NASA TECHNICAL REPORT

NASA TR R-294

C. 1

NASA TR R-294

A COMPARISON OF SEQUENTIAL DECODING METRICS BY COMPUTER SIMULATION

by Thomas V. Saliga

Goddard Space Flight Center
Greenbelt, Md.

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • JANUARY 1969

A COMPARISON OF SEQUENTIAL DECODING METRICS BY COMPUTER SIMULATION

By Thomas V. Saliga

Goddard Space Flight Center
Greenbelt, Md.

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 – CFSTI price $3.00

ABSTRACT

Sequential decoding of convolutional coded data offers essentially error-free communication at rates within 1/3 of channel capacity, thus making it attractive for space and other communication systems. Sequential decoding is a sub-optimum decoding technique that sequentially estimates transmitted symbols, using an appropriate confidence measure or metric. An a posteriori probability metric has been generally employed. However, the derivation of this metric, the exact system performance, and the operational sensitivity of a sequential decoder to the choice of its metric have not been adequately treated.

This paper determines a sequential decoder's performance and metric sensitivity by means of a computer simulation using two metrics:

(1) A log-a-posteriori probability metric, and

(2) A cross-correlation metric.

Both metrics are defined, derived, and tabulated for the memoryless gaussian channel. Simulations of a rate 1/2, constraint-length 32 coded data system are made using 16 level quantized metrics. When good metric and decoder parameters have been found, the decoder's computational load, overflow probabilities, and error probability are found as a function of channel signal-to-noise ratio, using at least 500 simulated telemetry frames per data point. It is shown that the correlation metric is inferior to the probability metric by at least 1.5 decibels and suffers a higher error rate. In addition, the correlation metric decoder degrades intolerably with 0.5 decibel signal-amplitude fluctuations, whereas the probability metric decoder is negligibly affected.

CONTENTS

A COMPARISON OF SEQUENTIAL DECODING METRICS BY COMPUTER SIMULATION*

by
Thomas V. Saliga
Goddard Space Flight Center

INTRODUCTION

A branch of communications theory known as coding theory has received considerable attention since the results of C. E. Shannon were published in 1949 (Reference 1). Much effort has been and is attracted to coding theory because of the possibility of error-free communications suggested by Shannon's "Noisy Coding Theorem." This theorem states that data can be so encoded for transmission over a noisy channel that the probability of a decoding error is arbitrarily small, provided that the data rate is less than a rate called the *channel capacity*. The converse of this theorem is important in space communications: Channel capacity is the highest data rate at which the probability of error can be made arbitrarily small.

Achieving maximum data rate in a space communications system through coding is equivalent to minimizing the transmitter power and weight and, therefore, the cost of a spacecraft. The difference between the channel capacity obtained by coding and that of a typical uncoded system is a factor of about 20 for acceptable error probabilities. The possibility of achieving a 20-times-improved communications system is obviously stimulating and has led to the study and invention of many encoding and decoding schemes.

In space communications coding, the M-ary block codes at first received considerable attention (References 2, 3, 4, and 5). However, their limited coding gain, large bandwidth requirements, and special synchronization needs kept them from being generally attractive.

Based upon work by Wozencraft (Reference 6), Fano (Reference 7), and Blustein and Jordan (Reference 8), I. M. Jacobs suggested the use of convolutional encoding with sequential decoding for space communications (Reference 9). This coding technique offers both large coding gain (nearly 10 times) and modest bandwidth requirements. The convolutional encoder is easy to implement and the sequential decoder can be implemented with a small general-purpose computer in many applications.

*Thesis submitted to the Faculty of the Graduate School of the University of Maryland in partial fulfillment of the requirements for the degree of Master of Science, 1968.

Sequential decoding is a sub-optimum decoding technique first studied by Wozencraft (Reference 6). An elegant algorithm due to Fano (Reference 7) in 1963 has made the decoder reasonably easy to implement. This algorithm estimates transmitted symbols as they are received in a serial or sequential fashion. If an error is made, subsequent measurements tend to indicate this fact. The algorithm then takes corrective action by backing up and re-estimating the transmitted symbols. It will then continue to estimate symbols until later measurements again indicate a lack of confidence. This "estimate, check, re-estimate if necessary" procedure continues until all received symbols have been decoded. Each symbol estimate has a confidence measure associated with it. This measure, or *metric,* could be Hamming distance, correlation, or an a posteriori probability.

The analytical and simulation work on sequential decoding of binary signals in the literature to date has employed an a posteriori probability metric. However, the derivation of this metric and the operational sensitivity of a sequential decoder to the choice of a metric have not been adequately treated; doubtless because such treatment is a difficult analytical problem.

In particular:

(1) The performance specification and optimum parameters for a sequential decoder using the probability metric or other metric are incomplete;

(2) The advantages or disadvantages of using metrics other than the a posteriori probability metric are not specified;

(3) The a posteriori probability metric is not derived and tabulated for the space channel; and

(4) The sensitivity of a sequential decoder's performance to amplitude fluctuations and non-optimum metric parameters is not specified.

The primary purpose of this paper is to determine the sensitivity of a sequential decoder's performance to the choice of a metric, doing so by means of a computer simulation using two metrics:

(1) A log-a-posteriori probability metric, and

(2) A cross-correlation metric.

In the above process, the sequential decoder's performance is determined, the required metrics are derived and tabulated, good metric parameters are found, and the decoder's sensitivity to amplitude fluctuations is noted.

The selection of the two metrics named is due to the fact that an optimum decoder works equally well using either an a posteriori probability or a correlation measure as a basis for signal estimation. Since sequential decoding is a sub-optimum procedure, however, it does not necessarily follow that it works equally well with either metric.

The computer simulation encodes pseudorandom binary data into a rate 1/2 convolutional code. A binary antipodal modulator-demodulator brackets an additive gaussian noise channel. The

sequential decoder's average computational load, error probability, and overflow probabilities are found for both metrics, after appropriate parameter optimization. Tests are also made to check the repeatability of the statistics, and the signal amplitude sensitivity of each metric.

CONVOLUTIONAL ENCODERS

A convolutional encoder is a type of sliding-parity check calculator. It was first introduced and investigated by Elias (Reference 10). Restricting our attention to digital binary systems, and serial binary data sources, we have a simple example of a convolutional encoder in Figure 1.

Two flip-flops, F_1 and F_2, are connected as a shift register. Modulo 2 addition gates are connected to the register in some prescribed manner to calculate a parity symbol, P(i). This parity symbol is thus a function of the current information bit, D(i), and the two previous information bits, D(i - 1) and D(i - 2). The output switch transmits the two symbols D(i), P(i) during the interval of one input information bit.

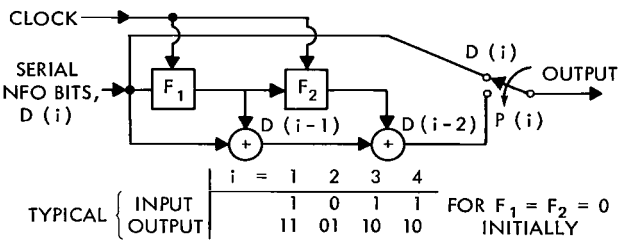

Figure 1—Binary convolutional encoder.

Since two symbols are transmitted for each input bit, the encoder may be called a "bandwidth expansion = 2" encoder. However, the literature ordinarily uses the reciprocal of this number and calls it a rate 1/2 encoder. Thus, by definition, the *code rate (R)* equals the reciprocal of the number of code symbols transmitted for each input information bit.

Since the parity symbol calculation is based on the current information bit and the two previous ones, the encoder's "constraint-length" is said to be 3. By definition, the encoder's *constraint-length (K)* is the largest span of information bits over which at least one parity symbol is calculated. This is simply 1 plus the number of shift-register flip-flops (a connection to the rightmost flip-flop implied).

Note that one of the transmitted symbols is the information bit itself. If one of the 1/R symbols per transmitted bit is the information bit itself, then the code is called a *systematic code;* otherwise, the code is called *nonsystematic.*

A more generalized convolutional encoder can now be envisioned and is illustrated in Figure 2. The shift register clock line and interconnections have been deleted for simplicity. The code rate, constraint-length, and connections to the modulo 2 adders all are important in determining performance of the coded communications system.

The encoder's connections may readily be described by using a "V" row by "K" column matrix. This connection matrix or *generator matrix (G),* as defined in this paper, is simply a "picture" of the connections to the adders that uses the form shown in Figure 2. A binary 1 implies a connection and a binary 0 implies no connection. Thus the G matrix for the example in Figure 1

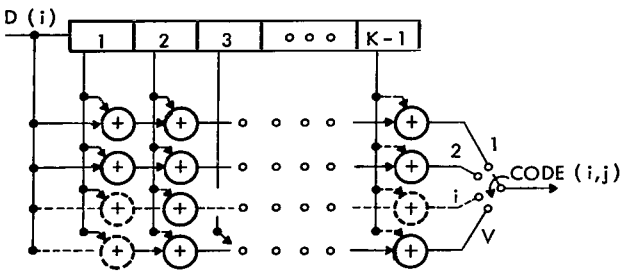

Figure 2—Generalized convolutional encoder.

is:

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

It is apparent that the generator matrix *completely* specifies the encoder's rate, constraint-length, and internal connections. The single connection in the first row in the above example also indicates that it is a systematic code. The flip-flops are assumed to have a "0" initial state.

The particular encoder used in this simulation is described by a two-row 32-column matrix. Thus the "bandwidth expansion" is only 2, and each code symbol is a function of 32 information bits.

IMPORTANT CONVOLUTIONAL CODE PROPERTIES

There are two important properties of convolutional codes fundamental to their use with a sequential decoder:

a. The tree property, and

b. The Hamming distance, or correlation properties, of the code's tree.

The practical problem of terminating the code must also be considered.

The Tree Property

Given the code's generator matrix and that the flip-flops are initially in the binary zero state, then the very first input data bit, D(1), can only give rise to one of two possible output vectors—"V" output symbols. When the second data bit is input to the encoder, then another two output vectors are possible. However, these two vectors depend on what D(1) was. Hence this time there are really four total possibilities. It is apparent that the number of possibilities doubles for each new data bit entered into the encoder. These possibilities may be shown exhaustively, for at least a few bits, using a "code tree" diagram. Using the code in Figure 1 as an example, such a code tree is depicted in Figure 3.

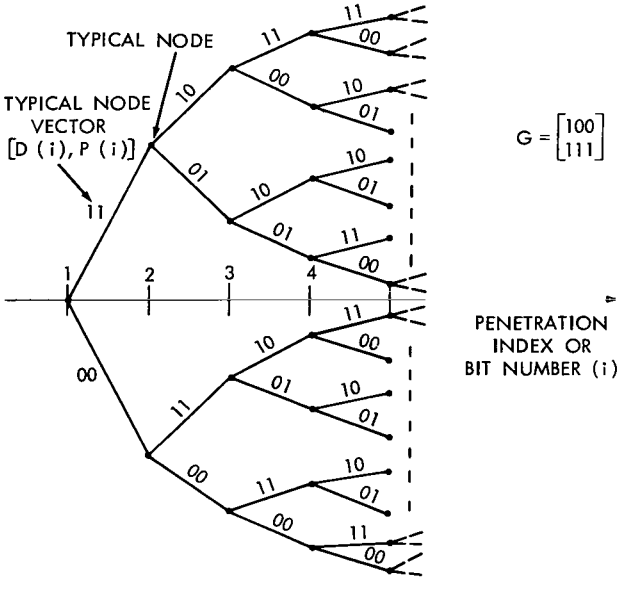

Figure 3—Convolutional code tree.

Each path through the tree is called a *branch,* and the junctions of branches are called *nodes.* Each of the $v(= 2)$ symbols associated with each data bit is called a *node vector.* Notice that any node vector may be calculated from knowing only the current information bit and the previous $K - 1$ bits.

Hamming Distance, or Correlation Properties

The second important property of a convolutional code is the Hamming distance,* or correlation coefficient, between branches of the code's tree. Indeed, it is the increased Hamming distance, or—equivalently—lower cross-correlation, between different branches in the tree which distinguishes a "coded" from an uncoded communications system. Using as an example the tree code shown in Figure 3, compare any two branches which differ in at least the first data bit. The comparison should be made only over one constraint-length. The uppermost branch and a lower one are compared below.†

	D_1	P_1	D_2	P_2	D_3	P_3	D_4	P_4	
Branch 1	1	1	1	0	1	1	1	1	..
Branch 2	0	0	1	1	1	0	1	1	..
Hamming Distance	(1	1	0	1	0	1)			= 4

Although the information sequences differ in only one digit, the additional parity checks at and beyond that *first* information digit difference do not generally agree. Clearly this condition will exist as long as the information digits in the encoder's shift register are different. Thus, making the constraint-length large will insure parity disagreements between such branches for a considerable penetration into the coding tree. The $K = 3$ encoder used in this example does not "remember" the disagreement in data bit D_1 when D_4 is put into the encoder. So the correct and incorrect branches will contain identical symbols beyond that point forever after as long as subsequent data bits are the same.

As with block codes, when the information sequences differ in at least one digit, it is desirable to choose the code so as to maximize the minimum Hamming distance between signals. This is equivalent to saying that their cross-correlation should be made uniformly as low as possible. An "ideal" convolutional encoder, in the author's opinion, can be better understood using a correlation description.

Let cross-correlation between binary sequences A, B of "n" digits each be defined as

$$\rho(A, B) = \frac{n - 2 \text{ weight } (A \oplus B)}{n}$$

*The Hamming distance between two binary vectors is equal to the number of digits in which they differ when compared digit by digit.
†Throughout this paper, D_1 is interchangeable with D(1), P_4 with P(4), etc. The dual forms are used to meet the demands of mathematical statement, on the one hand, and of programming language, on the other.

