CHERNOFF BOUND ON OVERALL ERROR PROBABILITY OF GMD DECODER FOR $(U_B/N_o)_I$ = 1.0 db, H=8, M=255

$(U_B/N_o)_o$=1.64 db
P=35
GMD $P_E<1.4\times10^{-4}$
TE $P_E<1.6\times10^{-4}$



$(U_B/N_o)_o$=1.84 db
P=45
GMD $P_E<4.5\times10^{-8}$
TE $P_E<8.1\times10^{-8}$



$(U_B/N_o)_o$=2.05db
P=55
GMD $P_E<2.3\times10^{-12}$
TE $P_E<6.7\times10^{-12}$

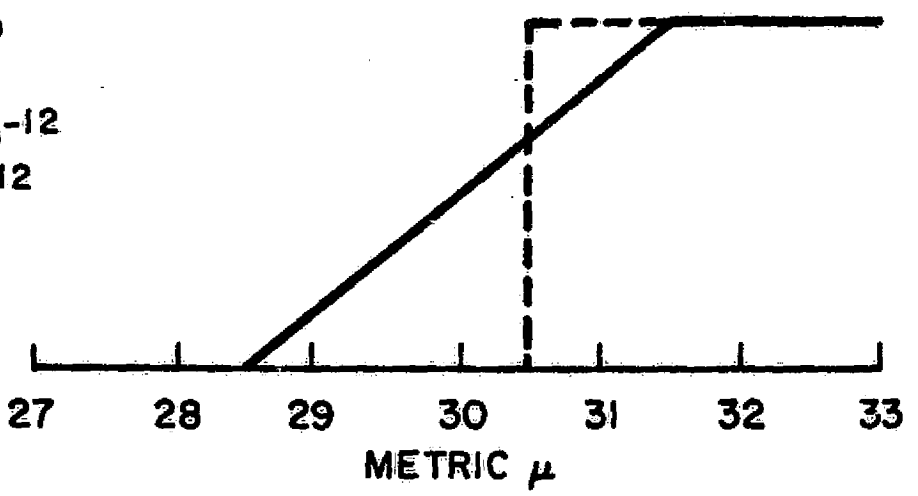


FIGURE 7.13 OPTIMUM $\alpha(\mu)$ DERIVED FROM CHERNOFF BOUNDS FOR GMD AND THRESHOLD ERASURE DECODERS FOR M=255, $(U_B/N_o)_l$=1.0 db

7.3 Comparison of Threshold Erasure, Fixed Erasure and GMD Decoders

For the reasons mentioned in the previous section, a more valid comparison among the three decoders would result if the Chernoff bounds of (4.11) and (4.21) were used for the Threshold Erasure and Fixed Erasure decoders, respectively, rather than the "exact" bounds of (4.6) and (4.16). These three bounds are presented in Figure 7.14 for the case H = 8 and $(U_B/N_o)_I$ = 1.0 dB.

The bounds in Figure 7.14 show virtually the same performance for all three decoders when the Threshold and Fixed Erasure decoders are each optimized. The GMD decoder is slightly better, as we know it must be if evaluated exactly.

The error correction capability required of the outer errors-and-erasures algebraic decoder is shown parenthetically along the curves, where the left-hand set of values applies to both the GMD and Threshold Erasure decoders and the right-hand set applies to the Fixed Erasure decoder. If N_e is the maximum error correction capability required of the outer BCH decoder, the decoding algorithms of Forney [1966] require a complexity varying as N_e^3. A more recent algorithm of Berlekamp's (Gallager [1968]) requires complexity varying as only N_e^2. If the bounds of Figure 7.14 can be taken to imply approximately equal actual performance of all three decoders, then the Fixed Erasure decoder has the advantage of requiring roughly only 1/3 to 1/5 the complexity of the other two.

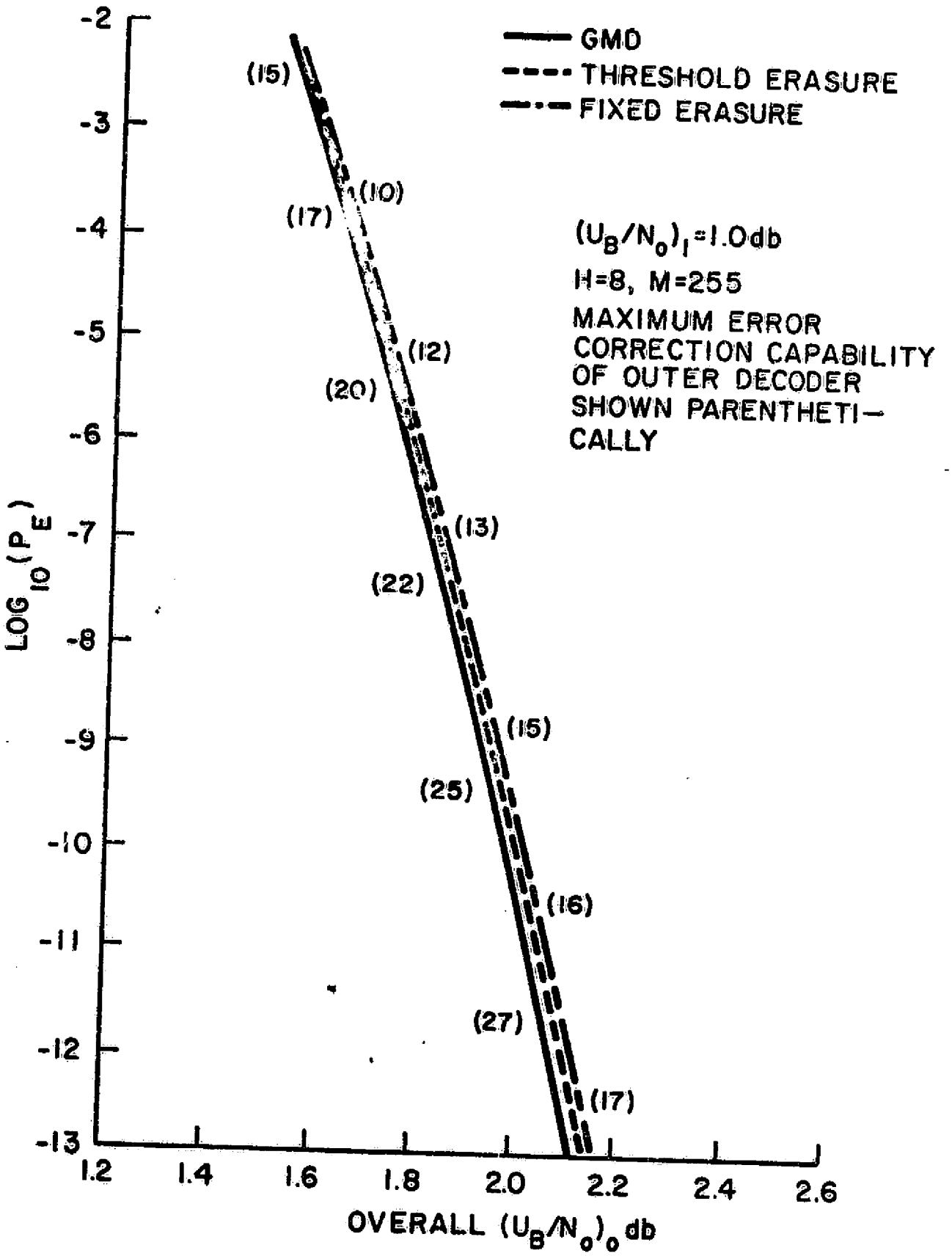


FIGURE 7.14 CHERNOFF BOUNDS ON ERROR PROBABILITY FOR GMD, THRESHOLD ERASURE AND FIXED ERASURE DECODERS

7.4 Comparison with Previous Results

From Figures 7.6 and 7.11 it is seen that the best error performance is achieved for an inner $(U_B/N_o)_I$ = 1.0 dB or slightly less. Odenwalder [1970] found his system performed best with an inner $(U_B/N_o)_I$ = 1.25 dB. Our best results are compared with his in Figure 7.15, where two cases drawn from Forney's [1966] results are shown also. At very low error probabilities we appear to gain about 0.3 dB over Odenwalder, who had gained somewhat more than that amount over Forney.

7.5 Effect of H

The parameter H has a strong influence on overall error performance. The size of the outer alphabet is 2^H, which limits the number of parallel inner channels $M \leq 2^H$ in order to employ maximal Reed-Solomon codes, which in turn limits the number of parity channels P for a given outer normalized rate (M-P)/M. To evaluate the effect of H the raw simulation data for $(U_B/N_o)_I$ = 1.0 dB was resorted for integer H between 4 and 10, the results appearing in Figure 7.16. Using these results, Figure 7.17 shows a plot of the performance of the Fixed Erasure decoder for these values of H using the maximum M and a RS outer code, for $(U_B/N_o)_I$=1.0 dB.

As will be seen in the next chapter, the amount of storage required in the system is approximately proportional to M, so increases exponentially with H. Likewise, the outer decoder complexity varies roughly as the square of the maximum number of correctable errors, which in turn is approximately proportional (for optimum operation) to M. Thus its complexity also increases exponentially with H.

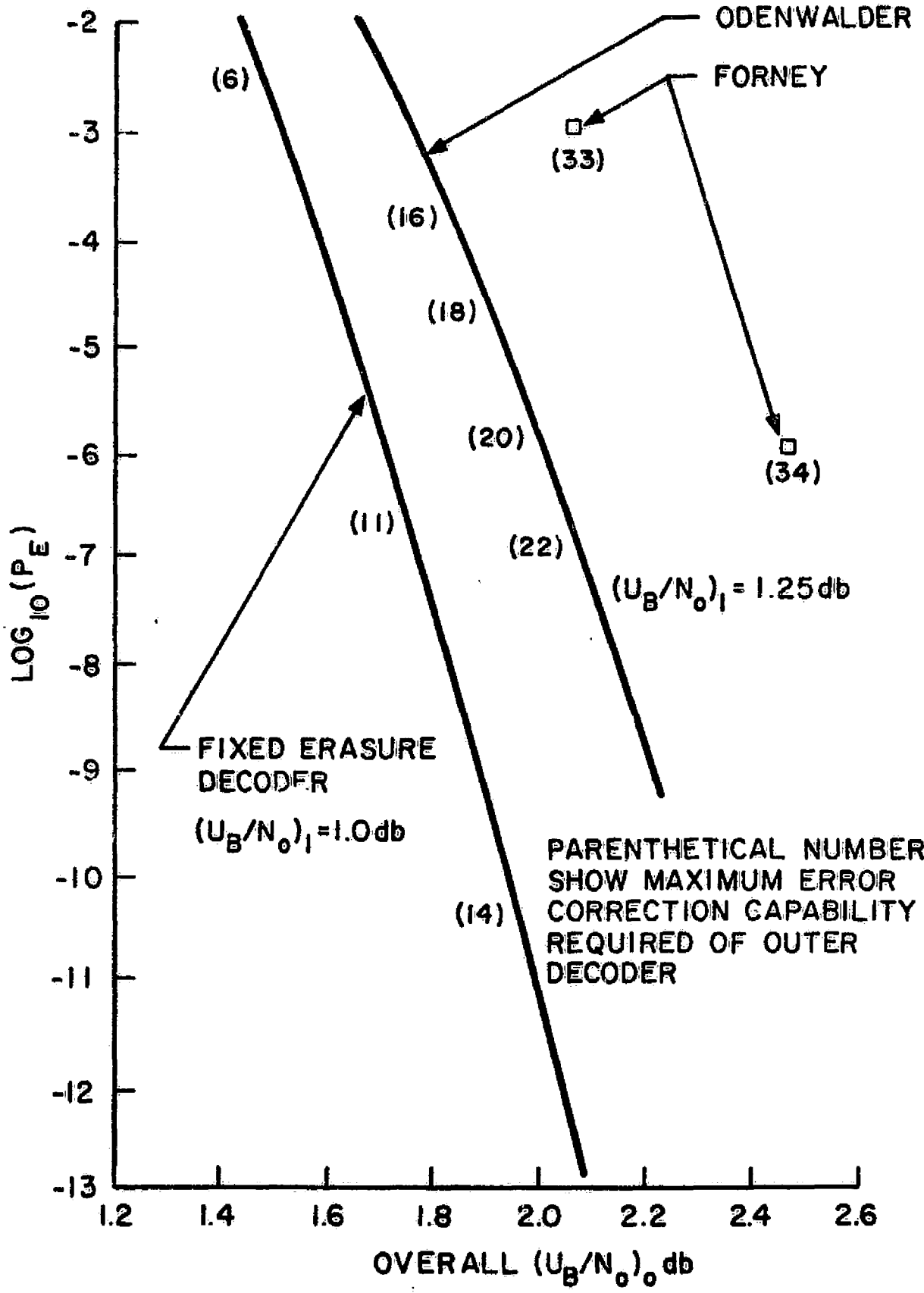


FIGURE 7.15 COMPARISON OF FIXED ERASURE DECODER SYSTEM PERFORMANCE WITH RESULTS OF ODENWALDER AND FORNEY

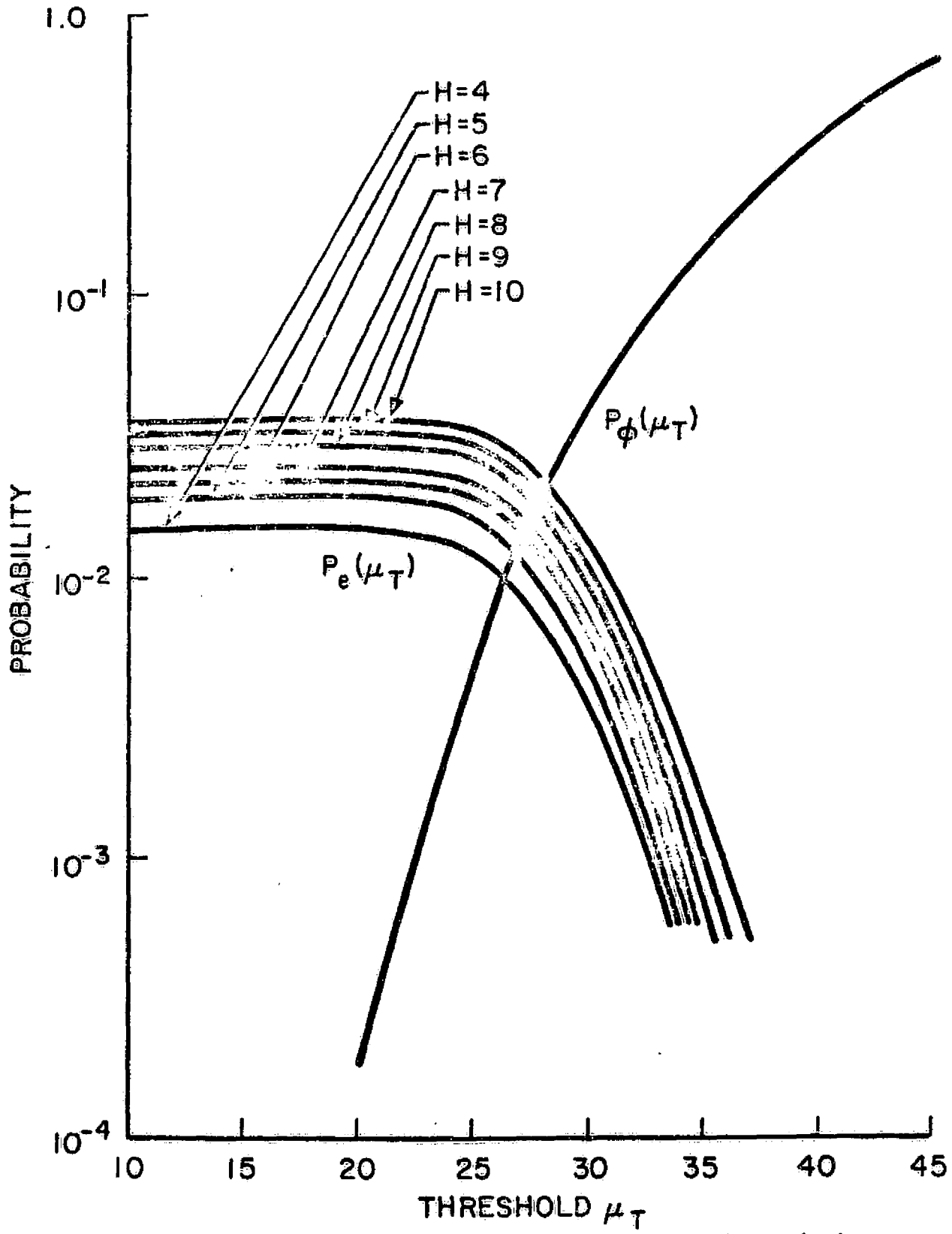


FIGURE 7.16 SIMULATION RESULTS FOR $P_e(\mu_T)$ AND $P_\phi(\mu_T)$ FOR $(U_B/N_o)_1 = 1.0$ db, $4 \leq H \leq 10$

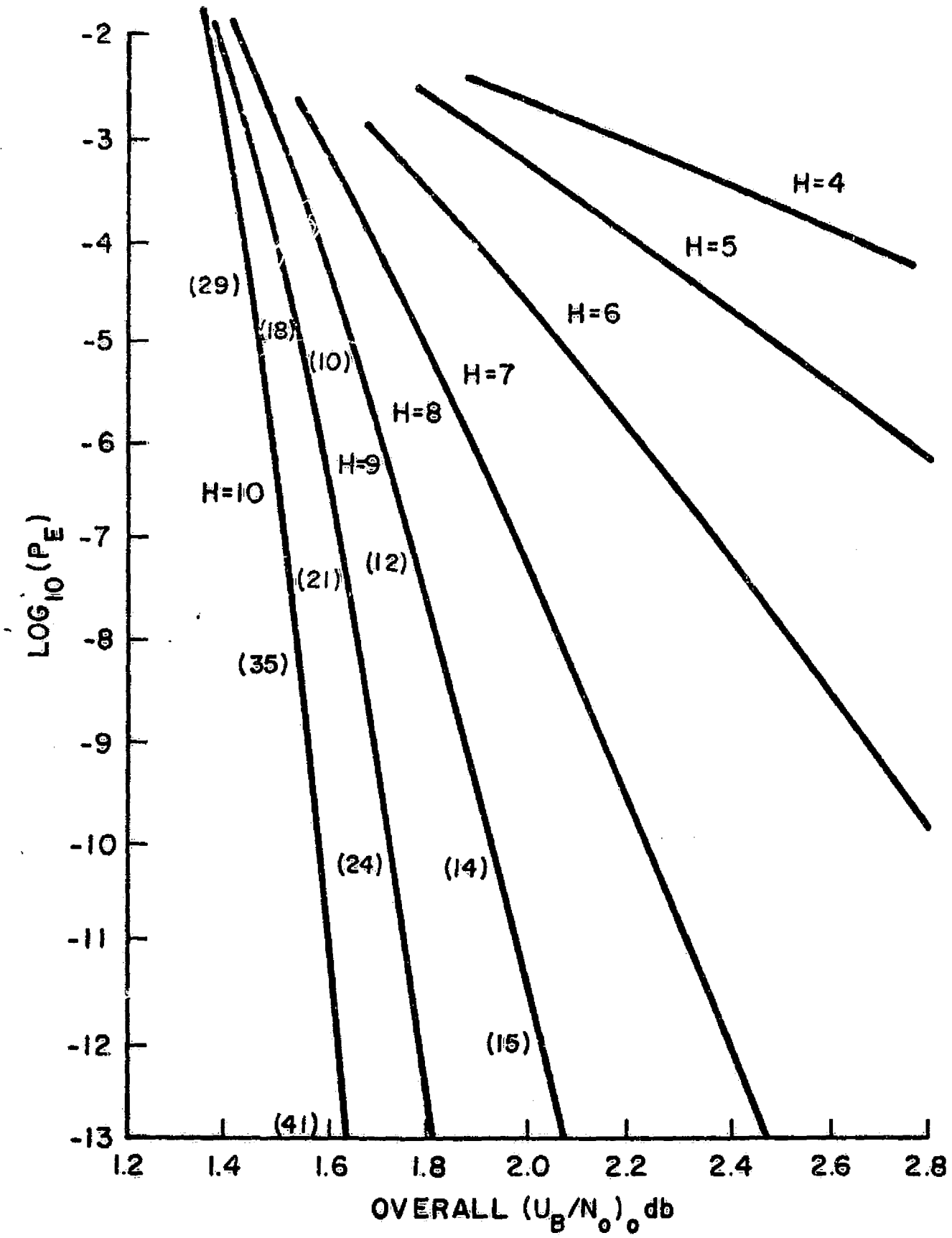


FIGURE 7.17 ERROR PROBABILITY OF FIXED ERASURE DECODER FOR $(U_B/N_o)_1 = 1.0$ db, $4 \le H \le 10$, $M = 2^H - 1$, REED-SOLOMON OUTER CODE

7.6 Quantization Loss

The output of the receiver correlation detector is a continuous random variable whose conditional probability density is given by (3.8). It is impractical to store and process highly accurate approximations to the continuous detector output, and improvement in error performance is dearly bought beyond 8 levels of quantization.

The exact loss due to quantization in any particular implementation would have to be determined experimentally. To obtain an approximate estimate of the loss, one might evaluate R_o (= R_{COMP} for sequential decoding), the break point in the convolutional codes error probability exponent in Figure 1.1, and also equal to the value of the convolutional error exponent itself for rates $R < R_o$.