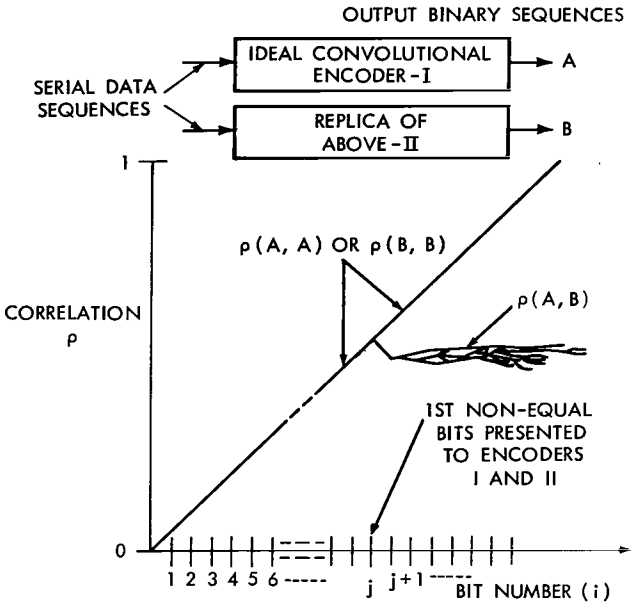

Figure 4—Cross-correlation of ideal convolutional codes.

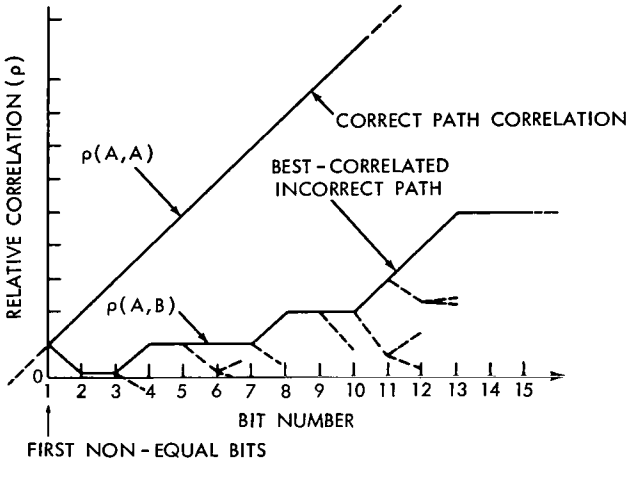

Figure 5—Branch correlation of a rate 1/2 code.

where

weight $(A \oplus B)$ is equivalent to Hamming distance.

Then

$$-1 \leq \rho(A, B) \leq 1 .$$

Now let two "ideal" convolutional encoders receive exactly the same data bit sequence up to the j^{th} bit. Let the j^{th} bits put into the two encoders be different, and let all subsequent data bits be arbitrary. Then their desired correlation characteristics are shown in Figure 4. Notice that, when $i = j$, the "different" pair of digits are introduced into the encoders and their cross-correlation thereafter *becomes non-increasing* for *any* subsequent data digits.

Encoder I may be interpreted as a transmitter, and encoder II as part of a receiver or decoder estimating what was transmitted. Then it is this sustained loss in correlation after bit "j" that can allow a decoder to recognize a poor previous data bit estimate and take corrective action.

To obtain a non-increasing branch cross-correlation, the code rate must be very small. A more practical rate 1/2 code gives inferior, but acceptable, correlation characteristics. Typical incorrect correlation for the rate 1/2 code used in these simulations is shown in Figure 5. The best-correlated incorrect path shown was found by inserting into encoder II those data bits which maximized each node correlation.*

Obtaining generator matrices which minimize the correlation of the best-correlated incorrect path is still a topic of research. Matrices for "good" codes have been presented by Bussgang (Reference 11), Lin and Lyne (Reference 12), and Massey (Reference 13).

*A better correlated path may exist, however, if the constraint of maximizing each node correlation sequentially is not imposed.

unable to recover. The causes of this problem will become clear later. However, the important points to be made here are how the encoder is flushed and the effects on the code's tree.

The convolutional encoder is flushed by terminating its input data sequence and inputting some *known* sequence of bits. This known sequence may simply be the all-zeros sequence. Since an encoder of constraint-length K has a "memory" of K - 1 previous information bits, the known sequence need only have length K - 1 to clear out all "history" of past data. During the interval when the known bits are being shifted into the encoder, transmission of the encoder output into the channel continues as before. The *tail symbols* are defined to be those symbols output from the encoder during this flush interval.

If the code rate is 1/V, then there are V(K - 1) tail symbols. If the code is systematic, then one of the symbols is known and it is really only necessary (and more efficient) to transmit the (V - 1)(K - 1) parity symbols.

The code tree ceases to grow in the flush interval. With each input being known and fixed, there is no alternate path. Thus the node vector associated with the last data bit in the tree actually has KV symbols. This gives the final bits in the tree the full Hamming-distance benefit of at least one constraint-length of parity symbols. Truncating the tail length would tend to increase the probability of error of the final bits in the tree relative to that of previous bits.

THE OPTIMUM DECODER

The optimum receiver or decoder of a data sequence $[D(i),\ i = 1, 2, \ldots]$ is here defined to be some processor which produces an estimated data sequence $[\hat{D}(i),\ i = 1, 2, \ldots]$ that is optimum in the sense of minimizing bit error probability. That is,

$$P[D(i) \neq \hat{D}(i)] \tag{1}$$

is minimized.

Let S(i) denote the transmitted signal associated with the i^{th} data bit, and R(i) the corresponding received signal. Then it is known that the optimum decoder is one which maximizes the a posteriori probability with the given received signals. That is, $[\hat{D}(i),\ i = 1, 2, \ldots]$ is selected so that

$$P\left\{[\hat{D}(i),\ i = 1, 2, \cdots] \mid [R(i),\ i = 1, 2, \cdots]\right\} \tag{2}$$

is maximized relative to any other choice of the $\hat{D}$ sequence. Using Bayes' Theorem and assuming that $P[D(i) = 0] = P[D(i) = 1] = 1/2$, then (2) may be rearranged to give the decoder decision function. Also, from the fact that the a posteriori probabilities are a monotonic increasing function of cross-correlation, the optimum decoder's decision function is: Select $[\hat{S}(i),\ i = 1, 2, \ldots]$ such

that

$$\rho\left(\left[\hat{S}(i),\ i = 1, 2, \cdots\right],\ \left[R(i),\ i = 1, 2, \cdots\right]\right)$$

is maximum. Then the data digits associated with $\hat{S}(i)$, namely $\hat{D}(i)$, are defined by the encoding process where

$\hat{S}(i)$ = the encoder output signal associated with the i^{th} data bit hypothesis,

$\rho(A, B)$ = the cross-correlation between signals A and B, and

$R(i)$ = the received signal associated with the i^{th} data bit.

The above decision function may be simply restated in terms of convolutional codes. If a sequence of, say, 100 data bits, $D(i)$, is shifted into an encoder, then there will be 100 node vectors, $S(i)$, transmitted. After being corrupted by additive noise, the received node vectors, $R(i)$, are cross-correlated with all possible transmitted sequences, $[\hat{S}(i),\ i = 1, 2, \ldots, 100]$. The $\hat{S}(i)$ sequence yielding the largest correlation coefficient is the best signal estimate. The 100 information bits which generated $[\hat{S}(i)]$, namely $[\hat{D}(i)]$, are then the estimated data bits.

Notice, however, that a serious implementation problem results since all 2^{100} cross-correlations must be performed in search of the largest. Because of this problem, any practical decoder of convolutional codes must use sub-optimum techniques.

Two practical decoding schemes for these codes are threshold decoding and sequential decoding. Sequential decoding is the more efficient of the two in the sense of minimizing signal-to-noise ratio threshold.

SEQUENTIAL DECODING OF BINARY CODES

Sequential decoding is an efficient decoding procedure which sequentially investigates paths through a tree code to make local data bit estimates. Bad estimates are sensed by a sustained loss in correlation (or metric value) between a local code generator and the received code as the decoder attempts to move ahead in the tree. Under control of an algorithm, the "investigation" moves back down the code tree, and systematically tries to find another tree path which does not have a sustained low correlation. The sequential decoder therefore attempts to find the highest correlated tree branch, like an optimum decoder. It does so, however, without having to investigate *all possible* branches of the code tree. If the noise is not too large, then the number of tree paths investigated will increase only linearly with the number of bits decoded.

System Diagram and Decoder Metric

Since the decoder's "investigation" of the code tree will move back and forth in the tree, some form of temporary memory for the received signals and data estimates is implied. A local replica

of the convolutional encoder used in the transmitter is necessary for comparing with the received signal. A simplified block diagram of a sequential decoding system, illustrating its basic components, is shown in Figure 6.

It is assumed that a sequence of M digits is encoded, transmitted, and received. The elementary code symbols are correlated in a matched filter, and the symbol correlation coefficients, $R_\rho(i, 1)$, are stored in the decoder's memory. Under algorithm control, the locally estimated node vector, $[\hat{S}(j, 1), 1 = 1, 2, \ldots, V]$, is compared with the corresponding received node vector, using some appropriate measure. The comparison measure or metric may be simply the cross-correlation coefficient, or a probability measure. This *sequential decoding metric and its effect on system performance are the main topics of this paper.* It will be described herein more completely later.

The estimated data register is capable of being shifted right or left under algorithm control. Shifting right corresponds to advancing into the code tree. When all M digits have been decoded, then $\hat{D}(1)$ is in the rightmost position and $\hat{D}(M)$ in the leftmost position, within the local encoder's shift register.

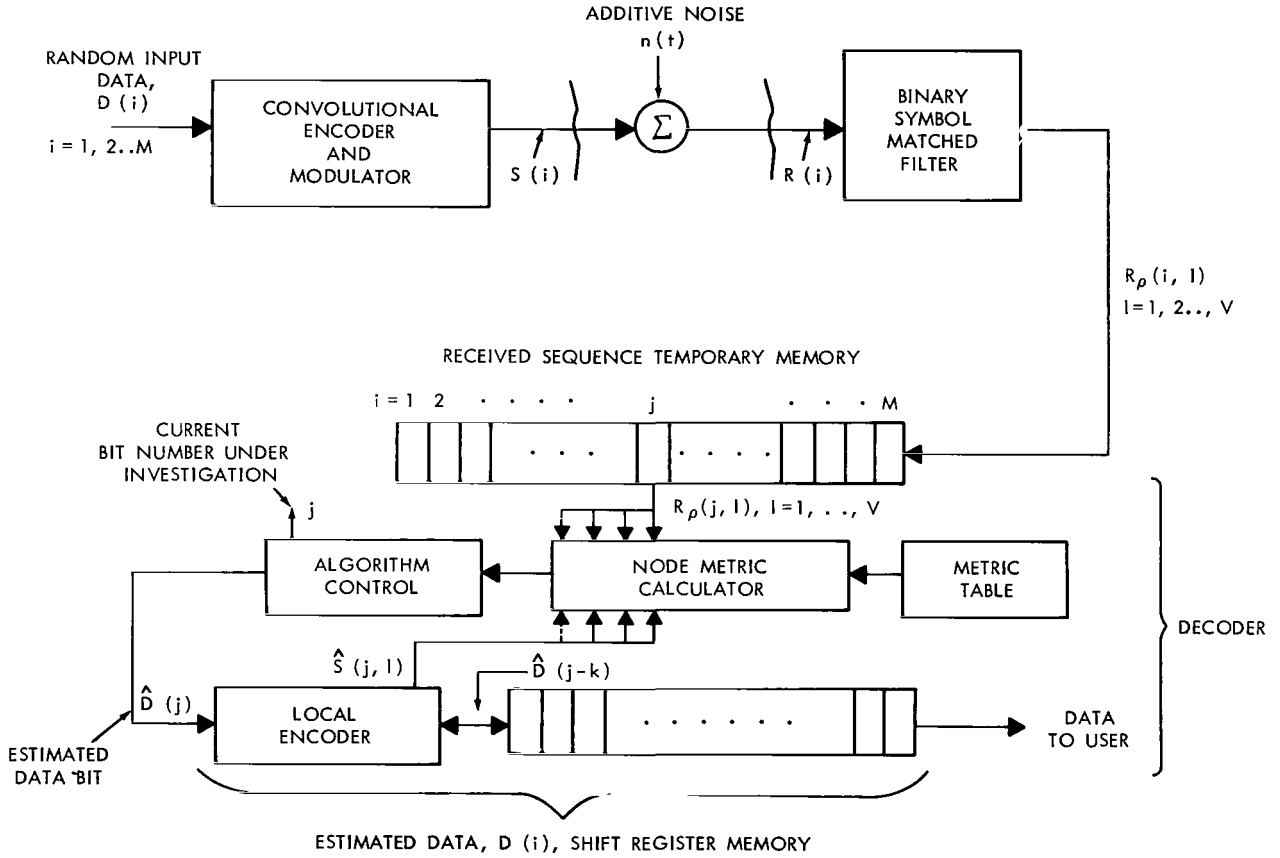

Figure 6—Simplified encoder-sequential decoder system.

The Tree Search Algorithm

The rules considered here for searching the tree paths are known as the Fano Sequential Decoding Algorithm. A flow diagram for this algorithm is shown in Figure 7. An artifice characterizing this algorithm is the use of a *threshold level (T)*. The decisions to move forward or backward in the code's tree are controlled by comparing the *accumulated node metric (Accum)* with this threshold. Accum will also be referred to simply as the *branch metric*. The flow diagram contains two primary loops: a move-forward loop and a search-mode loop. If the signal-to-noise ratio is high, then all data hypotheses will be correct, Accum will be greater than T, and the rules in the move-forward loop will be executed repetitively until all bits have been decoded.

This algorithm depends on the node metric being selected in such a way that when

(a) all estimated data bits are selected without error, the accumulated node metric tends to increase in value; and

(b) when one or more past bit errors exist, the accumulated node metric tends to decrease in value.

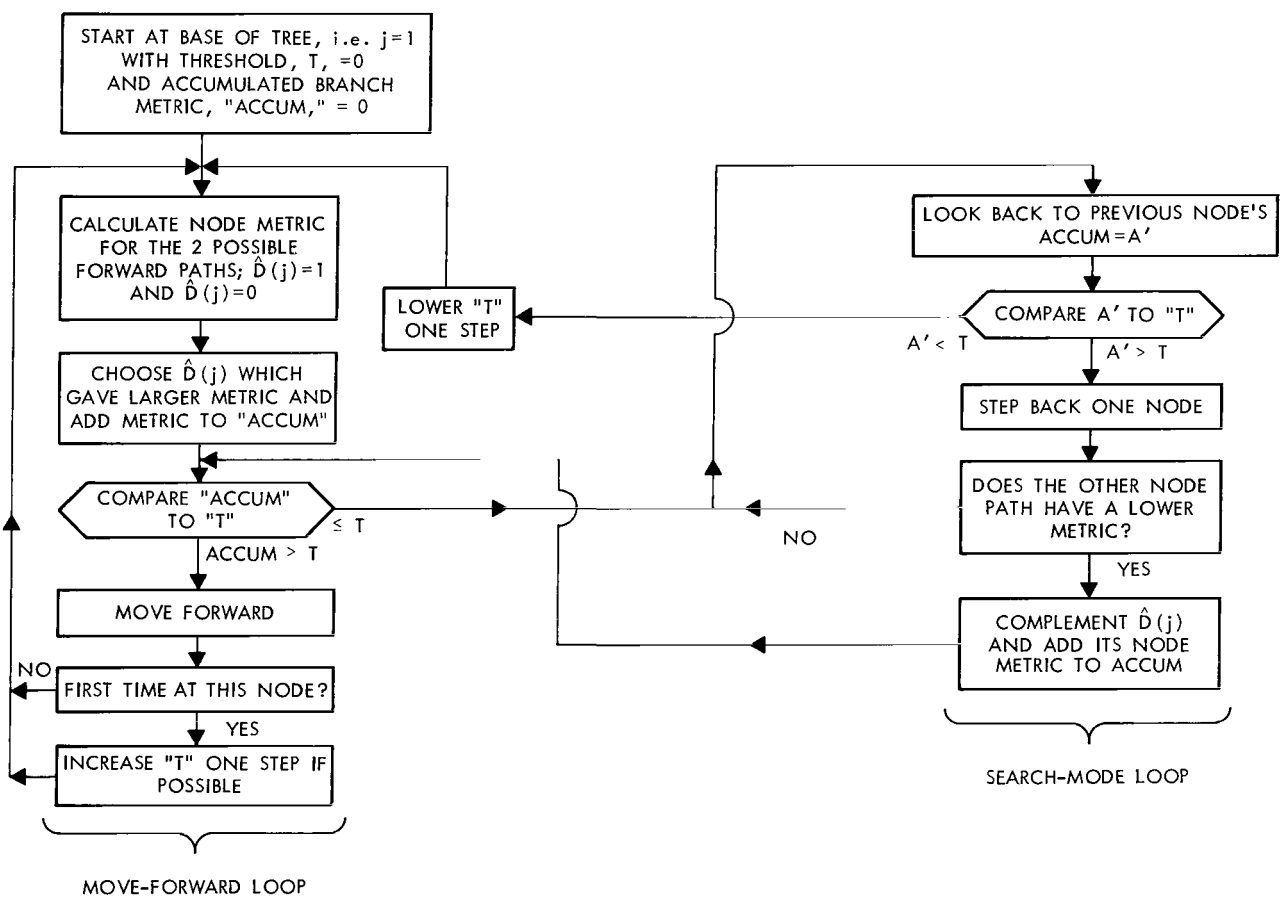

Figure 7—Flow diagram of Fano Sequential Decoding Algorithm.

The branch correlation as shown in Figures 4 and 5 can fulfill the above conditions if some appropriate constant is subtracted from each node correlation.

Thus, when some previous $\hat{D}(i)$ is in error, Accum should become less than T and the algorithm will initiate the search-mode loop. Basically, this loop forces the decoder to try alternate tree paths, attempting to find one which has a non-decreasing metric. An important feature of this search operation is that the threshold is never lowered until some previous node's accumulated metric, A', is less than the current threshold. The only way this can happen is for the decoder to have searched *all* possible available tree paths which have an Accum greater than T. This search-mode operation is probably best understood through an example.

Examples of Tree Searching

If a plot of the accumulated node metric (Accum) and threshold (T) versus the data bit number, (i), is made under high signal-to-noise ratio conditions, it might appear as in Figure 8.

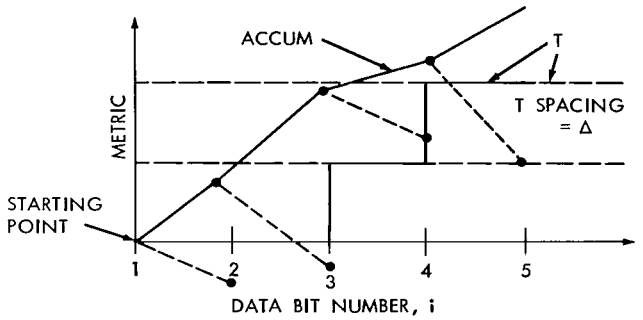

Figure 8—Typical Accum and T behavior with low noise.

At each new node, the node metric is found by trying $\hat{D}(i) = 1$ and 0. The larger metric is added into Accum and its associated $\hat{D}(i)$ is left in the local encoder. The smaller metric is discarded; but, if it were added into Accum, it would give the dotted paths shown. The threshold is increased in discrete steps, Δ, provided Accum still exceeds the new threshold. Since the signal-to-noise ratio is high, no errors in estimating D(i) occur and Accum continues to increase. Only the move-forward loop of the algorithm is then utilized.

With lower signal-to-noise ratios, occasional data bit estimates are in error and the search-mode loop of the algorithm must be employed to correct them. An example of decoder operation when a single bit error occurs follows.

Referring to Figure 9a, let bits 1 and 2 be hypothesized correctly, and let the noise sufficiently corrupt the received node vector associated with bit 3 so that the larger of two node metrics belongs with the incorrect data bit. Then, as shown in Figure 9a, the decoder moves forward through steps 1, 2, 3, and 4. However, when investigating bit 5, it finds that the largest forward Accum *violates* the current threshold; that is, (Accum < T). This causes loop 2 of the algorithm to be executed. The decoder now backs down the tree to the previous node, namely bit 4. After confirming Accum is still > T, it investigates the lower metric path—labeled 6. The Accum on this lower path also violates the threshold, returning decoder control to the search-mode loop. The decoder again backs up one bit. As shown in Figure 9b, the Accum at bit 3 is above the threshold, so that lower node metric path, labeled 7, is investigated and compared to the threshold. It also violates the threshold, returning decoder control to the search-mode loop.

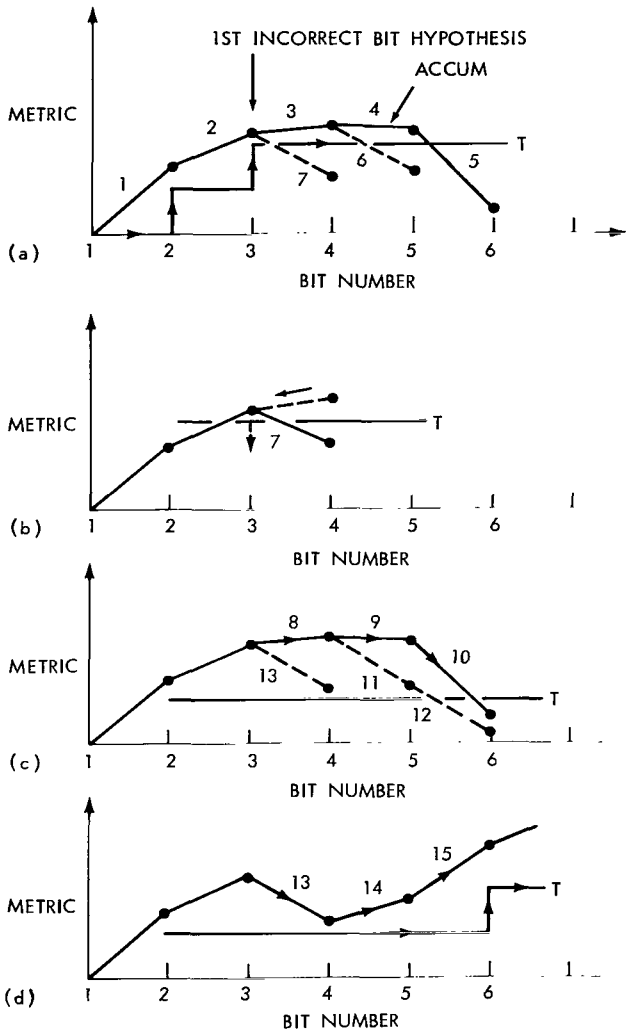

Figure 9—Sequential decoder operation analysis with bit hypothesis 3 initially in error.

The decoder now "looks back" from node 3 to the Accum of node 2. As seen in Figure 9b, that node Accum violates the threshold. The threshold is then lowered one step while staying at node 3, and decoder control given to the "look forward" loop.

The decoder now retraces all of its previous move-forward steps as shown in Figure 9c by paths 8, 9, 10. The thing that is different is the lower threshold. The *threshold is prevented from increasing by the question "First time at this node?" in the algorithm.* Again the search-mode loop backs up the investigation to bit number 4 and forces investigation of the complement path, labeled 11. Since path 11 is above the threshold, the decoder looks forward on the largest path, labeled 12. When path 12 violates threshold, the search-mode loop backs up investigation to bit 3 and path 13. The move-forward loop then investigates paths 13, 14, 15 and further. At bit 6, the decoder recognizes both that it is the first time at that node, and that a threshold increase is possible. T is then increased one step.

Move-forward operation continues until the accumulated node metric, Accum, again violates the threshold, or, of course, until the last data bit is received.

The question "First time at this node?" appears to be formidable, since a large memory of past investigations is implied. Indeed, if it were not for the ingenious use of a particular Accum and T relationship, it would be difficult to implement. Actually it is necessary only to ask the question "Is the previous node's Accum smaller than the current threshold plus a threshold step?" If the answer is "No," and T could otherwise be increased, then the present node must have been investigated previously or a threshold separation exceeding one step could not exist.

Characterization of Decoder Performance

An important decoder performance measure for any data system is the bit error probability versus system parameters. Since a sequential decoder attempts to detect and correct errors, it is more appropriate to say "undetected bit error probability." If D_i, $i = 1, 2, \ldots$, is the encoder's

input data sequence and $\hat{D}_i$, i = 1, 2, . . . , is the decoded output data sequence, then define the *bit error probability* as

$$P_{Bit}(\epsilon) = P_B(\epsilon) = P(D_i \neq \hat{D}_i) \text{ for any } i .$$

The encoder is periodically flushed and reset. Each such interval comprises a frame of data. Thus another error measure of interest is the frame error probability. The *frame error probability* is defined as

$$P_{Frame}(\epsilon) = P_F(\epsilon) = P(\text{one or more bit errors in a frame}) .$$

As shown earlier, under "Examples of Tree Searching," the decoder algorithm must perform a search whenever bit estimates are initially in error. A period of high channel noise may require a large amount of searching, and hence a large amount of computer time. It is therefore important to characterize the computational behavior of the decoder as a function of signal-to-noise ratio.

Define a decoder *computation* as the sequence of operations necessary to complete one search-mode loop or one move-forward loop in the decoding algorithm. Then knowing how many computations a decoder must perform to decode some received signal gives proportional measure for decoding time and telling how "hard" the decoder is working.

In particular, it is desirable to find the probability distribution associated with the number of decoder computations (C). Define a *frame* as the signal sequence transmitted between encoder resets. This frame is to include the tail symbols transmitted by the encoder as it is flushed and then reset. The most useful computation distribution is that which gives the fraction of frames requiring more than L computations. That is,

$$P(C > L) = f(\text{SNR, system parameters})$$

where

C = the number of computations required to decode a sequence of data, and
L = a variable limit.

If a decoder is allowed to make no more than "L" computations because of time limitations, etc., when attempting to decode a sequence of data, *"L" is called the overflow limit.* Then *P(C > L) is the overflow distribution.* It is desirable to characterize the average computational load of the decoder as

$$\bar{C}_B = \frac{E(C)}{n_d} \text{ (computations per bit)}$$

where

$\bar{C}_B$ = average number of decoder computations per received bit,

$E(C)$ = expected value of C, and

n_d = total number of data bits received.

In this paper, the three decoder characteristics mentioned will be used to describe the decoder, and thus the system performance. Again they are:

(1) undetected bit, $P_B(\epsilon)$, and frame, $P_F(\epsilon)$, error probability;

(2) overflow distribution = $P(C > L)$; and

(3) average computations per received bit = $\bar{C}_B$.

THE LOG-A-POSTERIORI PROBABILITY METRIC

During sequential decoding on a continuous, binary-symbol communications channel, each received symbol has two weightings attached to it. One weight is a measure of how confident we are that a binary 1 was transmitted. Similarly, the other weight is a measure of how confident we are that a binary 0 was transmitted. If the weights are proportional to the symbol's matched-filter correlation voltage, then the decoder is said to be using a "correlation metric." Another measure, however, is one where the weightings are proportional to the probability that a 1 or 0 was sent, given the correlation voltage. This is called an "a posteriori probability metric" and variations of it are among the best known for use in sequential decoding.

We shall now describe and tabulate the log-a-posteriori probability (LAP) metric as used in the simulations.

The LAP metric is described assuming a typical binary PCM space communications channel. Because the metric depends on the particular channel assumptions made, the system model and channel transition probabilities are defined previous to the description and tabulation of the metric.

The Channel Model

The mathematical model of the communications channel assumed is shown in Figure 10. For each data symbol, D_i, entering the encoder, V binary symbols are transmitted. The transmitter sends plus or minus $\sqrt{S}$ volts for a 1 or 0 symbol respectively. The modulation power is then S watts. Additive noise, $n(t)$, has N_0 watts/Hz single-sided power spectral density. A gain-controlled amplifier in the receiver maintains a constant signal amplitude into the filter. The matched filter is sampled at the end of a symbol period, giving a correlation voltage, C_{ij}. Because the AGC amplifier estimates the amplitude of $S(t)$ rather than $r(t)$, the mean value of C_{ij} is some essentially constant positive or negative value.