When a quantizer is used, the channel becomes equivalent to the discrete input-discrete output channel shown in Figure 3.2, with transition probabilities given by (3.10) through (3.13) for a symmetrical quantizer with uniform steps. As mentioned in Chapter III, error performance is not very sensitive to the choice of the quantizer step size in the neighborhood of its optimum (in terms of the measure R_o), which is near one-half the magnitude of the expected correlator output.

From (7.6), R_o for the channel of Figure 3.2 is given by

$$R_o = E_o(1) = 1 - \log_2 \sum_{j=1}^{4} \left[P(j|o) + P(j|1) + 2\sqrt{P(j\ o)\ P(j\ 1)} \right] \qquad (7.6)$$

in bits/use. For the discrete input-continuous output channel up to the input to the quantizer, the corresponding R_o is, using (3.8)

$$R_o = E_o(1) = 1 - \log_2 \int_o^\infty \left[p(z|o) + p(z|1) + 2\sqrt{p(z|o)\ p(z|1)} \right] dz$$

$$= 1 - \log_2 \left(1 + e^{-U_S/N_o} \right)$$

$$= 1 - \log_2 \left(1 + e^{-U_B/N_o} \right) \qquad (7.7)$$

where U_S is the signal energy per channel symbol (3.14) and U_B is the signal energy per information symbol.

Table 7.2 contains a tabulation of the values of R_o from (7.6) for the four values of U_B/N_o used in the simulation. From (7.7) the values of U_B/N_o required to achieve these same R_o's with infinite quantization are readily determined. The results appear in Table 7.3 together with the approximate loss attributable to quantization.

TABLE 7.3 – QUANTIZATION LOSS

U_B/N_o - dB 8-Level Quantizer	R_o Bits/Use	U_B/N_o - dB Infinite Quantization	Approximate Loss (dB)
1.25	0.275	1.07	0.18
1.0	0.261	0.81	0.19
0.5	0.234	0.30	0.20
0.0	.210	-0.23	0.23

7.7 Confidence Intervals

The optimum $(U_B/N_o)_I$ is close to 1.0 dB at very low error probabilities. The greatest amount of simulation data was obtained for this case, as shown in Table 7.1. We consider confidence intervals about $P_e(\mu_T)$ and $P_\varphi(\mu_T)$ for this case, for H = 8.

Using distribution-free methods (Mood [1950]), the probability that the p^{th} percentile point of a cumulative distribution lies between the r^{th} and s^{th} order statistic, s > r, is

$$\Pr(x_r < \xi_p < x_s) = \sum_{i=r}^{s-1} \binom{n}{i} p^i (1-p)^{n-i} \qquad (7.8)$$

for a total sample size n. Applying this technique yields the 95 percent confidence intervals shown in Figure 7.18.

To estimate the sensitivity of overall error probability to P_e and P_φ, the error probability for the Fixed Erasure decoder was evaluated for the case defined above, holding one of the probability functions fixed as it is shown in Figure 7.2 while scaling the other by a constant factor. It was found that a 10 percent change in either $P_e(\mu_T)$ or $P_\varphi(\mu_T)$ produced changes of 2.5:1 and 5:1 in P_E at high and low probability levels, respectively.

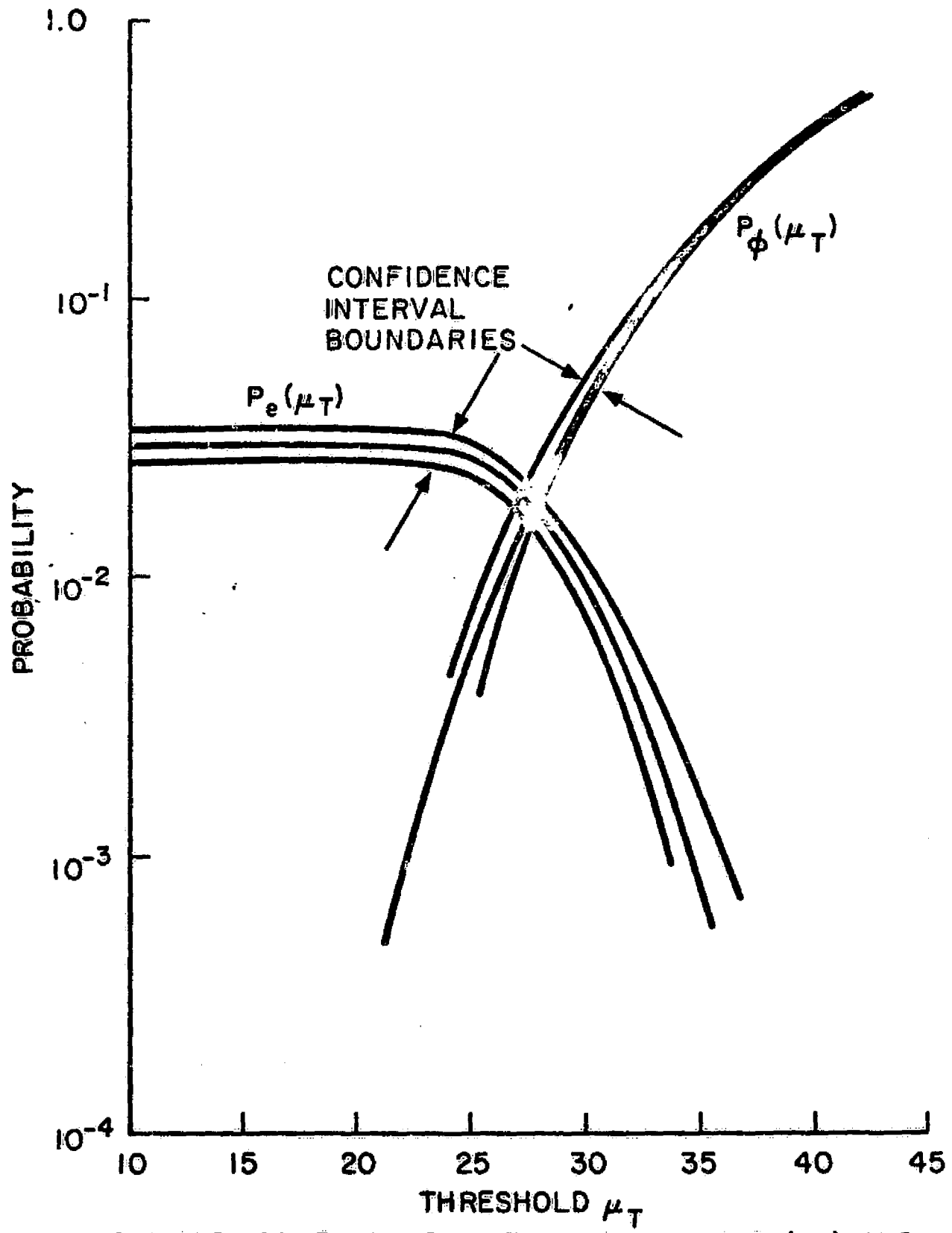


FIGURE 7.18 95% CONFIDENCE INTERVALS ABOUT $P_e(\mu_T)$ AND $P_\phi(\mu_T)$ FOR $(U_B/N_o)_1$=1.0 db, H=8

APPENDIX C*

STRUCTURAL PROPERTIES OF CONVOLUTIONAL CODES

by

W. J. Rosenberg

*extracted from Ph.D. Dissertation of W. J. Rosenberg "Structural Properties of Convolutional Codes", 1971. Chapter numbers correspond to dissertation chapters.

CHAPTER II

ENUMERATION OF CATASTROPHIC ERROR PROPAGATING CONVOLUTIONAL CODES

There are many ways to characterize a convolutional code, and in subsequent sections several will be used. In this section the most convenient is by the use of polynomials as given by Massey and Sain (1968).

(2.1) Definition: A rate 1/n convolutional code has n outputs for each input symbol and is characterized by a 1xn matrix of polynomials in the indeterminate D.

Using this definition of a code, the input sequences are represented by power series in the indeterminate D with coefficients from a finite field GF(q). The output sequences are 1xn matrices whose elements are also power series over GF(q) and are the products of the input sequences with the 1xn matrix of polynomials.

For convolutional codes so defined, Massey and Sain give the following necessary and sufficient conditions for a code to be free of catastrophic error propagation.

(2.2) Theorem (Massey and Sain): A rate 1/n convolutional code is free of catastrophic error propagation if and only if the greatest common divisor of the n polynomials is equal to D^L for some value of L.

Consider the class of convolutional codes defined as above such that at least one of the polynomials has degree = m and the other

n-1 polynomials have degree $\leq$ m. Also with no loss of generality consider only polynomials which have a non-zero constant term. Then it is easy to see that over the field GF(q) there is a total of:

$$\theta_n(m) = (q-1)^n \, q^{n(m-1)} \, (q^n-1) \text{ for } m > 0 \qquad (2.3)$$

codes in this class. For m=0 there are obviously only $(q-1)^n$ such codes.

Now since we know the total number of codes in the class we may compute the number which are subject to catastrophic error propagation and then observe what fraction of all codes this represents. We will need the following definition:

(2.4) Definition: The Delay of a polynomial p(D), is the largest value of s such that D^s divides p(D). Similarly the Delay of a set of polynomials is the minimum of the delays of each of the polynomials in the set. We will consider only sets of polynomials all of which have the same delay.

With this definition in mind consider the result of multiplying each of the n polynomials of a code by an arbitrary polynomial, p(D), of degree k. If the delay of p(D) is zero the result will be another code of degree m+k, while if the delay of p(D) is greater than zero the result will not be a code in the above class but will be a set of polynomials in some larger class. It is clear, however, that the number of sets of degree m and delay s is equal to the number of codes of degree (m-s) and delay necessarily zero. But this is

simply $\theta_n(m-s)$. Therefore, the number of sets of degree m and arbitrary delay is

$$T_n(m) = \sum_{s=0}^{m} \theta_n(m-s) \tag{2.5}$$

and since

$$\theta_n(m) = \begin{cases} (q-1)^n \, q^{n(m-1)} \, (q^n-1) & m > 0 \\ (q-1)^n & m = 0 \end{cases} \tag{2.6}$$

it may be seen that

$$T_n(m) = (q-1)^n \, q^{nm} \tag{2.7}$$

for all m.

Now let $\Psi_n(m)$ be the number of codes of degree m (and delay zero) such that the greatest common divisor of their generator polynomials is 1 (that is, the number of codes which are free from catastrophic error propagation). Then each set of polynomials of degree m and arbitrary delay may be written in a unique way as the product of a code of degree (m-k), and greatest common divisor of 1, and a monic polynomial of degree k for some value of k. Consequently we have

$$T_n(m) = \sum_{k=0}^{m} \Psi_n(m-k) \, Q(k) \tag{2.8}$$

where Q(k) is the number of monic polynomials of degree k.