Indices "i" and "j" denote the j^{th} symbol associated with the i^{th} data bit. The correlation, C_{ij}, is a continuous random variable because of the additive gaussian noise.

Given each C_{ij}, the metric is found by hypothesizing a 1 or 0 as being transmitted. Let $\hat{x}$ denote this hypothesis. The symbol metrics, $M_c(\hat{x} \mid C_{ij})$, are then summed by the decoder to compute a branch metric. It is impractical to use continuous values in a digital machine, so the C_{ij} correlations are usually digitized. Using more than about 16 levels (4-bit A to D) gains little in system performance (Reference 14). The quantized correlations, Q_{ij}, are then used to find the quantized metric, $M(\hat{x} \mid Q_{ij})$.

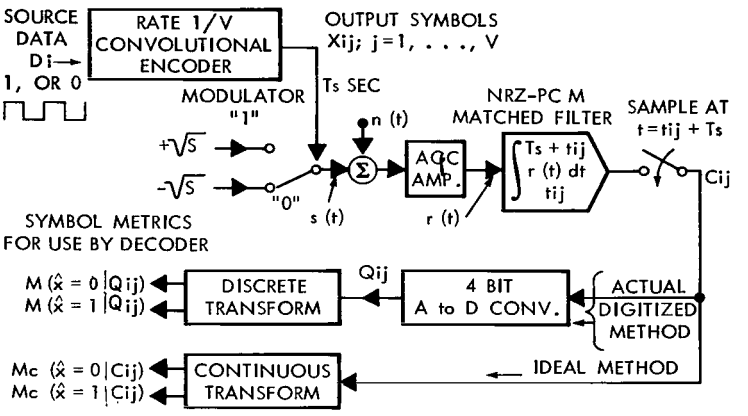

Figure 10—Communications system model.

Channel Conditional Probabilities

The symbol's correlation voltage, C_{ij}, will have a gaussian probability density, since the receiver's amplifier and filter are linear devices. The mean to standard deviation ratio of C_{ij} may be shown to be (see Appendix A)

$$\frac{\mu}{\sigma} = \sqrt{2\,\frac{ST_s}{N_0}} \tag{3}$$

where

μ = mean value of C_{ij},

σ = standard deviation of C_{ij},

S = modulation power in watts,

T_s = duration of binary symbol in sec, and

N_0 = noise density of $n(t)$ in watts per Hz.

If x_{ij} is a binary 1, then μ is some positive voltage. When x_{ij} is a 0, then μ is the same voltage but negative. Let the gain-controlled amplifier adjust this voltage magnitude to 1 volt. Then the conditional densities of C_{ij} are

$$P(C_{ij} \mid x_{ij} = 1) = \text{Normal}\left[\mu = +1, \sigma = \left(2\,\frac{ST_s}{N_0}\right)^{-1/2}\right] \tag{4}$$

and

$$P(C_{ij} \mid x_{ij} = 0) = \text{Normal}\left[\mu = -1, \sigma = \left(2\,\frac{ST_s}{N_0}\right)^{-1/2}\right] \tag{5}$$

The Sequential Decoder's LAP Metric

A sequential decoder is basically a conditional probability computer. In its algorithmic search for the correct path or branch in the coding tree, it computes the probability that the hypothesized symbols, $\hat{x}_{ij}$, could have been transmitted given the received C_{ij}. This conditional branch probability is

$$P(\hat{x}_{1,1}, \hat{x}_{1,2}, \cdots, \hat{x}_{K,V} \mid C_{1,1}, C_{1,2}, \cdots, C_{K,V}) \tag{6}$$

where

$\hat{x}_{ij}$ = the binary symbols hypothesized by the decoder for this branch,

K = the number of information bits shifted into the encoder to generate the branch,

V = the number of encoder symbols transmitted per input information bit, and

C_{ij} = the received symbol correlation voltages.

To simplify notation, let the $\hat{x}_{ij}$ and C_{ij} sequences in Equation 6 be replaced by $\hat{x}_n$ and C_n respectively. The "n" subscript represents a one-dimensional serial ordering of the sequences and furthermore implies $n = KV$. Using the same notation, X_n represents the actual transmitted sequence of KV binary symbols. The conditional branch probability using hypothesized symbols is thus

$$\triangleq P(\hat{X}_n \mid C_n) \tag{7}$$

The probability in Equation 7 is the hypothesized "a posteriori branch probability." It is a measure of how confident we are that $\hat{X}_n = X_n$. For $V \geq 2$, K large, and the noise not too high, we expect $P(\hat{X}_n \mid C_n)$ to be near unity only if $\hat{X}_n = X_n$. Otherwise, it should be small for "good" convolutional codes. An optimum decoder selects $\hat{X}_n$ so as to maximize this probability.

The a posteriori branch probability for true transmitted sequence, X_n, may be found using Bayes' Theorem:

$$P(X_n \mid C_n) = \frac{P(X_n, C_n)}{P(C_n)} = \frac{P(C_n \mid X_n)\, P(X_n)}{P(C_n)} \tag{8}$$

Since the channel model has no memory from symbol to symbol, the symbol probabilities are independent. Thus the sequence conditional probability in Equation 8 may be expressed as the product of the symbol probabilities:

$$\frac{P(C_n \mid X_n)\, P(X_n)}{P(C_n)} = P(X_n) \prod_{m=1}^{m=n=KV} \frac{P(C_m \mid X_m)}{P(C_m)} \tag{9}$$

Now the $P(X_n)$ is just the branch probability. Each branch is a priori equi-probable with random data, and there are 2^K unique branches in an $n(= KV)$ symbol code tree. So

$$P(X_n) = 2^{-K}. \tag{10}$$

To simplify implementation of the decoder, an additive measure of probability is desirable. The logarithm (base = 2) of $P(X_n \mid C_n)$ allows this without disturbing the monotonic nature of the conditional probability. Note that very small branch probabilities become large negative numbers.

Combining Equations 8, 9, and 10, taking the logarithm, and using the double subscript symbol indexing gives

$$\log_2 P(X_n \mid C_n) = -K + \sum_{i=1}^{K} \sum_{j-1}^{V} \log_2 \frac{p(C_{ij} \mid x_{ij})}{p(C_{ij})} .$$

Of course, the actual x_{ij} sequence is not known and it is the task of the optimum decoder to hypothesize these symbols, $\hat{x}_{ij}$, so as to maximize this probability. Thus substituting in $\hat{x}_{ij}$ and letting $K = KV/V$ so it may be combined in the summation gives

$$\log_2 P(\hat{X}_n \mid C_n) = \underbrace{\sum_{i=1}^{K} \underbrace{\left[\sum_{j=1}^{V} \underbrace{\left\{ \log_2 \left[\frac{p(C_{ij} \mid \hat{x}_{ij})}{p(C_{ij})} \right] - \frac{1}{V} \right\}}_{\text{Symbol LAP Metric}} \right]}_{\text{Node LAP Metric}}}_{\text{Branch LAP Metric}} . \tag{11}$$

Thus the decoder computes each symbol log-a-posteriori probability metric and sums them over a node (= V symbols). Information bit hypotheses are based on this node metric. The sum of the node metrics then forms the branch LAP metric (≡ Accum).

The quantity

$$\log_2 \frac{p(C \mid \hat{x})}{p(C)},$$

as in Equation 11, is called the mutual information between x and C, and the quantity $1/V$ is the convolutional code rate. The channel conditional probabilities, $p(C \mid \hat{x} = 1)$ and $p(C \mid \hat{x} = 0)$, have been obtained earlier, under the heading of the same name.

Define

$$\log_2 \frac{p(C \mid \hat{x})}{p(C)} \triangleq I(\hat{x}, C) \tag{12}$$

and the continuous symbol LAP metric $\triangleq M_c(\hat{x} \mid C_{ij})$, then it follows that

$$M_c(\hat{x} \mid C_{ij}) = I(\hat{x}, C_{ij}) - \frac{1}{V} . \tag{13}$$

It is clear from Equation 13 that it is necessary only to evaluate $I(\hat{x}, C)$ for the channel and then the symbol metric, M_c, may be found for any code rate by simply subtracting $1/V$.

The Continuous, Gaussian-Channel Metric

The symbol LAP metric, $M_c(x \mid C_{ij})$, is a continuous function of C. As noted earlier, it is easier to implement when digitized to as few bits as possible. However, it is useful to compare the continuous metric to the quantized metric to see how closely the latter approximates the former.

Define x_0 to mean $\hat{x} = 0$ and x_1 to mean $\hat{x} = 1$, then, using Equation 12

$$I(x_1, C) = \log_2 \frac{p(C \mid x_1)}{p(C)} = \log_2 \frac{p(C \mid x_1)}{p(C \mid x_0)\, p(x_0) + p(C \mid x_1)\, p(x_1)} .$$

For random data and linear parity check codes, it is generally true that $p(x_0) = p(x_1) = 1/2$. So

$$I(x_1, C) = 1 - \log_2 \left[\frac{p(C \mid x_0)}{p(C \mid x_1)} + 1 \right] . \tag{14}$$

Likewise,

$$I(x_0, C) = 1 - \log_2 \left[\frac{p(C \mid x_1)}{p(C \mid x_0)} + 1 \right] . \tag{15}$$

Using the conditional densities in Equations 4 and 5,

$$I(x_1, C) = 1 - \log_2 \left\{ \frac{\mathrm{EXP}\left[- \frac{(C+\mu)^2}{2\sigma^2} \right]}{\mathrm{EXP}\left[- \frac{(C-\mu)^2}{2\sigma^2} \right]} + 1 \right\}$$

$$= 1 - \log_2 \left\{ \mathrm{EXP}\left[- \frac{2\mu C}{\sigma^2} \right] + 1 \right\} . \tag{16}$$

Similarly,

$$I(x_0, C) = 1 - \log_2\left\{\text{EXP}\left[+\frac{2\mu C}{\sigma^2}\right] + 1\right\} \cdot \quad (17)$$

Since $I(x_0, C) = I(x_1, -C)$ it is really necessary only to calculate $I(x_1, C)$. With $\mu = 1$ volt, then $\sigma = (2\,ST_s/N_0)^{-1/2}$ (from Equation 3, and finally

$$I(x_1, C) = 1 - \log_2\left[\text{EXP}\left(-4C\frac{ST_s}{N_0}\right) + 1\right] \cdot \quad (18)$$

Figure 11 is a plot of Equation 18 for ST_s/N_0 values useful for V = 2 and 4 codes.

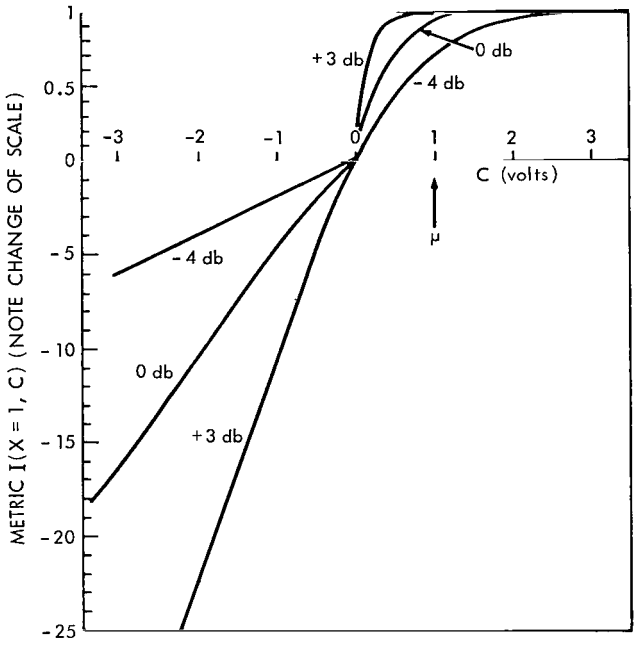

Figure 11—Continuous symbol metric for several ST_s/N_0 ratios.

The Quantized, Gaussian-Channel Metric

As shown in Figure 10, an A to D conversion of a symbol correlation, C, gives rise to a number, Q, representing one of a set of 16 quantiles of C. The quantized symbol LAP metric is defined on Q as follows:

$$M(\hat{x}_{ij} \mid Q_{ij}) = \log_2\left[\frac{P(Q_{ij} \mid \hat{x}_{ij})}{P(Q_{ij})}\right] - \frac{1}{V} \cdot \quad (19)$$

This is similar to Equation 11, but is a ratio of quantile probabilities rather than densities. A result analogous to Equation 14 can be readily obtained:

$$I(x_1, Q) = 1 - \log_2\left[\frac{P(Q \mid x_0)}{P(Q \mid x_1)} + 1\right], \quad (20)$$

where

$$P(Q \mid x_i) = \int_Q \frac{1}{\sigma\sqrt{2\pi}} \text{EXP}\left(-\left[Z + \mu(x_i)\right]^2/2\sigma^2\right) dz$$

and

$$\mu(x_i) = \begin{cases} +1 & \text{if } x_i = x_1 \\ -1 & \text{if } x_i = x_0 \end{cases} .$$

Finally,

$$M(\hat{x} \mid Q) = I(\hat{x}, Q) - \frac{1}{V} . \tag{21}$$

Before calculation can proceed, there is one practical question that must be answered: Over what range of C should the quantization levels be distributed?

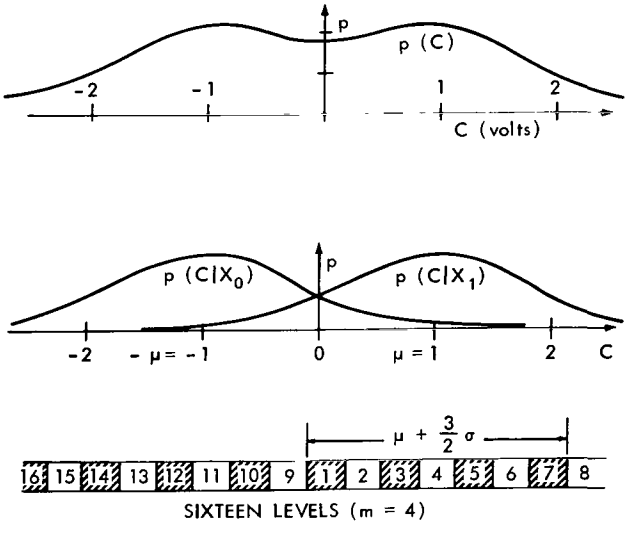

Figure 12—Quantile assignment nomograph.

The optimum level assignment to make is difficult to arrive at, since an exact cost function on the decoder's performance is not known. The assignments selected for this paper were chosen to give a good approximation to the continuous metric over all C where p(C) was non-negligible. It is known from previous simulations that the level assignments are not critical. Figure 12 summarizes the assumptions made. Note that the level assignments are a function of σ and hence of the particular symbol SNR selected. The ordinary A-D converter constraint of equi-spaced quantization intervals is assumed.

Table 1 lists the quantized mutual information, $I(x = 1, Q)$, for four symbol SNR's. Using Equation 21, the LAP metric can easily be found, given the code rate.

The symbol SNR's, ST_s/N_0, in Table 1 have been selected esepcially for rate 1/2 and 1/4 codes. Defining the E_{BIT}/N_0 decoding thresholds as 3 db and 2 db for rate 1/2 and 1/4 codes respectively, we list the corresponding symbol SNR's below:

code rate	Threshold	Threshold +3 db
1/2	0 db	3 db
1/4	-4 db	-1 db

Table 1

Quantized Mutual Information, $I(x = 1, Q)$, Tabulation.*

Quantile Number	$ST_s/N_0 = -4$ db, $\mu = 1$, $\sigma = 1.121$	$ST_s/N_0 = -1$ db, $\mu = 1$, $\sigma = 0.793$	$ST_s/N_0 = 0$ db, $\mu = 1$, $\sigma = 0.707$	$ST_s/N_0 = +3$ db, $\mu = 1$, $\sigma = 0.501$
1	0.201	0.311	0.358	0.539
2	0.510	0.702	0.767	0.925
3	0.712	0.881	0.923	0.989
4	0.835	0.954	0.975	0.998
5	0.908	0.983	0.992	0.999
6	0.948	0.993	0.996	1.000
7	0.972	0.998	0.999	1.000
8	0.989	0.999	1.00	1.000
9	-0.23	-0.40	-0.47	-0.87
10	-0.80	-1.42	-1.75	-3.31
11	-1.47	-2.66	-3.26	-6.06
12	-2.21	-4.00	-4.88	-8.88
13	-3.01	-5.39	-6.54	-11.70
14	-3.84	-6.79	-8.21	-14.53
15	-4.69	-8.21	-9.89	-17.35
16	-6.04	-10.17	-12.12	-20.71

*See Figure 12 for quantile assignments.

Since most operational telemeters normally operate above the threshold of the system, SNR's 3 db above threshold are also given.

Comparison of Continuous and Quantized Metrics

If the quantized metric is superimposed on the continuous metric, as in Figure 13, it is apparent that the quantized metric is a good stepwise approximation to the continuous metric.* Should a quantized metric be desired with different SNR's, etc., one could readily use a graphical approximation to the continuous metric.

Typical Branch LAP Metric Behavior

To get a feel for the behavior of the LAP metric, an example is given here. The branch LAP metric associated with the correct path and best-correlated incorrect path for the rate 1/2 code

*Metric and mutual Information are used interchangeably, since their only difference is the addend $-1/V$.

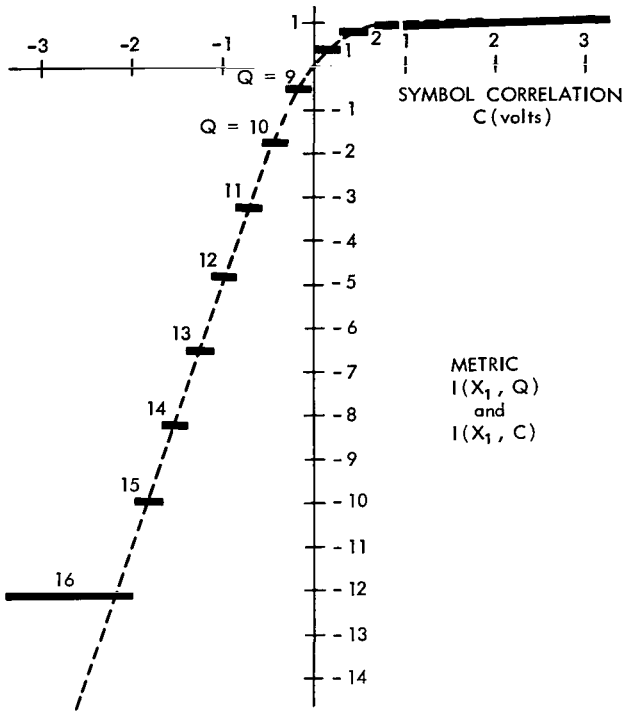

Figure 13—Comparison of the continuous and quantized LAP metrics.

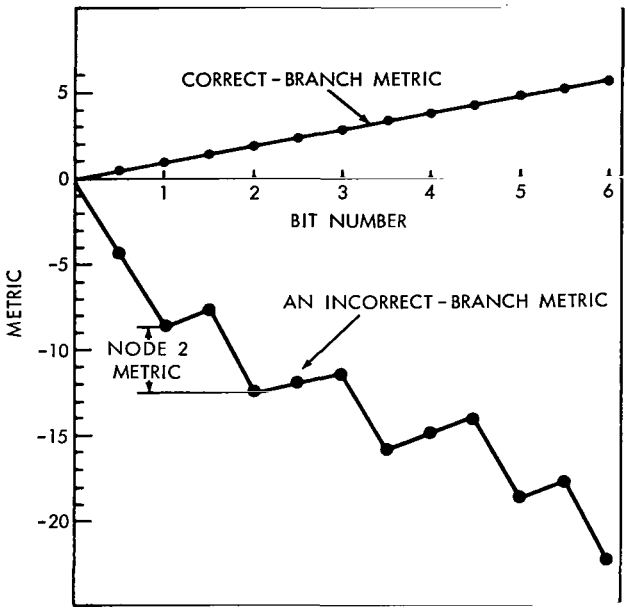

Figure 14—Typical LAP branch metric behavior.

shown in Figure 5 is shown in Figure 14. It is assumed that C "happens" to be very close to $+\mu$ or $-\mu$ so that

$$M_c\left(C \cong +\mu \mid \hat{x} = 1\right) = 0.475 = M_c\left(C \cong -\mu \mid \hat{x} = 0\right) \tag{22}$$

and

$$M_c\left(C \cong -\mu \mid \hat{x} = 1\right) = -4.3 = M_c\left(C \cong +\mu \mid \hat{x} = 0\right). \tag{23}$$

These values include the addend $-1/v = -1/2$. Notice that the sum of two symbol metrics forms a node metric and the sum of the node metrics comprises a branch metric.

Notice further that the correct branch tends to have an increasing metric and incorrect branches tend to have a rapidly decreasing metric. Thus the LAP metric fulfills the metric conditions mentioned earlier, under "The Tree Search Algorithm."

THE CROSS-CORRELATION METRIC

The optimum decoder, as mentioned earlier, under the heading of the same name, requires that the largest correlated tree branch be selected as the best estimate of the transmitted sequence.

It was noted that the branch a posteriori probability was a non-decreasing function of the branch cross-correlation function. Furthermore, since correlation is a linear process, a branch correlation coefficient could be found as the sum of the symbol correlation coefficients, giving the desirable additive property for any selected decoding metric. Thus it seems natural to consider cross-correlation as a candidate for a sequential decoding metric.

For a correlation metric to be compatible with the Fano algorithm, the metric associated with the correct branch must tend to increase, and all incorrect branch metrics must tend to decrease. To accomplish this, a constant number must be subtracted from each correlation coefficient. The linear nature of the correlation process is left undisturbed.

In the following sections, the correlation metric is defined and derived. Simple bounds for the magnitude of the additive constant are found. However, because of the complex decoder statistical behavior, the optimum value must be found by simulation.

Channel Model and Definitions

The same channel mode as employed for the LAP metric and shown in Figure 10 is assumed here. The transforms shown in Figure 10 are all that is changed. However, to distinguish between the LAP metric and the correlation metric, the following definitions are used:

$MC_c(\hat{x} \mid C_{ij})$ = The continuous symbol correlation metric for the symbol hypothesis, $\hat{x}$, given the received symbol correlation, C_{ij}.

$MC(\hat{x} \mid Q_{ij})$ = The quantized symbol correlation metric for the symbol hypothesis, $\hat{x}$, given that the symbol correlation, C_{ij}, was in the Q^{th} quantile.

As before, the analog-to-digital converter assigns the correlation voltage to one of 16 quantiles, or levels. As was done with the LAP metric, the continuous correlation metric will be derived first and then the practical problem of quantizing it will be treated.

Correlation Metric Deviation

If the decoder receives KV symbols of the received signal, $r(t)$, and cross-correlates them with a locally hypothesized signal, $\hat{s}(t)$, then the cross-correlation associated with this KV symbol code branch is:

$$\text{Branch Correlation, } C_B = \int_{t_0}^{t_0+KVT_s} \hat{s}(t)\, r(t)\, dt \tag{24}$$

where

t_0 = some starting time, and

K = the number of information bits shifted into the encoder in the interval under consideration,

V = the reciprocal of the code rate, and

T_s = the duration of a symbol in sec.

Now let $+\mu$ and $-\mu$ be the mean values of the symbol correlations, C_{ij}, when a 1 or 0 is transmitted, respectively. Furthermore, let $\hat{s}(t)$ take on the following values defined as the modulation process:

$$\begin{aligned} &\text{If } \hat{x} = 1, \text{ then } \hat{s}(t) = +1; \text{ and} \\ &\text{If } \hat{x} = 0, \text{ then } \hat{s}(t) = -1. \end{aligned} \tag{25}$$

It follows from Equation 24 that if $\hat{x} = x$ over the entire KV symbol branch, then

$$E(C_B) = KV\mu \tag{26}$$

where $E(C_B)$ denotes mathematical expectation. On the other hand, if one or more data bit errors have been made, then $\hat{x} = x$ perhaps only for half of the KV symbols. In this case, it follows that

$$E(C_B) \cong \frac{KV\mu}{2} - \frac{KV\mu}{2} = 0. \tag{27}$$

To make the branch correlation behavior compatible with the Fano algorithm requirement, a constant will be subtracted from C_B. Let this constant be some fraction of the average branch correlation, $KV\mu$, and define this fraction to be the correlation metric bias, β. Thus

$$\text{Branch Correlation Metric} \triangleq C_B - \beta KV\mu \tag{28}$$

where

C_B = the branch correlation as defined in Equation 24, and

β = the correlation metric bias.

The symbol correlation metric, based upon Equation 28, will now be found and then some bounds on β established.

It follows from Equations 24 and 28 that

$$\text{Branch Correlation Metric} = \int_{t_0}^{t_0+T_s} \hat{s}(t)\, r(t)\, dt + \int_{t_0+T_s}^{t_0+2T_s} \hat{s}(t)\, r(t)\, dt + \cdots$$

$$+ \int_{t_0+(KV-1)T_s}^{t_0+KVT_s} \hat{s}(t)\, r(t) - \beta KV\mu = \hat{s}_1 \int_{t_0}^{t_0+T_s} r(t)\, dt + \cdots + \hat{s}_{KV} \int_{t_0+(KV-1)T_s}^{t_0+KVT_s} r(t) - \beta KV\mu, \tag{29}$$

where $\hat{s}_m$ denotes the estimated symbol associated with the m^{th} received symbol on the KV symbol branch.

Let C_m denote the symbol correlation:

$$C_m \triangleq \int_{t_0+(m-1)T_s}^{t_0+mT_s} r(t)\,dt . \tag{30}$$

Then

$$\text{Branch Correlation Metric} = \left(\hat{s}_1 C_1 - \beta\mu\right) + \left(\hat{s}_2 C_2 - \beta\mu\right) + \cdots + \left(\hat{s}_{KV} C_{KV} - \beta\mu\right) . \tag{31}$$

Now, changing to the double subscript notation where "i" denotes the information bit number and "j" the associated symbol number, Equation 31 becomes

$$\text{Branch Correlation Metric} = \sum_{i=1}^{K}\sum_{j=1}^{V} \underbrace{\left(\hat{s}_{ij} C_{ij} - \beta\mu\right)}_{\text{Symbol Correlation Metric}} . \tag{32}$$

Node Correlation Metric

Since $\hat{s}_{ij}$ is constrained to $\hat{x}$ by Equation 25, it follows that the continuous symbol correlation metric is

$$MC_c\left(\hat{x} \mid C_{ij}\right) = \begin{cases} C_{ij} - \beta\mu & \text{if} \quad \hat{x} = 1 \\ -C_{ij} - \beta\mu & \text{if} \quad \hat{x} = 0 \end{cases} . \tag{33}$$

Thus Equation 32 may be rewritten as

$$\text{Branch Correlation Metric} = \sum_{i=1}^{K}\sum_{j=1}^{V} MC_c\left(\hat{x} \mid C_{ij}\right) . \tag{34}$$

Bounds on the Metric Bias

One obvious upper bound for the bias, β, is unity. In this case, the average metric for the *correct* branch is zero. However, the Fano algorithm requires that the correct branch tend to be increasing. This result follows directly from Equations 33 and 34.

$$\begin{aligned} \text{Average Branch Correlation Metric (correct branch)} &= E\left[\sum_{KV}\sum MC_c\left(\hat{x} \mid C_{ij}\right)\right] \\ &= KV\left[E(C) - \beta\mu\right] \text{when } \hat{x} = x \\ &= KV\mu(1-\beta) \\ &= 0 \text{ when } \beta = 1 . \end{aligned}$$

A simple lower bound for β follows from the fact that the Fano algorithm requires that all incorrect paths tend to decrease, and from the fact that the average branch cross-correlation for a convolutional code is no lower than zero. Thus:

$$\text{Average Branch Correlation Metric (incorrect branch)} = E\left[\sum_{KV}\sum MC_c\left(\hat{x} \mid C_{ij}\right)\right]$$

$$= \frac{KV}{2}\left[-E(C) - \beta\mu\right] + \frac{KV}{2}\left[E(C) - \beta\mu\right]$$

for $\hat{x} \neq x$ for KV/2 of the symbols, and

$$\text{Average Branch Metric} = -KV\beta\mu ,$$

and this is ≥ 0 for $\beta \leq 0$. Therefore, some strict, but wide, bounds for β are: $0 < \beta < 1$. For a rate 1/2 code, the correlation of incorrect branches is considerably above zero and so we might expect reasonable values of β to be near 1.