Now define the three generating functions

$$T_n(x) = \sum_0^\infty T_n(m)\, x^m$$

$$\Psi_n(x) = \sum_0^\infty \Psi_n(m)\, x^m \qquad (2.9)$$

$$\text{and } Q(x) = \sum_0^\infty Q(m)\, x^m$$

Then it follows that $T_n(x) = \Psi_n(x)\, Q(x)$. Since, however, $Q(m) = q^m$

$$Q(x) = \sum_0^\infty q^m x^m = \frac{1}{1-qx} \qquad (2.10)$$

and it follows that $\Psi_n(x) = T_n(x)\,[1-qx]$. (2.11)

Equating coefficients of x^m on both sides yields the relations

$$\Psi_n(m) = T_n(m) - q\, T_n(m-1) \quad m > 0$$

$$\Psi_n(o) = T_n(o) = (q-1)^n \qquad (2.12)$$

Consequently $\Psi_n(m) = (q-1)^n\, q^{nm}\,[1-q^{1-n}] \quad m > 0$ (2.13)

Now let $\Delta_n(m) = 1 - \frac{\Psi_n(m)}{\theta_n(m)}$ (2.14)

Then $\Delta_n(m)$ is the fraction of the class of codes of degree m which are subject to catastrophic error propagation.

$$\Delta_n(m) = \Delta_n = \frac{q-1}{q^n-1} \qquad m > 0 \qquad (2.15)$$

In particular for binary codes (q=2) this reduces to

$$\Delta_n = \frac{1}{2^n - 1} \qquad (2.16)$$

and this is the fraction of all binary, zero delay, convolutional codes of rate 1/n which are subject to catastrophic error propagation.

Equivalence Classes of Codes

Even though the above formulas enumerate the non-catastrophic convolutional codes it would be useful to refine the analysis slightly for some applications. In particular, in any computer search for good codes it is imperative that the search be restricted to representatives of classes which are equivalent in some sense. In this section we will examine one such equivalence relation and will enumerate the distinct equivalence classes which are generated.

The equivalence relation to be considered is that formed by permutations of the generator polynomials. Two codes will be said to be equivalent if their sets of generator polynomials are identical including repetitions if there are any. More formally this becomes:

(2.17) Definition: Two codes, C_1, defined by the ordered n-tuple $(g_1'(D), g_2'(D), \text{---}, g_n'(D))$ are equivalent if there is some permutation of the indices

$$\pi: \quad \begin{array}{l} 1 \to i_1 \\ 2 \to i_2 \\ \quad \vdots \\ n \to i_n \end{array}$$

such that

$$g_{i_1}(D) = g_1'(D)$$
$$g_{i_2}(D) = g_2'(D)$$
$$\vdots$$
$$g_{i_n}(D) = g_n'(D)$$

This is a very natural definition of equivalence, since if the generator polynomials are the same for two codes then the codewords of one will be simply permutations of the codewords of the other and all the weight and error correcting properties of one will apply to the other as well. Now what we want to know is the number of equivalence classes of codes induced by this relation.

To find this number we will make use of Burnside's Lemma (see any text on combinatorial mathematics, e.g. C. L. Liu (1968)) which we will write as follows:

$$E = \frac{1}{|G|} \sum_{\pi \in G} \Psi(\pi) \qquad (2.18)$$

in which $|G|$ is the cardinality of the finite permuation group G, E is the number of equivalence classes of codes induced by G, and $\Psi(\pi)$ is the number of codes which are invariant under the permutation π, a member of G. In our application of this Lemma G is the full symmetric group on n objects, S_n, and its cardinality, $|S_n|$, equals n!. A code is seen to be invariant under a permutation π if each cycle of π merely rearranges identical generator polynomials of the code. Burnside's Lemma has now changed the problem to one of finding $\Psi(\pi)$, the number of codes which are invariant for each permutation π

in S_n. We will need the following definitions.

(2.19) Definition: Let $\mathscr{C}_n(m)$ denote the set of all rate 1/n codes of degree m which are free of catastrophic error propagation.

(2.20) Definition: For each permutation $\pi \in S_n$ which has ν cycles, we say that $C' \in \mathscr{C}_\nu(m)$ is a base for $C \in \mathscr{C}_n(m)$, with respect to π, if and only if C is invariant under π and if $g_i'(D)$, the i^{th} generator of C' is a representative of the i^{th} cycle of C under some ordering of cycles.

In order for C' to be a base for C with respect to π we see that several conditions must be satisfied. First, the number of generator polynomials in C' must equal ν, the number of cycles in the permutation π. Also each distinct generator polynomial in C must occur at least once in C'. Finally the ordering of the generator polynomials of C' is fixed by the ordering of the cycles of π. This ordering is arbitrary, however for each occurance of the specific permutation, π, it must be the same.

We note that for a given permutation $\pi \in S_n$ which has ν cycles each code $C' \in \mathscr{C}_\nu(m)$ is the base for exactly one code $C \in \mathscr{C}_n(m)$ which is invariant under π, and conversely each code $C \in \mathscr{C}_n(m)$ which is invariant under π will have a base $C' \in \mathscr{C}_\nu(m)$. Therefore, there is a one to one correspondence between elements of $\mathscr{C}_\nu(m)$ and elements of $\mathscr{C}_n(m)$ which are invariant under a permutation $\pi \in S_n$ which has ν cycles.

Now if we denote by $E_n(m)$ the number of equivalence classes of $\mathscr{C}_n(m)$ we see that Burnside's Lemma becomes

$$E_n(m) = \frac{1}{n!} \sum_{k=1}^{n} c(n,k) \; |\mathscr{C}_k(m)| \tag{2.21}$$

where $c(n,k)$ is the number of permutations $\pi \in S_n$ which have k cycles and $|\mathscr{C}_k(m)|$ is the cardinality of $\mathscr{C}_k(m)$. From combinatorial theory it is known that the numbers $c(n,k)$ are the signless stirling numbers of the first kind and are given by the generating function:

$$c_n(t) = \sum_{k=1}^{n} c(n,k) \; t^k = t(t+1)\;(t+2)\text{---}(t+n-1) \tag{2.22}$$

(see for example Riordan (1958)).

From above we have the relation

$$|\mathscr{C}_k(m)| = (q-1)^k \; q^{mk} \; (1-q^{1-k}) \tag{2.23}$$

and of course $|S_n| = n!$

so we have that

$$E_n(m) = \frac{1}{n!} \sum_{k=1}^{n} c(n,k) \; (q-1)^k \; q^{mk} \; (1-q^{1-k}) \tag{2.24}$$

or, rewriting the above,

$$E_n(m) = \frac{1}{n!} \sum_{k=1}^{n} c(n,k) \; [q^m(q-1)]^k - \frac{q}{n!} \sum_{k=1}^{n} c(n,k)[q^{m-1}(q-1)]^k \tag{2.25}$$

to which we may apply the expansion of the generator function of $c(n,k)$ to obtain

$$E_n(m) = \frac{1}{n!}\frac{[(q-1)q^m+n-1]!}{[(q-1)q^m - 1]!} - \frac{1}{n!}\,q\,\frac{[(q-1)q^{m-1} + n-1]!}{[(q-1)q^{m-1} - 1\ ]!} \qquad (2.26)$$

which may be rewritten in terms of binomial coefficients as:

$$E_n(m) = \binom{q^m(q-1) + n-1}{n} - q\binom{(q-1)\,q^{m-1} + n-1}{n} \qquad (2.27)$$

In particular, for the case of binary codes, q=2 the relation reduces to:

$$E_n(m) = \binom{2^m + n-1}{n} - 2\binom{2^{m-1} + n-1}{n} \qquad (2.28)$$

This gives an upper bound on the number of codes which would need to be considered in any computer search of convolutional codes with good distance properties.

CHAPTER III

FINITE STATE MACHINE MODEL OF CONVOLUTIONAL CODES

It is quite reasonable to discuss convolutional codes in terms of finite-state machines, since the standard shift register encoders for these codes are themselves very straightforward examples of finite-state machines. Recently Viterbi (1969) and Forney (1970) have separately discussed finite state machine representations of convolutional codes and have obtained results about weight enumerators and canonic forms respectively. The approach here will differ somewhat from those authors and will concern itself primarily with the state transition matrix of a finite state machine. A relationship between the finite state machine representation and the generator polynomial representation of the previous chapter will also be presented.

First we will consider the obvious shift register realization of a convolutional encoder. In this realization the encoder consists of a shift register of m+1 stages and n GF(q) adders which are connected to the various stages of the shift register in order to form n outputs which are linear combinations of the contents of the shift register. In this way the first m stages may be considered to define the state of the machine and consequently there are q^m states. Then since there can be q inputs to the machine each state will have q paths leading from it on a state transition diagram, or if we think

in terms of a state transition matrix, which will be $q^m \times q^m$, each row will have q non-zero entries. We choose to represent these entries in a particular way:

$a_{ij} = D^k$ if the transition from state i to state j resulted in k non-zero outputs (a weight k output)

$a_{ij} = 0$ if the transition is not possible

It will be seen that the all zero state of the shift register only occurs as the initial and final states in any sequence of state transitions of interest. We will not have occasion to consider code-words which involve an intermediate return to the all zero state. Consequently we will, for the most part, consider the reduced finite state machine and its transition matrix which is formed by deleting the zero state. This yields a $(q^m-1) \times (q^m-1)$ matrix.

At this point it will be useful to introduce the concept of an extension of a finite state machine:

(3.1) Definition: The k-th Extension of a linear finite state machine is that machine whose inputs are sequences of k symbols and whose outputs are the sequences of outputs which would have been produced by the original machine in response to the same sequence of inputs.

From this definition it is clear that the number of states does not change in going from the original machine to its k-th extension;

however, the number of possible state transitions does increase. The number of non-zero entries per row in the transition matrix increases from q to q^k. The new state transition matrix for the k-th extension is found quite simply by taking the k-th power of the state transition matrix of the original machine. We note that there is a difference depending on whether the zero state row and column are deleted before or after the matrix is raised to the k-th power. We will ignore this difference for the moment since it will not matter. In general we will do the deletion first.

Return for the moment to the original state transition matrix of a machine corresponding to a code with catastrophic error propagation. There will be at least one element which is unity corresponding to a non-zero input and zero output. Then there may be some elements which are unity but which correspond to both a zero input and a zero output. All the other elements will be either zero or D to some positive power. The following definition gives a natural way of reducing the complexity of such a matrix by eliminating the non essential elements.

(3.2) Definition: The Evaluated transition matrix is the result of letting D equal zero in the matrix. The evaluated matrix consists only of zeros and ones.