The Quantized Correlation Metric

For practical reasons, noted previously, it is necessary to quantize the symbol metric. For a good simulation comparison, the quantile assignments have been taken identical to those made for the LAP metric as shown in Figure 12. Using a stepwise linear approximation to the continuous metric in Equation 33, we get the plot in Figure 15. A mean value, μ, of one volt is assumed, and the bias shown as 0.8. The metric for the alternate hypothesis, $\hat{x} = 0$, is simply the mirror image of that for $\hat{x} = 1$. That is,

$$M(\hat{x} = 0 \mid C) = M(\hat{x} = 1 \mid -C) .$$

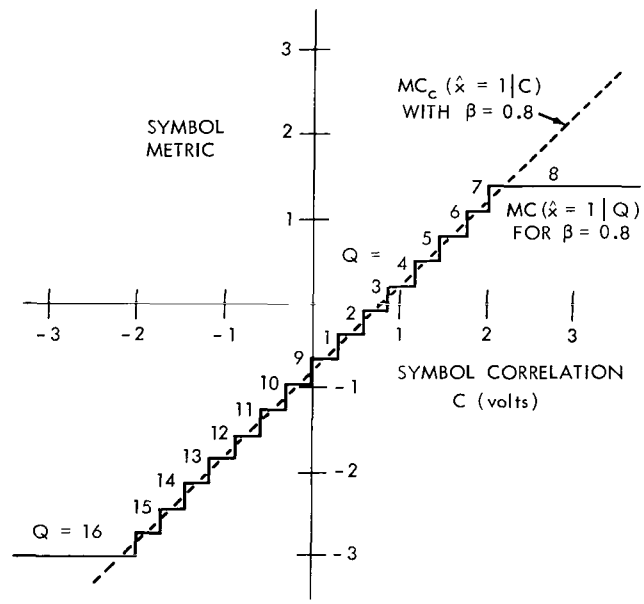

Figure 15—Continuous and quantized correlation metrics.

THE RATE 1/2 CODE SIMULATIONS

Because of the statistical complexity of the sequential decoder, it is necessary to resort to simulation techniques to compare accurately the relative merits of decoding metrics. But, with any simulation of a complex system, there will be a number of parameters for which

optimum values are not known. It is then necessary to select parameters which are "reasonable" from a practical standpoint or have been shown to be "good" from previous work. This is the case for the encoder generator matrix and code rate selected. However, inadequate information was available regarding the Fano algorithm threshold spacing. So it was necessary and feasible to search for a "good" spacing. In addition, it was necessary to find a "good" correlation metric bias.

A rate 1/2 convolutional code, as opposed to a lower rate code, was selected for the simulations for a number of reasons:

1. Current interest in the field centers on the use of a rate 1/2 code because of its modest bandwidth requirements and substantial coding gain,

2. The disadvantages of a non-optimum decoder metric should be most pronounced with a high-rate code, and

3. The computer execution time is lower.

Before the simulation results are presented, the basic communications system model is described and a synopsis of parameters given. In addition, criteria for "goodness" of performance are defined. The choice of "good" decoder parameters is then based on these criteria.

The Simulation Model

The system model used for the simulations is shown in Figure 16. A constraint-length 32, rate 1/2 convolutional encoder accepts random binary data and feeds coded symbols to the channel modulator. The computer simulation actually uses a pseudorandom binary data source. A frame consists of 224 information bits and 479 binary symbols.

An idealized additive, white, gaussian noise channel is assumed, and the symbol matched filter has perfect synchronization. Thus the symbol correlation voltage, c_{ij}, will have a gaussian density with a mean of +1 volt when $x_{ij} = 1$ and a mean of -1 volt when $x_{ij} = 0$. It is a simple matter for the computer to simulate the channel/matched-filter combination by simply directly synthesizing the c_{ij}. The additive gaussian noise samples are generated by adding 12 samples from a uniform number generator and appropriately normalizing. The latter generator employs Hutchinson's method (Reference 15). The gaussian samples have been extensively tested for independence and normality out to 3.5 standard deviations.

The quantization of the symbol correlations follows exactly the model given for the LAP metric earlier, under "The Quantized, Gaussian-Channel Metric."

The simplified model of the sequential decoder in Figure 16 is shown with somewhat more memory than actually required for a frame. The decoder starts once the Q memory is filled with

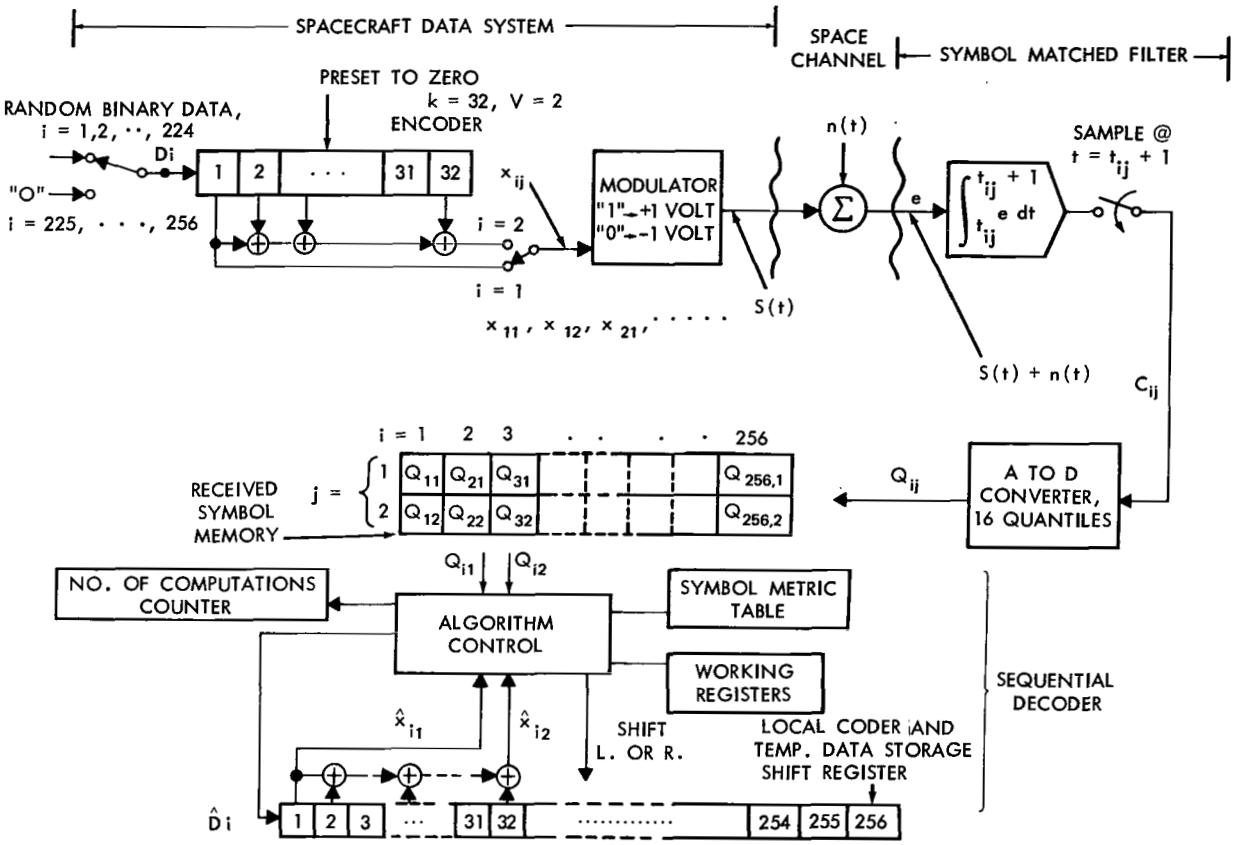

Figure 16—Rate 1/2 coded data system model.

a frame, and stops computation when either bit 224 is decoded with a metric greater than current threshold or the number of computations exceeds an overflow limit. Once a frame is decoded, the data, $\hat{D}_i$, are searched for errors. The number of bit errors and the number of computations required for each frame are recorded for performance evaluation.

Synopsis of System Parameters

Encoder:

Type: Convolutional, systematic, rate 1/2, constraint-length = 32

$$G = \begin{bmatrix} 1000 & 0000 & 0000 & 0000 & 0000 & 0000 & 0000 & 0000 \\ 1101 & 0101 & 1001 & 0111 & 0001 & 1101 & 1110 & 1101 \end{bmatrix}$$

Initial State: all zeros

PCM Format:

Number of information bits per frame = 224

Number of symbols per frame = 479

Tail: (binary zero encoder input, parity only out) = 31

Channel and Matched Filter:

System SNR Parameter: E_B/N_0 = energy per bit/noise power density averaged over a frame

Symbol Correlation Mean: $+1V$ if $x = 1, -1V$ if $x = 0$

Symbol Correlation Standard Deviation: $\sigma_c = \left(\frac{E_B}{N_0} \frac{(2)(224)}{479}\right)^{-1/2}$

Synchronization and Filtering Losses: None

Decoder:

Algorithm: Modified Fano Algorithm (see Appendix B)

Symbol Metrics:

1. Log-a-Posteriori Probability Metric at 0 db ST_s/N_0 (see the earlier section dealing with this type of metric)
2. Biased Correlation Metric, $\beta = 0.9$ (see the preceding section, "The Cross-Correlation Metric")

Threshold Spacing (IDELTA):

1. With LAP Metric: = 4 × max. node metric
2. With Correlation Metric: = 1 × max. node metric

Number of Symbol Correlation Quantiles: 16 (see earlier heading, "The Quantized, Gaussian-Channel Metric")

Tail Symbol Treatment: The node metric pair for the last frame information bit is calculated using that node and the entire tail. The local bit estimates, $\hat{D}_i$, are forced to be zero for calculating the tail metric.

"Goodness" of Performance Criteria

Most of the system parameters have been defined and fixed. Only the decoder threshold spacing and correlation metric bias parameters are unspecified. These parameters are to be selected to give "good" performance, which here means minimizing the average computations per bit and the undetected error probability. It is not generally true, however, that both can be simultaneously minimized.

Furthermore, since exact cost functions are not known for the error probability or for the average computational load, optimum parameter selection cannot be made. It is usually true that undetected errors are serious and that it is acceptable to increase the computational load somewhat if the error probability can be substantially reduced.

Thus the simple rule used for "good" parameter selection was to select those parameters which tend to minimize the average computations per bit while giving an acceptable error probability.

The LAP Metric Simulation Results

Using the log-a-posteriori symbol metric in the sequential decoder, a threshold spacing search was first made at an E_B/N_0 of 3.0 decibels. Each data point is based on a run of 250 frames. Each run used the same "noise" and data derived from the pseudorandom number generator. Hence relative performance should be accurate. These results are shown in Figure 17, which plots the average computational load, $\bar{C}_B$, versus a normalized spacing. A broad minimum exists from $\Delta_\eta = 3$ to $\Delta_\eta = 9$.

Since no bit errors occurred in any of these runs, a $\Delta_\eta = 4$ was selected for these simulations, primarily on the basis of minimizing $\bar{C}_B$.

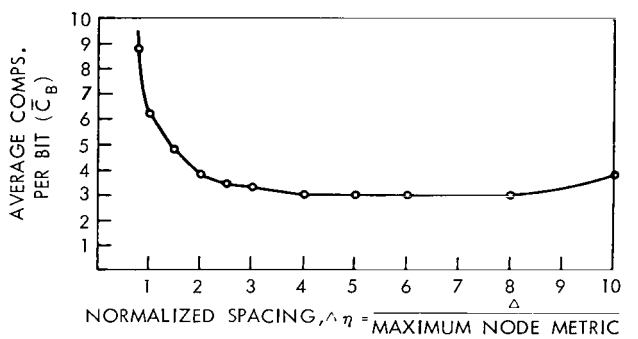

Figure 17—Decoder computational load versus threshold spacing.

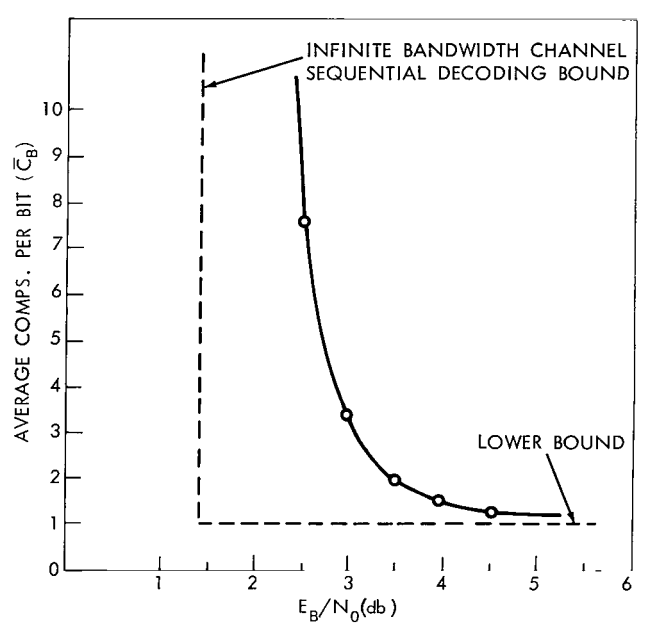

Figure 18—Average computational load versus E_B/N_0.