The evaluated matrix is seen to represent that portion of the machine which is responsible for the catastrophic behavior of the code. If the code is subject to catastrophic error propagation then this evaluated transition matrix will indicate a cycle of k transitions for some value of k. In the k-th extension of the evaluated transition

matrix the catastrophic cycle will appear as an element on the main diagonal which will, obviously, be unity. Conversely any non-zero element on the main diagonal of the k-th extension will correspond to a cycle in the original machine. Also, since the largest cycle possible is one involving all q^m-1 non-zero states we have the following theorem:

(3.3) Theorem: A convolutional code is subject to catastrophic error propagation if and only if the k-th extension of the evaluated transition matrix contains a 1 on its main diagonal for some value of $k \leq q^m-1$.

At this point it will be instructive to consider an example before discussing the relationship between this finite state approach and the generator polynomial approach of the previous chapter. Consider the convolutional code defined by the following shift register encoder.

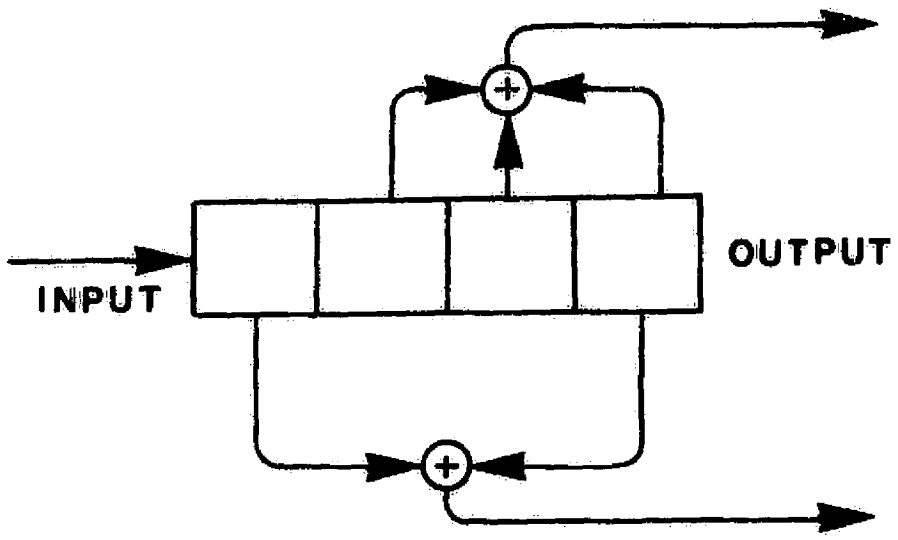


Figure 3.1 Shift Register Encoder

This is a rate 1/2 binary encoder for which m=3. Therefore there are 2^3=8 states and the finite state diagram is:

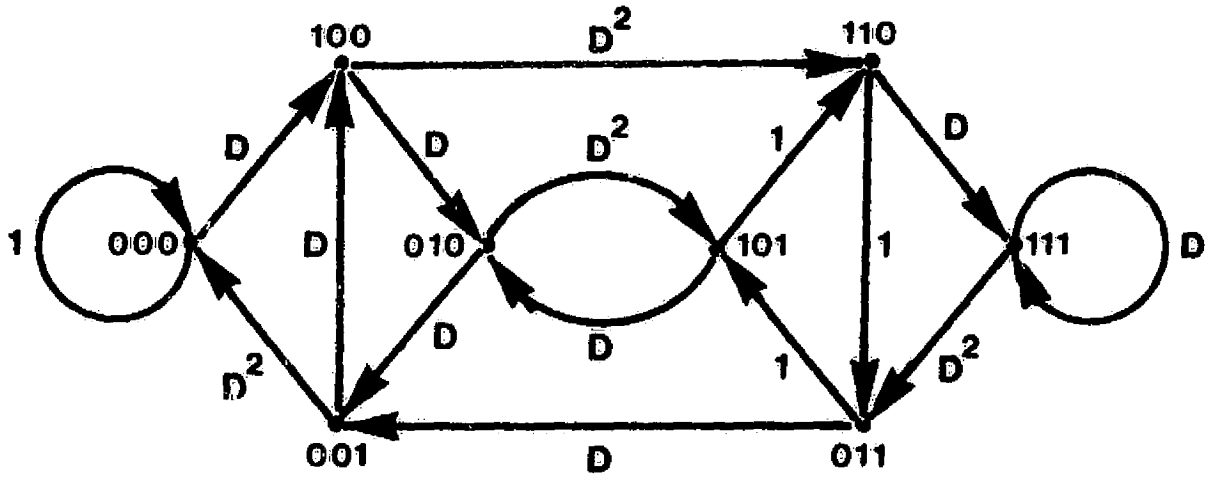


Figure 3.2 State Transition Diagram

Consequently the state transition matrix is:

	000	100	010	110	001	101	011	111
000	1	D	0	0	0	0	0	0
100	0	0	D	D^2	0	0	0	0
010	0	0	0	0	D	D^2	0	0
110	0	0	0	0	0	0	1	D
001	D^2	D	0	0	0	0	0	0
101	0	0	D	1	0	0	0	0
011	0	0	0	0	D	1	0	0
111	0	0	0	0	0	0	D^2	D

This is a good place to mention that the finite state diagrams of these codes will always be, as above, one of the familiar de Bruijn graphs (Golomb (1967)). The only difference from one code to the next is in the labelling of the transitions; that is, the weight assignments for the transitions.

The evaluated form of the matrix, after the zero state deletion, is:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

and the second and third extensions follow immediately:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

2^{nd} extension

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

3^{rd} extension

From the above it is easily seen that there is a cycle of length 3. That the cycle is of length 3 became apparent in the 3rd extension, and it is readily seen from the three diagonal entries that the states involved are 101, 011, 110. From the original state diagram or from the original transition matrix it is seen that the sequence 101

repeated indefinitely will result in catastrophic error propagation.

Now, with the above example as a reference, we may say more about the cycles. Using the D-transform notation for sequences, the sequence 101 becomes D^2+1 and the endless repetition of it is

$$i(D) = \frac{D^2+1}{1-D^3} \tag{3.4}$$

In general the endless repetition of a sequence of length k represented by the polynomial f(D), of degree $\leq$ k-1 is given by

$$i(D) = \frac{f(D)}{1-D^k} \tag{3.5}$$

and its effect on any one of the generator polynomials of the encoder, $g_j(D)$, is given by

$$y_j(D) = g_j(D)\ i(D) = \frac{g_j(D)\ f(D)}{1-D^k} \tag{3.6}$$

For catastrophic error propagation to occur, $y_j(D)$ must be of finite length (that is, a polynomial) for every choice of j. In the above example $g_1(D) = D^2+D+1$ and $g_2(D) = D^3+1$. Consequently $y_1(D) = 1+D$ and $y_2(D) = 1+D^2$ when $f(D) = 1+D^2$. It is clear that $1-D^k$ must divide the product $g_j(D)\ f(D)$ and since $1-D^k$ cannot divide f(D) there must be some common factor of the $g_j(D)$ which divides $1-D^k$ with the quotient f(D). Assume that this value of k is the smallest for which the above is possible and let c(D) be the corresponding common factor of the $g_j(D)$. Since k is the least such value, c(D) must be irreducible, say of degree d. Then following Berlekamp (1968), k is

the period of c(D) and k must divide q^d-1. This gives us the following theorem relating the two representations:

(3.7) Theorem: Each catastrophic error producing cycle in the finite state machine representation has as its length the period of a common divisor of the generating polynomials in the polynomial representation of the encoder.

Also we know a little bit more about which extensions might help in making the decisions, for a particular code, as to its susceptibility to catastrophic error propagation. Clearly if a sequence of length k causes catastrophic error propagation when repeated, any multiple of k will also be acceptable (only the minimum values correspond physically to the lengths of the cycles). Consequently, since the value of k corresponding to an irreducible common divisor c(D) of degree d divides q^d-1 we need look only at the (q^d-1)-th extensions for all $d \leq m$. We may state this as:

(3.8) Theorem: A code is free of catastrophic error propagation if and only if the (q^d-1)-th extensions of its evaluated transition matrix are zero on the main diagonal for all $d \leq m$.

We also observe that it is relatively easy to compute (q^d-1)-th extensions and especially for the binary case when q=2. This is true since going from the d-th case to the (d+1)-th case involves taking the (q^d-1)-th extension to the q^{th} power and then multiplying by the (q-1)-th extension. In the binary case this amounts to

squaring and multiplying by the original transition matrix.

Finite State Representation and Non-Catastrophic Codes

In the more general case of non-catastrophic codes a problem which arises out of decoding considerations is that of determining the lowest weight codeword. There are two parts to this problem. The first concerns itself with finding the minimum distance codeword in the absolute sense of having the smallest weight. The second part is concerned with finding the arbitrarily long codeword which has the minimum weight per encoded symbol. We will treat this second part of the problem in this section.

We will again consider the state transition matrix of a code as defined in the previous section, but here we will need some more detailed notation, as follows:

Let A be the transition matrix with elements a_{ij} defined as before. The k-th extension will be denoted as A^k and its elements will be $a_{ij}^{(k)}$. These elements $a_{ij}^{(k)}$ will be polynomials in the indeterminate D. The delay (the degree of the lowest degree term) of the polynomial $a_{ij}^{(k)}$ will be denoted by $m_{ij}^{(k)}$.

The elements $a_{ij}^{(k)}$ of the matrix A^k are seen to be the enumerators by weight of the k step transitions from state i to state j. The degree of each term of $a_{ij}^{(k)}$ is the weight produced by one of the k step paths from state i to state j. The coefficient of each term is the number of such paths of that weight.

With this notation we see that the weight of the minimum

weight, k step path from state i to state j is given by $m_{ij}^{(k)}$. Consequently, the weight of the minimum weight, non-zero, codeword for k input symbols is given by $m_{00}^{(k)}$ when it exists, where the initial state is denoted by 0.

Now we notice two properties of the matrix A:

1) The matrix A is irreducible. That is, there is no renumbering of the states which would permit the representation:

$$A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ 0 & A_{2,2} \end{bmatrix}$$

where $A_{1,1}$ and $A_{2,2}$ are square submatrices. This is necessarily so since every state is eventually reachable from every other.

2) If we consider D to be a real number greater than zero, then the matrix A is non negative ($a_{ij} \geq 0$ for all i,j)

From this, and a theorem of Frobenius (Varga (1962)), we know that the matrix, A, has an eigenvalue, r, with the following properties:

1) r is real and $r > 0$
2) if r' is any other eigenvalue of A then $|r'| \leq r$
3) r is a simple eigenvalue of A
4) the eigenvector, z, corresponding to r is positive (all its elements are positive)
5) r is an increasing function of each a_{ij}

Also we may say a little more about A. Since each state is reachable

from each other state in exactly m steps (and possibly less), we know that $A^m > 0$, all elements, $a_{ij}^{(m)}$, of A^m are positive. And since $A \geq 0$ it is clear that $A^k > 0$ for all $k \geq m$. This means that there is always a k step path from any state to any other provided that $k \geq m$.