Computer runs of 500 frames each were then made to determine decoding statistics versus E_B/N_0. A number of additional runs with different pseudorandom data and noise were made to ascertain the repeatability of the statistics and to check some anomalous behavior.

Figure 18 plots the average computations per bit verus E_B/N_0. For comparison, some basic sequential decoding bounds are also indicated. The decoder must, of course, make at least one computation for each received bit, giving the $\bar{C}_B$ lower bound. It is well known that $\bar{C}_B$ must approach infinity when the system is approaching operation at one-half channel capacity (Reference 16). This corresponds to an E_B/N_0 of 1.4 decibels for the "very noisy," continuous channel.

A better indicator of a sequential decoder's performance is its so-called buffer overflow probability. It is here defined as the probability that the number of decoder computations, C, required to decode one frame of data exceeds some limit, L. It is desired to find this probability as a function of signal-to-noise ratio. That is,

$$P(C > L) = \text{func}(L, E_B/N_0) .$$

It has been shown that this distribution is of the Pareto type (References 17 and 18). That is,

$$P(C > L) \sim L^{-\alpha}$$

where α is the Pareto exponent. These simulation results tend to bear this out. Figure 19 plots the relative frequency of overflow. The ordinate is labeled with $\hat{P}(C > L)$ to emphasize that it is an estimated probability.

To help determine the integrity of the results, additional 500 frame runs were made at most of the E_B/N_0 ratios treated. This accounts for the double set of points associated with the curves. It can be seen that, over the range of the curves plotted, the repeatability was such that E_B/N_0 differences of 0.2 db. or better could be resolved.

The peculiar break in the 4 decibel curve cannot be explained. Subsequent results indicate that it is dependent on the Δ_n selected.

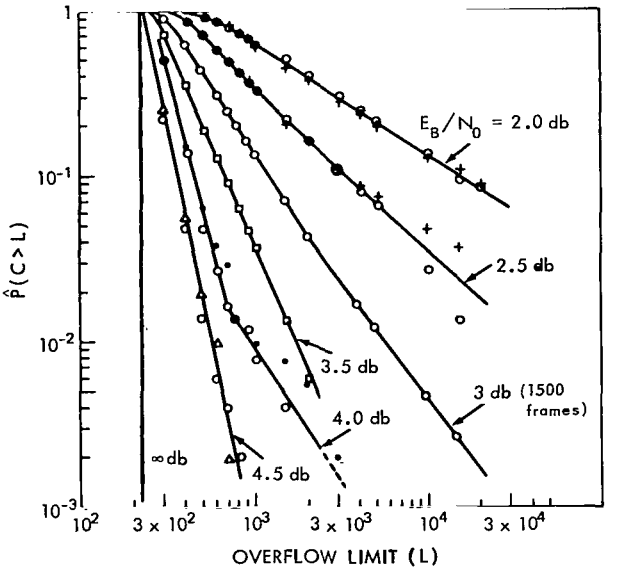

Figure 19—Decoder overflow distributions with the LAP metric.

At 3.0 decibels, 1500 frames were run to extend the curve meaningfully below $\hat{P}(C > L) = .01$. The results indicate another break in the curve giving a lower Pareto exponent for large L. This tends to indicate that all the curves may break at higher L.

By definition, the communications system threshold is that E_B/N_0 ratio for which the Pareto exponent, α, is equal to unity in the $\hat{P}(C > L)$ distributions. For these simulations, system threshold occurs at about 2.4 decibels.

So few bit errors occurred in these runs that it is difficult to make a good point estimate of the error probability. Table 2 summarizes the error results. These results will be compared with several theoretical bounds. The bounds assume an $\alpha \geq 1$, an "average" rate 1/2 convolutional

Table 2

Error Summary for LAP Metric.

E_B/N_0 (db)	No. of Frames Run	No. of Frames with ≥ 1 Error	Total No. of Bit Errors	$P_B(\epsilon)$ Point Estimate (224 bits/frame)
2.0	1000	3	19	8.5×10^{-5}
2.5	1500	3	8	2.4×10^{-5}
3.0	1500	2	10	3.0×10^{-5}
3.5	500	0	0	$\leq 1 \times 10^{-5}$
4.0	1000	1	6	2.7×10^{-5}
4.5	1000	1	6	2.7×10^{-5}

code, and a LAP metric decoder with a large overflow limit. The frame error probability is:

$$P_F(\epsilon) \leq n_d\, 2^{-k/2}$$

where:

n_d = the number of data bits per frame of data,

k = the constraint length of a systematic rate 1/2 code,

and for this case:

$$P_F(\epsilon) \leq 224 \times 2^{-16} \cong 3.4 \times 10^{-3} .$$

At 2.5 db. E_B/N_0,

$$\hat{P}_F(\epsilon) = 3/1500 = 2.0 \times 10^{-3} .$$

The bit error probability bound is:

$$P_B(\epsilon) \leq 6 \times 2^{-16} \cong 9.1 \times 10^{-5}$$

and the actual bit error probability (at 2.5 db.) was:

$$\hat{P}_B(\epsilon) = 2.4 \times 10^{-5}$$

Considering the limited number of samples involved and the possibility of poor encoder connections, the agreement between the theoretical bound and these results is considered excellent.

The Correlation Metric Simulation Results

With the normalized threshold spacing, Δ_η, held at a trial value of 1.5 and an E_B/N_0 of 4.0 decibels, a search for a good correlation metric bias, β, was made. The results are plotted in Figure 20. As with the LAP metric parameter

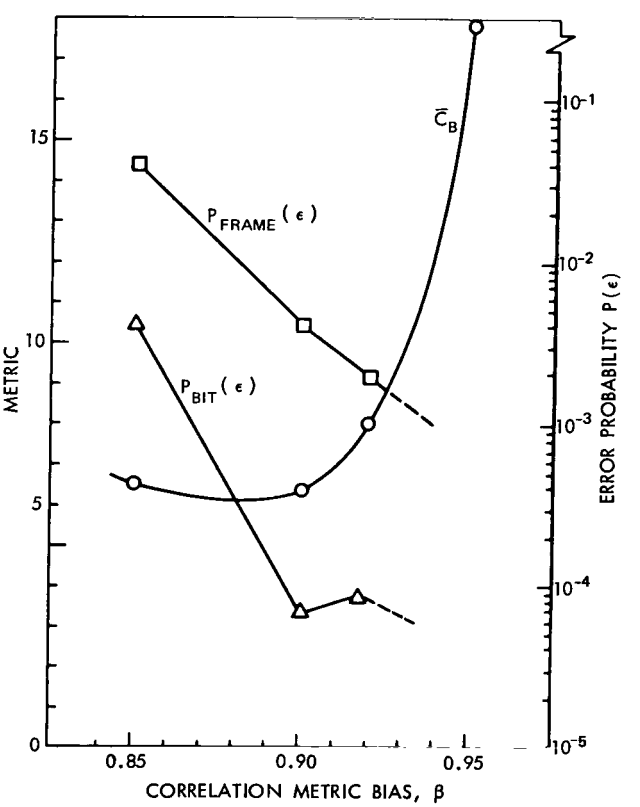

Figure 20—Correlation metric bias search results.

search, each data point is based on a run of 250 frames, with each run using the same pseudo-random noise and data sequences. The average number of computations per decoded bit, $\bar{C}_B$, rises sharply with $\beta > 0.9$, while the bit error probability, $P_B(\epsilon)$, tends to level off. However, the decreasing frame error probability tends to indicate that the error probability is actually a continuously decreasing function of β. For these simulations, a $\beta = 0.90$ was selected.

A search was next made for a good threshold spacing, Δ_η. It is desirable to optimize β and Δ_η simultaneously, but it is not clear that they may be independently optimized. The Δ_η finally selected differs little from the $\Delta_\eta = 1.5$ employed in the β search. Thus no further searching appeared warranted. Figure 21 plots the run-averaged decoder computations per bit, $\bar{C}_B$, and the frame error probability versus normalized threshold spacing for a 4.0 decibel E_B/N_0 ratio. In this case, both the error probability and computational load are minimized at $\Delta_\eta = 1$. Since additional data were later available for $\Delta_\eta = 1$, the error probability shown is for a 1000-frame run.

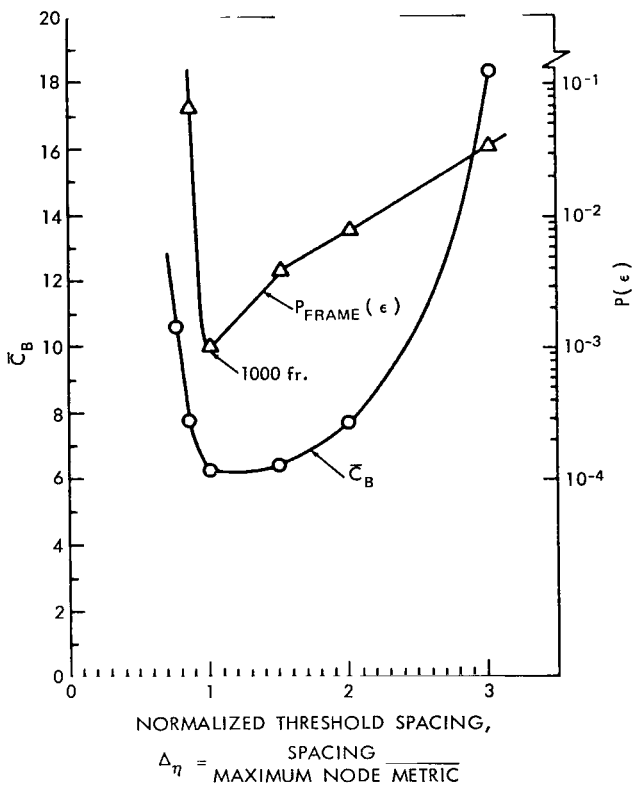

Figure 21—Decoder spacing search for the correlation metric.

With parameters $\beta = 0.9$ and $\Delta_\eta = 1$ selected, runs were made at various E_B/N_0 ratios to determine the $\bar{C}_B$, $P(C > L)$, and $P(\epsilon)$ statistics. It was found that the variances of $\hat{P}(C > L)$ point estimates were somewhat greater than for the equivalent LAP metric statistics for 500-frame runs. Thus all statistics were based on at least 1000-frame runs.

Figure 22 plots $\bar{C}_B$ versus E_B/N_0, and shows a rapid increase in $\bar{C}_B$ for E_B/N_0 under 4.5 decibels. As before, the basic sequential decoding lower bounds are shown as dashed lines.

The corresponding overflow distributions are shown in Figure 23. To test the repeatability of the results, a second 1000-frame run was made at 4.5 db. The repeatability appears sufficient to resolve the curves to within a 0.2 db E_B/N_0. A definite curvature is observable in the 5.0 and 6.0 db distributions, suggesting an undesirably low Pareto exponent for large L.

As with the LAP metric simulations, so few undetected bit errors occurred in these runs that it is difficult to make a good point estimate of the error probabilities. Table 3 summarizes the error data for the correlation metric simulation runs. Point estimates of the frame and bit error probabilities are shown, and serve as reasonable "order of magnitude" estimates.

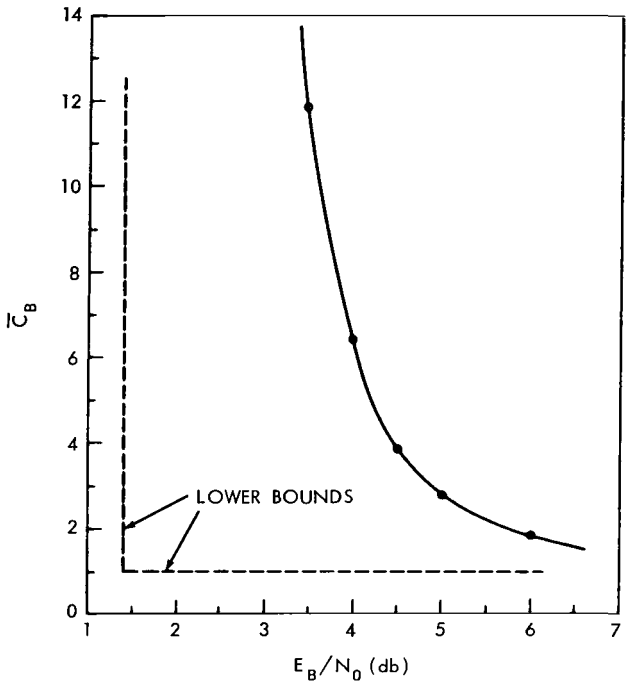

Figure 22—Average computational load versus E_B/N_0 for the correlation metric.

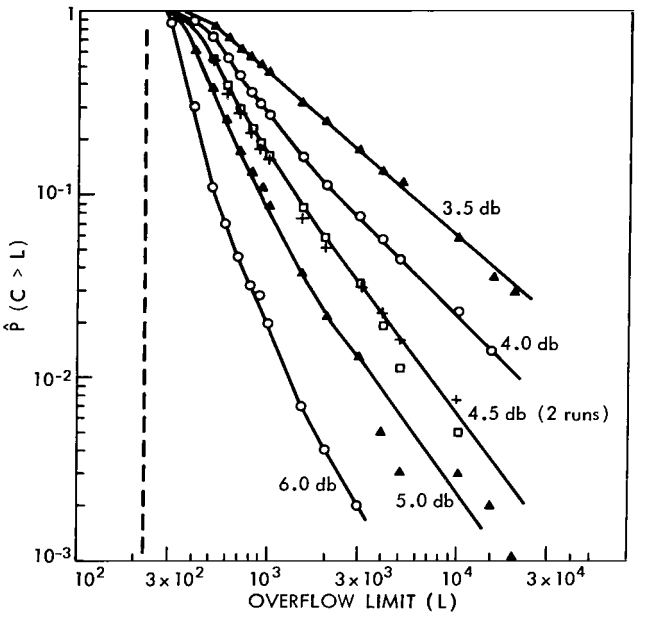

Figure 23—Decoder overflow distributions with the correlation metric.