Now let us consider the maximal eigenvalue, r. Since r is an eigenvalue of A with the corresponding eigenvector, z, we have:

$$Az = rz \tag{3.9}$$

and consequently $A^k z = r^k z$ for all powers k and if A is an nxn matrix and the elements of z are $z_1, z_2, \cdots z_n$ we have:

$$\sum_{j=1}^{n} z_j \, a_{i,j}^{(k)} = z_i r^k \quad \text{for } i=1,2,\cdots,n \tag{3.10}$$

Now consider, for two distinct states, s and t, the difference in weight of the minimum weight, k-step, paths from a given state i to state s and from state i to state t. This difference is given by $|m_{i,s}^{(k)} - m_{i,t}^{(k)}|$. We will show that this difference may be bounded independent of k.

Assume for the moment that $m_{i,s}^{(k)} \geq m_{i,t}^{(k)}$ for some very large value of k. Then there will be a path from state i to state s through state t. In fact we will find a particular path of that form. Consider first the segment from state t to state s. This will be of some length α, to be specified below. As long as $\alpha \geq m$ a path of that length will exist. Then we must find a path from state i to state t of length $k-\alpha$. We wish to choose this path in a particular

way. Let k be large enough such that every path of length k from state i to state t contains a loop; that is, at least one state which is entered more than once. Let α be the length of this loop in the minimum weight path of length k. Then we have the following two inequalities:

$$m_{i,s}^{(k)} \geq m_{i,t}^{(k-\alpha)} + m_{t,s}^{(\alpha)} \tag{3.11}$$

$$m_{i,t}^{(k-\alpha)} \leq m_{i,t}^{(k)} \tag{3.12}$$

From this it follows that

$$m_{i,s}^{(k)} - m_{i,t}^{(k)} \leq m_{t,s}^{(\alpha)} \tag{3.13}$$

In the case in which $m_{i,t}^{(k)} > m_{i,s}^{(k)}$ it follows similarly that:

$$m_{i,t}^{(k)} - m_{i,s}^{(k)} \leq m_{s,t}^{(\alpha)} \text{ for some } \alpha \leq q^m \tag{3.14}$$

Consequently, it must be true that

$$|m_{i,s}^{(k)} - m_{i,t}^{(k)}| \leq \max_{\alpha \leq q^m}\{m_{t,s}^{(\alpha)}\} + \max_{\alpha \leq q^m}\{m_{s,t}^{(\alpha)}\} \tag{3.15}$$

Therefore, we have that $|m_{i,s}^{(k)} - m_{i,t}^{(k)}|$ is bounded above by a bound which is independent of k.

However, for all but catastrophic codes we know that $m_{i,j}^{(k)}$ must be an increasing function of k. And since the eigenvector, z, is constant with k, we have that the delay of z_i is independent of k. Therefore we have

$$\lim_{k\to\infty} \frac{\text{delay } z_i a_{ij}^{(k)}}{\text{delay } z_i a_{ii}^{(k)}} = \lim_{k\to\infty} \frac{\text{delay } z_i + \text{delay } a_{ij}^{(k)}}{\text{delay } z_i + \text{delay } a_{ii}^{(k)}} \tag{3.16}$$

$$= \lim_{k\to\infty} \frac{m_{ij}^{(k)}}{m_{ii}^{(k)}} = 1 \text{ for all } i,j \tag{3.17}$$

Since delay $\left(\sum_{j=1}^{n} z_j a_{ij}^{(k)}\right) = \min_{1 \le j \le n} (\text{delay } z_j + m_{ij}^{(k)})$ and (3.17) is true for the value of j which minimizes (delay $z_j + m_{ij}^{(k)}$) we have

$$\lim_{k\to\infty} \frac{\text{delay} \sum_{j=1}^{n} z_j a_{ij}^{(k)}}{\text{delay } z_i a_{ii}^{(k)}} = 1 \tag{3.18}$$

But this implies that

$$\lim_{k\to\infty} \frac{\text{delay } z_i r^k}{\text{delay } z_i a_{ii}^{(k)}} = 1 \quad \text{for all } i \tag{3.19}$$

and then

$$\lim_{k\to\infty} \frac{\text{delay } r}{\frac{1}{k} \text{delay } a_{ii}^{(k)}} = 1 \quad \text{for all } i \tag{3.20}$$

However, the quantity delay r needs some interpretation, since, in general, r will not be a polynomial in D but will involve D to some fractional powers. In these cases delay r will be interpreted in the natural way to be the smallest rational exponent of D appearing in r. This will be the rational power of D which r approaches asymptotically as $D \to 0$. Since r is an increasing function of each matrix element

a_{ij}, which in turn are increasing functions of D we know that this smallest rational exponent will be positive.

From all this we have that

$$\lim_{k \to \infty} \frac{1}{k} \text{ delay } a_{ii}^{(k)} = \text{delay } r \quad \text{for all } i \tag{3.21}$$

and we will now consider for the special case of i=0, the distinguished initial state, since $a_{00}^{(k)}$ is the enumerator by weight of the codewords on k input symbols. Consequently $\frac{1}{k}$ delay $a_{00}^{(k)}$ is the average weight per input symbol of those codewords with minimum weight. Therefore, we have proven the following theorem:

(3.22) Theorem: Given a code C and its corresponding state transition matrix A, the average weight per input symbol of the minimum weight codewords tends toward some value ω as the codeword length becomes large. Moreover, this value ω is the rational exponent of the indeterminate D towards which the maximal eigenvalue, r, of A tends asymptotically as $D \to 0$.

CHAPTER IV

CONSECUTIVE ZEROS IN CONVOLUTIONAL CODEWORDS

We know that in a code which is subject to catastrophic error propagation there exist codewords which have arbitrarily long sequences of consecutive zeros. In fact it is this very property which causes the code to be catastrophic. In this section we will present some bounds on the number of consecutive zeros in a non catastrophic code. Alternatively, of course, these may be viewed as necessary and sufficient conditions under which a code is catastrophic. We realize that a rate 1/n encoder generates n symbols for each input symbol; however we are only going to be concerned in this section with branches all of whose n symbols are zero. Consequently, all of the following results will be on the number of consecutive blocks of n zeros which may occur.

Recently Odenwalder (1970) has given the following upper bound on the number of consecutive zero weight branches which may occur in a non catastrophic convolutional code:

(4.1) Theorem (Odenwalder): If the encoder for a rate 1/n non-catastrophic convolutional code is an m+1 stage linear shift register then the maximum number of consecutive zero weight branches which may occur is m-1.

This theorem has also been developed independently by Bahl and Jelinek (1970) for rate 1/2 convolutional codes.

In this section we will give a refinement of this theorem. We

will give a lower bound to the number of consecutive zero weight branches and finally we will give a constructive technique which achieves the lower bound.

First we must consider the form of the infinite generator matrix representation of a convolutional code. We will assume that the code has rate 1/n and constraint length m+1. Then we may represent the n linear shift register tap sequences as follows:

$$\begin{aligned} g_1 &= (g_{1,0}g_{1,1}g_{1,2}\cdots,g_{1,m}) \leftrightarrow g_1(D) = g_{1,0}+g_{1,1}D + \cdots + g_{1,m}D^m \\ g_2 &= (g_{2,0}g_{2,1}g_{2,2}\cdots,g_{2,m}) \leftrightarrow g_2(D) = g_{2,0}+g_{2,1}D + \cdots + g_{2,m}D^m \\ &\vdots \\ g_n &= (g_{n,0}g_{n,1}g_{n,2}\cdots,g_{n,m}) \leftrightarrow g_n(D) = g_{n,0}+g_{n,1}D + \cdots + g_{n,m}D^m \end{aligned} \tag{4.2}$$

From this we know that we may represent the encoding process in a matrix form similar to the generator matrix for a block code as follows:

$$\underline{y} = (y_1, y_2, \cdots) = (x_1, x_2, \cdots)\, G \tag{4.3}$$

where G is the arbitrarily large matrix:

$$G = \begin{bmatrix} (g_{1,0}g_{2,0}\cdots g_{n,0}) & (g_{1,1}g_{2,1}\cdots g_{n,1}) & \cdots & (g_{1,m}\ g_{2,m}\cdots g_{n,m}) & & \\ & (g_{1,0}g_{2,0}\cdots g_{n,0}) & \cdots & (g_{1,m-1}g_{2,m-1}\cdots g_{n,m-1}) & (g_{1,m}\ g_{2,m}\cdots g_{n,m}) & \\ & & & \vdots & (g_{1,m-1}g_{2,m-1}g_{n,m-1}) & \\ & & & (g_{1,0}\ g_{2,0}\cdots g_{n,0}) & (g_{1,1}\ g_{2,1}\cdots g_{n,1}) & \\ & & & & (g_{1,0}\ g_{2,0}\cdots g_{n,0}) & \\ & & & & & \ddots \end{bmatrix} \tag{4.4}$$

and each y_i is actually the n-tuple corresponding to one branch of the code.

Now, if we are considering, as in the statement of Odenwalder's theorem, an arbitrary segment of m consecutive branches, we must consider the general form of m consecutive n-tuples of columns of the matrix G. Such a section of G will appear as G', a 2m x nm matrix:

$$G' = \begin{bmatrix} g_{1,m} & g_{2,m} & \cdots & g_{n,m} & & & & & & & & \\ g_{1,m-1} & g_{2,m-1} & \cdots & g_{n,m-1} & g_{1,m} & g_{2,m} & \cdots & g_{n,m} & & & & \\ \vdots & & & & & & & & \ddots & & & \\ & & & & g_{1,m-1} & g_{2,m-1} & \cdots & g_{n,m-1} & & \ddots & & \\ & & & & \vdots & & & & & & g_{1,m} & g_{2,m} & \cdots & g_{n,m} \\ g_{1,0} & g_{2,0} & \cdots & g_{n,0} & g_{1,1} & g_{2,1} & \cdots & g_{n,1} & & & g_{1,m-1} & g_{2,m-1} & \cdots & g_{n,m-1} \\ & & & & g_{1,0} & g_{2,0} & \cdots & g_{n,0} & & & \vdots & & & \\ & & & & & & & & \ddots & & & & & \\ & & & & & & & & & & g_{1,0} & g_{2,0} & \cdots & g_{n,0} \end{bmatrix} \quad (4.5)$$

Considering the segment G' of the matrix G as representing the input-output relation of the encoder for an arbitrary sequence of m consecutive branches we may restate Odenwalder's theorem as follows:

(4.6) Theorem: For a non-catastrophic convolutional code, the 2m rows in a section of the generator matrix such as G' are linearly independent.

This theorem is easily seem to be equivalent to the previous statement, Theorem (4.1), since the 2m rows and nm columns as illustrated correspond to m consecutive branches, and linear independence of the rows is obviously equivalent to not being able to produce an all zero output over those n branches.

There is a natural rearrangement of the columns of G' which

exploits the fact that each column is merely the tap sequence vector for one of the n generators. This rearrangement is as follows:

$$\begin{bmatrix}
g_{1,m} & & & & g_{n,m} & & & \\
g_{1,m-1} & g_{1,m} & & & g_{n,m-1} & g_{n,m} & & \\
g_{1,m-2} & g_{1,m-1} & & & \vdots & g_{n,m-1} & & \\
\vdots & \vdots & & & \vdots & \vdots & & \\
g_{1,1} & g_{1,2} & \cdots g_{1,m} & \cdots & g_{n,1} & g_{n,2} & \cdots g_{n,m} & \\
g_{1,0} & g_{1,1} & \cdots g_{1,m-1} & & g_{n,0} & g_{n,1} & \cdots g_{n,m-1} & \\
 & g_{1,0} & \vdots & & & g_{n,0} & \vdots & \\
 & & g_{1,0} & & & & g_{n,0} &
\end{bmatrix} \qquad (4.7)$$

The condition of linear independence of the 2m rows of this matrix is equivalent to the condition that the column rank of this matrix must be 2m, which in turn is equivalent to the condition that all column vectors of dimension 2m may be formed as linear combinations of the above columns. In fact, from the above nm columns we know that we can find a basis of cardinality 2m which spans the space of column vectors of dimension 2m.

As written in the above partitioned form, however, it is apparent that these conditions are equivalent to a general statement about polynomials. Since, within any one partition, the columns are merely shifts of their predecessors, we may consider them as polynomials as follows:

$$\left[\begin{array}{ccc|c|ccc}
D^{m-1}g_1(D) & D^{m-2}g_1(D) & \cdots g_1(D) & \cdots & D^{m-1}g_n(D) & D^{m-2}g_n(D) & \cdots g_n(D)
\end{array}\right] \qquad (4.8)$$

Thus it is a simple consequence that the linear independence condition is equivalent to the statement that any polynomial, p(D), of degree $\leq$ 2m-1 has a representation of the form:

$$p(D) = a_1(D)\, g_1(D) + a_2(D)g_2(D) + \text{---} + a_n(D)g_n(D) \qquad (4.9)$$

where the $a_i(D)$ are polynomials of degree $\leq$ m-1.

For the special case in which n=2 (rate 1/2 codes) it is a well known result of algebra (for example, Littlewood (1950)) that all such polynomials, p(D), may be written as:

$$p(D) = a_1(D)g_1(D) + a_2(D)g_2(D)$$
$$\text{with degree } \{a_i(D)\} \leq m-1 \qquad (4.10)$$

if and only if the greatest common divisor of $g_1(D)$ and $g_2(D)$ is unity. This, of course, is simply Massey and Sain's condition for a code to be free of catastrophic error propagation. In the appendix we prove that this is true for all $n \geq 2$; that if and only if the greatest common divisor of the generator polynomials is unity then all polynomials, p(D), of degree $\leq$ 2m-1, may be written, in the form of equation (7), as linear combinations of the generator polynomials. In fact, it will be shown that, in general, tighter bounds on the degrees of the polynomial coefficients, $a_i(D)$, than merely degree $\{a_i(D)\} \leq$ m-1 may be imposed. These bounds will be shown to have further implications on the number of consecutive zeros for certain codes.

The specific form of the theorem which we will prove in the

appendix is:

(4.11) Theorem: Let $\{g_1(D), g_2(D),---,g_n(D)\}$ be a set of n monic polynomials such that degree $\{g_i(D)\} = m$ for $i=1,2,---,n$. Let $\delta_i(D)$ be the greatest common divisor of $\{g_1(D), g_2(D),---,g_i(D)\}$ for $i=1,2,---,n$ and let d_i = degree $\{\delta_i(D)\}$. Assume that d_n=g.c.d.$\{g_1(D),g_2(D),---,g_n(D)\} = 1$. Then any polynomial, p(D), of degree $\leq$ 2m-1 may be written in the form:

$$p(D)=a_1(D)g_1(D)+a_2(D)g_2(D)+ \text{---} +a_n(D)g_n(D)$$

such that degree $\{a_i(D)\} < d_{i-1}-d_i$ for $i=2,3,---n$ and degree $\{a_1(D)\} < m$

From this theorem we see that if the greatest common divisor of the generator polynomials, $\{g_i(D)\}$, is unity then the set of polynomials:

$$S = \{D^k g_i(D) \mid 0 \leq k \leq m-1 \ , \ 1 \leq i \leq n\} \qquad (4.12)$$

spans the space of polynomials of degree $\leq$ 2m-1. Consequently, we conclude that the 2m rows of G' in equation (4.5) are linearly independent and we have proven Odenwalder's theorem, Theorem 4.1.

This theorem gives an upper bound to the number of consecutive zero weight branches which may occur in a non-catastrophic code. We may, at this point derive a corresponding lower bound. The lower bound will tell us that a non-catastrophic code must have at least one codeword with some prescribed number of consecutive zero weight branches.

(4.13) Theorem: A rate $\frac{1}{n}$ convolutional code of constraint length m+1 must have a codeword with a sequence of $\{\frac{m}{n-1} - 1\}$ consecutive zero weight branches, where $\{x\}$ denotes the least integer greater than or equal to x.

This theorem is easily seen from a consideration of the portion of the generator matrix corresponding to k consecutive branches, similar to the matrix G' in equation (4.5). It will have m+k rows and nk columns. Consequently, for any $k < \frac{m}{n-1}$ there must be more rows than columns and, therefore, the rows are linearly dependent. The theorem follows since $\{\frac{m}{n-1} - 1\}$ is the largest such integer $< \frac{m}{n-1}$.

For n=2, that is a rate $\frac{1}{2}$ code, this is the strongest such bound possible since it claims that there must be sequences of m-1 consecutive zero weight branches. We know, however, that for a non-catastrophic code we can have at most m-1 consecutive zeros so these bounds are tight for n=2. For lower rate codes, that is n > 2, we will be able to improve both the upper and lower bounds. First we will develop the improved lower bound.

(4.14) Theorem: A rate $\frac{1}{n}$ convolutional code of constraint length m+1 such that at most ℓ of the generator tap vectors are linearly independent must have a codeword with a sequence of $\{\frac{m}{\ell-1} - 1\}$ consecutive zero weight branches. Again $\{x\}$ denotes the least integer $\geq x$.

Consider again the portion of the generator matrix corresponding to k consecutive branches as in the proof of Theorem (4.13). It will have m+k rows and nk columns. At most ℓk of these columns can be

linearly independent, however. Consequently for any $k < \frac{m}{\ell-1}$ the m+k rows must be linearly dependent and the theorem follows.

We will now develop the corresponding upper bound for a rate $\frac{1}{n}$ code such that only ℓ of its generator tap vectors are linearly independent. From Theorem (4.11) we saw that we could make the claim in equation (4.12) that the set

$$S = \{D^k g_i(D) \mid 0 \leq k \leq m-1 \quad , i=1,n\} \tag{4.15}$$

spans the space of polynomials of degree $\leq 2m-1$. From this set, S, we will choose a basis which will lead to the following theorem:

(4.16) Theorem: Let $\{g_1(D), g_2(D), ---, g_n(D)\}$ be the generators for a rate 1/n code. Assume that $\ell \leq n$ of the $g_i(D)$ form a maximal linearly independent set, and that they are $g_1(D), g_2(D), ---, g_\ell(D)$. Then if the code is non-catastrophic the maximum number of consecutive zero weight branches is $\leq m-\ell+1$.

To prove this theorem we will need a lemma which we will now state and prove:

(4.17) Lemma: Let $\{g_1(D), g_2(D), ---, g_n(D)\}$ be the set of generator polynomials of a non-catastrophic rate 1/n code. Assume that the set of polynomials:

$$S = \{D^{k_i} g_i(D) \mid 0 \leq k_i \leq K_i \ , \ 1 \leq i \leq n\} \tag{4.18}$$

is a basis for the space of polynomials of degree $\leq 2m-1$. Assume further that $K_1 = m-1$. Then if $K = \max_{2 \leq i \leq n} \{K_i\}$, the

maximum number of consecutive zero weight branches is K.

First observe that if for some $K \leq m-1$ the set of polynomials:

$$S_K = \{D^k g_i(D) \mid 0 \leq k \leq K \ , \ 1 \leq i \leq n\} \tag{4.19}$$

spans the space of polynomials of degree $\leq m+K$ then the maximum number of consecutive zero weight branches which may occur is $\leq K$. For consider the segment, G', of the generator matrix, G. If S_K spans the space of polynomials of degree $\leq m+K$ then the portion of the matrix, G', corresponding to K+1 consecutive branches (the first n(K+1) columns and m+K+1 rows) will have full rank equal to m+K+1 since this is the dimension of the space of polynomials of degree $\leq m+K$. Consequently, the m+K+1 rows are linearly independent.

Now, as in equation (4.18), if the set

$$S = \{D^{k_i} g_i(D) \mid 0 \leq k_i \leq K_i \ , \ 1 \leq i \leq n\}$$

is a basis for the space of polynomials of degree $\leq 2m-1$ then the subset of S formed by deleting $\{D^{K+1}g_1(D), D^{K+2}g_1(D), \cdots, D^{m-1}g_1(D)\}$ from S must be a basis for the subspace of polynomials of degree $\leq m+K$. Thus we have proven Lemma (4.17).

To prove Theorem (4.16) from this we will show that it is possible to construct a basis, B, for the space of polynomials of degree $\leq 2m-1$ which has the following properties:

1) $g_i(D) \in B \quad i = 1,2,\cdots, \ell$

2) $D^k g_1(D) \in B \quad k=0,1,2,\cdots, m-1$

3) $D^k g_i(D) \in B$ implies that $D^{k-1} g_i(D) \in B$

for $k > 0$ and all $i = 1,2,---, \ell$.

Then since a basis, B, for this space has cardinality 2m it will, of course, be true that $m-\ell+1 \geq \max_{2 \leq i \leq \ell} \{k_i \mid D^{k_i} g_i \in B\}$. Then by Lemma (4.17) we will have proven Theorem (4.16). All that remains is to show that such a basis exists. We will do this by a construction. First remember that the set S of equation (4.12) spans the space. Then since ℓ is assumed maximal notice that

$$S' = \{D^k g_i(D) \mid 0 \leq k \leq m-1 \ , \ 1 \leq i \leq \ell\} \qquad (4.20)$$

must also span the space. Now to start the construction let B contain:

$$\{D^k g_1(D) \mid 0 \leq k \leq m-1\}$$

and also:

$$\{g_i(D) \mid i = 2,3,---, \ell\}$$

If $m=\ell-1$ we are finished. Otherwise augment B with any $D^k g_i(D)$ such that $D^{k-1} g_i(D) \in B$ for $2 \leq i \leq \ell$, and $D^k g_i(D)$ is not already in the space spanned by B thus far. Since a basis for the whole space can only contain 2m such polynomials this process must terminate after at most $m-\ell+1$ such augmentations. We claim that the process will take exactly that many steps and that B will then be the required basis. Since we know that S' spans the space we need only show that B spans S'. Let $k_i = \max\{k \mid D^k g_i(D) \in B\}$ and let $K = \max_{2 \leq i \leq \ell} \{k_i\}$. By the

construction process we know that since $D^{k_i+1} g_i(D)$ for $2 \leq i \leq \ell$ was not included in B it must be in the span of B. Using this inductive reasoning we may see that B augmented with

$$\{D^k g_1(D) \mid m \leq k \leq m+K-1\}$$

must span the space of polynomials of degree $\leq 2m+K-1$ and that therefore B must be a basis for the space of polynomials of degree $\leq 2m-1$. Theorem (4.16) then follows as indicated above.

Now we will use Theorem (4.11) to develop a constructive proof that the bound of Theorem (4.13) may actually be achieved. That is, we will demonstrate constructively the existence of codes which do not have sequences of consecutive zero weight branches of length $\{\frac{m}{n-1}+1\}+1$.

We know from Theorem (4.11) that if we let

$$d_i = \text{degree}[\text{g.c.d.}\{g_1(D), g_2(D), \text{---}, g_i(D)\}] \quad i=1,2,\text{---},n \tag{4.21}$$

then we may choose a basis for the space of polynomials of degree $\leq 2m-1$ which has the form:

$$B=\{D^{k_i} g_i(D) \mid 0 \leq k_i < d_{i-1}-d_i \text{ for } 2 \leq i \leq n \text{ such that } d_{i-1}=0 \text{ and } 0 \leq k_1 < m\}$$

If we let $i^* = \min\{i \mid d_i=0\}$ then we see that

$$|B| = \sum_{i=2}^{i^*} (d_{i-1}-d_i)+m = m + d_1+d_{i^*} = 2m$$

which is the cardinality of a basis, as required. Consequently, using Lemma (4.17) we have:

(4.22) Theorem: For a rate $\frac{1}{n}$ non-catastrophic code the maximum number of consecutive zero weight branches is $< \max_{2 \leq i \leq i^*} \{d_{i-1}-d_i\}$, where d_i is as defined in equation (4.21).

Now consider a rate $\frac{1}{n}$ code which has only ℓ of the n generator polynomials in a maximal linearly independent set and assume that they are $\{g_1(D), g_2(D), ---, g_\ell(D)\}$. It is clear that if $d_n=0$, which is necessary for the code to be non-catastrophic, then $d_\ell=0$ and only those generator polynomials in the maximal linearly independent set appear in the basis B. Consequently if we are to find codes which have the minimum maximum length sequences of consecutive zero weight branches we must consider those codes all of whose generator tap vectors are linearly independent. Also it is apparent that we must consider those codes for which $\max_{2 \leq i \leq n} \{d_{i-1}-d_i\}$ is a minimum, which means that all the differences $(d_{i-1}-d_i)$ are approximately equal.

We may easily construct these codes. To illustrate the procedure consider a rate 1/3 code with m even. Choose three polynomials $\{p_1(D), p_2(D), p_3(D)\}$, each of degree = $\frac{m}{2}$, which are pairwise relatively prime. From these construct the code generator polynomials as follows:

$$g_1(D) = p_1(D)\, p_2(D)$$
$$g_2(D) = p_1(D)\, p_3(D)$$
$$g_3(D) = p_2(D)\, p_3(D)$$

Then we have:

$$d_1 = \text{degree } g_1(D) = m$$

$$d_2 = \text{degree } p_1(D) = m/2$$

$$d_3 = \text{degree } 1 = 0$$

thus: $d_1 - d_2 = m/2$

$$d_2 - d_3 = m/2$$

We see, therefore, that the maximum number of consecutive zero weight branches is < m/2 for this code. We know, by Theorem (4.13), however, that a rate 1/3 code must have a codeword with a sequence of $\{\frac{m}{2} - 1\}$ consecutive zero weight branches. Therefore, we have demonstrated a technique for achieving this lower bound.

In the general case of rate 1/n codes the procedure is very similar. Again the method is to choose n polynomials $\{p_1(D), p_2(D), ---, p_n(D)\}$ each of degree $\frac{m}{n-1}$ which are pairwise relatively prime. Having chosen these polynomials the generator polynomials are formed by taking the n possible products of the n polynomials taken n-1 at a time. It is assumed here that $\frac{m}{n-1}$ is an integer. Each of the generator polynomials will therefore have degree m and in all cases $(d_{i-1} - d_i)$ will equal $\frac{m}{n-1}$.

For the case in which $\frac{m}{n-1}$ is not an integer it will not be possible to have $(d_{i-1} - d_i)$ constant for all $i \geq 2$. We will however wish to have it be nearly constant. That is we will want it to be $[\frac{m}{n-1}]$, the greatest integer less than $\frac{m}{n-1}$, for some values of i and to be one greater for others. The construction process will be to

choose some of the $p_i(D)$ to have degree $[\frac{m}{n-1} + 1]$ and the others to have degree $[\frac{m}{n-1}]$. Again the $p_i(D)$ are chosen to be pairwise relatively prime and in this case none of them should be divisable by (D+1). Since $\frac{m}{n-1}$ is not an integer, there exists some integer, k, $0 < k < n-1$, such that $\frac{m+k}{n-1}$ is an integer. We should choose n-1-k of the $p_i(D)$ to have degree $[\frac{m}{n-1} + 1]$ and the other k+1 to have degree $[\frac{m}{n-1}]$. As a consequence of this k+1 of the generator polynomials thus formed will have degree m and the other n-k-1 will have degree m-1. The latter may all be multiplied by (D+1) to increase their degree to m. Either way the differences $d_{i-1}-d_i$ for the generator polynomials will be, as required, either $[\frac{m}{n-1}]$ or $[\frac{m}{n-1} + 1]$ for $i \geq 2$.

Upon applying Theorem (4.22) we see that these codes do achieve the lower bound on consecutive zeros as given in Theorem (4.13). Consequently, whenever it is possible to find the required set of pairwise relatively prime polynomials $\{p_1(D), p_2(D), \cdots, p_n(D)\}$ we may construct a code which achieves the lower bound of Theorem (4.13). We have thus proved the following theorem:

(4.23) Theorem: For a given rate = $\frac{1}{n}$ and constraint length = m+1 let K, $0 < K \leq n-1$, be that integer such that $\frac{m+K}{n-1}$ is an integer. If there exist n polynomials $\{p_1(D), p_2(D), \cdots, p_n(D)\}$ such that

$$\text{degree } p_i(D) = \begin{cases} \frac{m+K}{n-1} - 1 & i=1,2,\cdots,K+1 \\ \frac{m+K}{n-1} & i=K+2,\cdots,n \end{cases}$$

and which are pairwise relatively prime, then there is a code which has no more than $\{\frac{m}{n-1} - 1\}$ consecutive zero weight branches.

Therefore, we have a class of convolutional codes which share the property of having the fewest possible number of consecutive zero weight branches. The above construction is not always possible (for low rates and short constraint lengths) but in the case of long constraint lengths and moderately high rates there will be no trouble in finding the appropriate starting polynomials. In that case it will usually be possible to choose them to be irreducible since the number of irreducible polynomials grows rapidly with degree (Peterson (1961), Berlekamp (1968)).

APPENDIX TO CHAPTER IV

In this appendix we will prove the theorem given in the text, namely:

(A.1) Theorem: Let $\{g_1(D), g_2(D), ---, g_n(D)\}$ be a set of n polynomials such that degree $[g_i(D)]=m$ for $i=1,2,---,n$.

Let $\delta_i(D)$ be the greatest common divisor of $\{g_1(D), g_2(D), ---, g_i(D)\}$ for $i=1,2,---,n$ and let d_i = degree $[\delta_i(D)]$.

Then if p(D) is any polynomial of degree < 2m which is a multiple of $\delta_n(D)$ it has a representation of the form:

$$p(D) = a_1(D)g_1(D)+a_2(D)g_2(D)+ --- + a_n(D)g_n(D)$$

such that

$$\text{degree } [a_i(D)] < d_{i-1}-d_i \quad \text{for } i=2,---,n$$

$$\text{and degree } [a_1(D)] < m$$

In particular, we are interested in the special case in which $\delta_n(D) = 1$.

For n=2 this theorem is well known and may be found in the following form which we will need:

(A.2) Theorem: Let $g_1(D)$ and $g_2(D)$ be two polynomials of degrees m_1 and m_2 respectively such that g.c.d. $\{g_1(D), g_2(D)\}= \delta(D)$ of degree d. Then any polynomial, p(D), a multiple of $\delta(D)$ of degree < m_1+m_2-d may be written as:

$$p(D) = a_1(D)g_1(D) + a_2(D)g_2(D)$$

such that

$$\text{degree } [a_1(D)] < m_2 - d$$

$$\text{degree } [a_2(D)] < m_1 - d$$

We will prove the general theorem by induction on the number of polynomials, n. First we will assume that it is true for all $n \leq N-1$ and show that it is necessarily true for $n=N$ ($N \geq 3$).

Since p(D) is some multiple of $\delta_N(D)$ there exists some representation

$$p = \alpha_1 g_1 + \alpha_2 g_2 + \text{---} + \alpha_N g_N$$

where the $\{\alpha_i\}$ are polynomials of arbitrary degree.

we may rewrite this as

$$p = P_{N-1} + \alpha_N g_N$$

and we may further rewrite $\alpha_N g_N = q\,\delta_{N-1} + r$ degree $(r) < d_{N-1}$

so $p = P_{N-1} + q\delta_{N-1} + r$

now notice that g.c.d. $\{g_N, \delta_{N-1}\} = \delta_N$ and $\delta_N \mid r$

so $r = \alpha\delta_{N-1} + \beta g_N$

with degree $\{\alpha\} < m - d_N$ degree $\{\beta\} < d_{N-1} - d_N$.

If we combine these terms we have

$$p = P_{N-1} + q\delta_{N-1} + \alpha\delta_{N-1} + \beta g_N$$

but $\delta_{N-1} \mid (p_{N-1} + q\,\delta_{N-1} + \alpha\delta_{N-1})$

and degree $\{p_{N-1}+q\delta_{N-1}+ \alpha\delta_{N-1}\} < 2m$

so by the induction hypothesis we have:

$$p_{N-1}+q\,\delta_{N-1}+\alpha\delta_{N-1} = a_1g_1+a_2g_2+\text{---}+ a_{N-1}g_{N-1}$$

such that degree $\{a_1\} < m$

degree $\{a_i\} < d_{i-1}-d_i \quad i=2,\text{---},N-1$

and since degree $\{\beta\} < d_{N-1}-d_N$ the proof is complete.