Table 3

Error Summary for Correlation Metric.

E_B/N_0 (db)	No. of Frames Run	No. of Frames with $\geq$ 1 Error	Total No. of Bit Errors	$P_B(\epsilon)$ Point Estimate
3.5	1000	10	89	4×10^{-4}
4.0	1000	1	17	7.6×10^{-5}
4.5	2000	0	0	$<2.3 \times 10^{-6}$
5.0	1000	0	0	$<5 \times 10^{-6}$
6.0	1000	0	0	$<5 \times 10^{-6}$

PERFORMANCE COMPARISON OF DECODER METRICS

The purpose of this chapter is to demonstrate quantitatively, using the simulation results, the sensitivity of a sequential decoder to the choice of a metric. This is done by comparing appropriate decoder performance parameters when using the LAP metric with a decoder using a cross-correlation metric. A correlation metric gives optimum results in an ideal decoder, a fact which motivates the choice of this metric. Also, the performance of the sequential decoder is of interest in itself, since little experimental data have been published to date. The LAP metric and

correlation metric decoders are compared on the basis of their computational and error-rate behavior at signal to noise ratios near threshold. In addition, the relative sensitivity of the two metrics to variations in the input signal amplitude is noted. Other study areas of interest related to sequential decoding are also mentioned.

Comparison by Decoder Computational Behavior

For some data users, deletion of data caused by decoder overflow is just as serious as bit errors. To others, occasional data deletions are acceptable provided the error rate in the remaining data is very small. Indeed, the error probability can be made very small with convolutional coding by choosing the constraint-length large. The 32-bit constraint-length encoder in these simulations gave a $P_B(\epsilon) \cong 3 \times 10^{-5}$. If it was increased to 52 bits, then a $P_B(\epsilon) \cong 3 \times 10^{-8}$ could be expected. Thus, from a data user's standpoint, the overflow or deletion rate is the main limitation on system performance.

The overflow rate is dependent upon the decoder overflow limit selected. This in turn depends upon the speed advantage of the decoder relative to the input bit rate, buffer size, etc. Overflow comparisons will be made using a normalized overflow limit, L_n. Define

$$L_n = \frac{L}{n_d} = \frac{\text{Actual Decoder Overflow Limit}}{\text{The Number of Bits per Frame}} .$$

This limit, L_n, may be interpreted as the decoder speed advantage relative to the input bit-rate.

Now, by using the overflow distributions of the preceding section, "The Rate 1/2 Code Simulation," and using L_n = 3, 10, 30, and 100, the plots in Figures 24 and 25 can be found. The ordinate labeled $\hat{P}$ (overflow) is identical to $\hat{P}(C > L)$ previously plotted, with the caret signifying experimental point-estimated probabilities.

A number of observations may be made. First, for any specified L_n and overflow probability, the LAP metric decoder can tolerate a 1.5 decibel to 2.5 decibel lower E_B/N_0 ratio than can the correlation metric decoder. In addition, for overflow probabilities under 10^{-3}, the trend of the results indicates the correlation metric decoder may be inferior to the LAP metric decoder by more than 3 decibels.

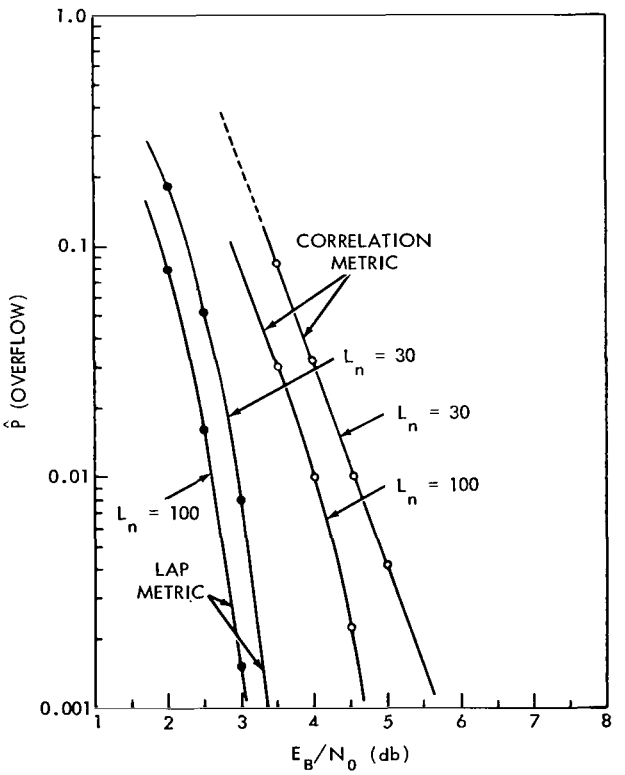

Figure 24—Decoder overflow probability versus E_B/N_0 for moderate overflow limits.

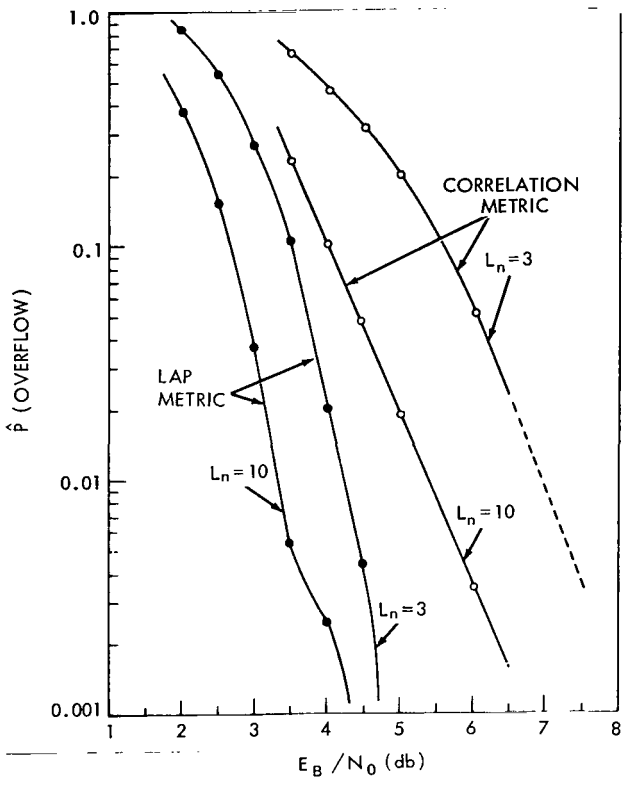

Figure 25—Decoder overflow probability versus E_B/N_0 for large overflow limits.

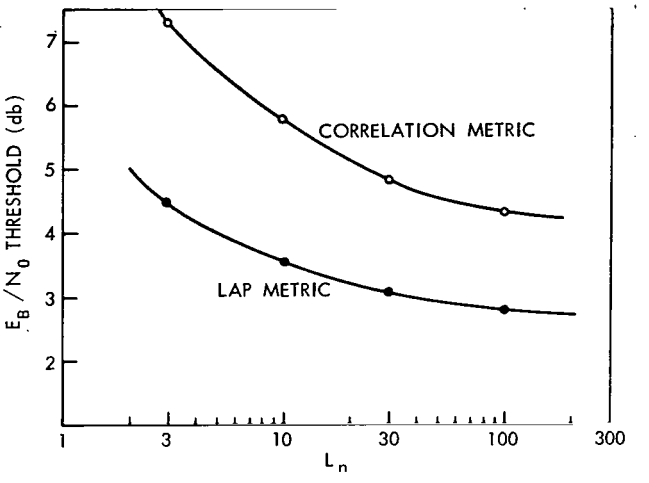

Figure 26—E_B/N_0 threshold for 0.5% deletions versus normalized overflow limit.

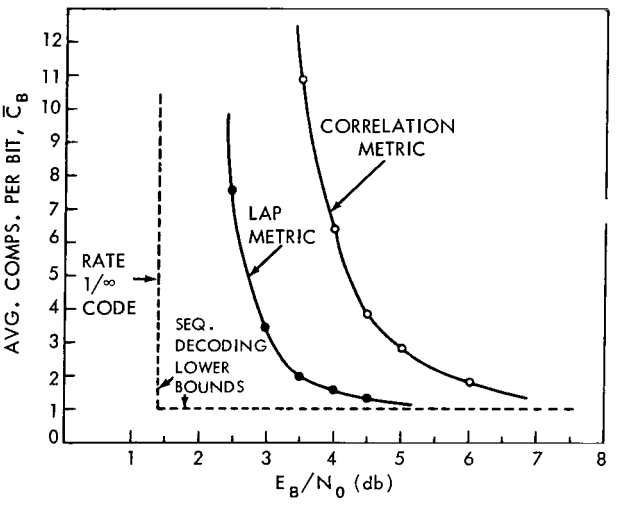

Figure 27—Average computational load versus E_B/N_0.

Another observation to be made for either metric decoder is the diminishing improvement to be gained by choosing L_n larger and larger. For example, if a probability of overflow, and hence deletion rate, of 0.5 percent was the maximum tolerable rate for some data user, then the corresponding E_B/N_0 "threshold" could be found as a function of L_n from Figures 24 and 25. Results for this case are plotted in Figure 26. It is clear from this plot that selecting L_n larger than 100 gains little E_B/N_0 improvement. For this 0.5 percent deletion rate, the correlation metric is at least 1.6 decibels poorer.

One final decoder computational comparison of interest is the average computations per bit, $\bar{C}_B$, versus E_B/N_0 for each metric. These data were based on an L_n of 134 in the simulations but differ little from an $L_n = \infty$ for $\bar{C}_B$ under about 4. These results, plotted in Figure 27, again show the correlation metric inferior by about 2 decibels.

Note further from Figure 27 that the E_B/N_0 advantage to be had by using very low rate convolutional codes is under 1.5 decibels. From Shannon's capacity bound (Reference 1) and associated

value of E_B/N_0 , and from the preceding section, "The Rate 1/2 Code Simulation," it can be established that for the LAP metric a rate 1/2 coded data system gives good performance to within 1/3 of channel capacity on the gaussian channel.

Comparison by Decoder Error Probability

As noted earlier, so few bit errors occurred in the simulations that good point estimates of error probability could not be made. In addition, from a theoretical and practical viewpoint the error probability can be readily reduced by increasing the code's constraint-length. Thus the main purpose of this section is to compare the LAP and correlation metrics' relative error probability performance, although both could be made acceptably small.

The error results in the preceding section, "The Rate 1/2 Code Simulation," are based upon an overflow limit, L_n, of 134. It was observed that a substantial fraction of the incorrectly decoded frames also required many more computations than the typical correctly decoded frame. Therefore, reducing the overflow limit from 134 will also reduce the error probabilities observed (while increasing deletion rate). None the less, the relative error-rate performance should be the same for any L_n.

Table 4 compares the bit error probability estimates for the simulations.

Table 4

Comparison of LAP and Correlation Metric Bit Error Probability Results.

E_B/N_0 (db)	LAP		Correlation	
	No. Frames in Error	$\hat{P}_B(\epsilon)$	$\hat{P}_B(\epsilon)$	No. Frames in Error
2.0	3	8.5×10^{-5}	-	-
2.5	3	2.4×10^{-5}	-	-
3.0	2	3.0×10^{-5}	-	-
3.5	0	$<1.0 \times 10^{-5}$	4.0×10^{-4}	10
4.0	1	2.7×10^{-5}	7.6×10^{-5}	1
4.5	1	2.7×10^{-5}	$<2.3 \times 10^{-6}$	0
5.0	-	-	$<5.0 \times 10^{-6}$	0
6.0	-	-	$<5.0 \times 10^{-6}$	0

From this table, several observations can be made. First, the correlation metric again gives inferior performance relative to the LAP metric. Second, at 4.0 decibels and higher, both metrics gave similar error rates, considering that only 1 or no frame errors occurred.

If the bit error probabilities are compared at E_B/N_0 ratios that give about the same computational load, then the correlation metric still gives higher error rates. For example, at 3.5 db, $\bar{C}_B \cong 11$ for the correlation metric; at 2.0 db., $\bar{C}_B > 11$ for the LAP metric. Yet the bit error rate is nearly five times higher for the correlation metric.

It appears that the correlation metric decoder does not discard incorrect tree paths as fast as the LAP metric. It therefore tends to search farther into the tree on incorrect paths. This correspondingly increases the chances of making errors by clearing out all history of past errors in the local encoder's shift register.

Sensitivity of Decoder Metrics to Signal Amplitude Fluctuations

Although the simulation model assumed that the signal amplitude at the receiver was held constant, any practical system must consider the consequences of amplitude fluctuations. Time has not permitted detailed investigation of these effects and related problems, but with the aid of past results and two additional computer runs, the relative sensitivity of the LAP and correlation metrics to amplitude fluctuations can be found.

One type of amplitude fluctuation typical of an AGC receiver is considered: Let the symbol matched-filter output due to signal alone, namely $+\mu$ or $-\mu$, be a random variable in time. Let the magnitude, μ, have a probability density function: $P(\mu)$ = Normal (mean = 1, σ_μ = 0.06). Furthermore, let μ be slowly changing. More precisely, if $\phi(\tau)$ is the autocorrelation function of μ, then let

$$0 < \phi(\tau) \leq 1 \qquad \text{for} \qquad \tau < 150\, T_s$$

and

$$\phi(\tau) = 0 \qquad \text{for} \qquad \tau \geq 150\, T_s ,$$

where T_s is the duration of a transmitted symbol. Thus, over a duration of several constraint-lengths, μ may be considered essentially constant.

The above model will cause typical amplitude deviations of about 0.5 decibels from the mean (~6 percent voltage change) and is representative of a good AGC system working with a noisy signal.

A signal amplitude increase or decrease over a period of many bits is equivalent to a decrease or increase, respectively, of the correlation metric bias, β. It follows from Equation 33 that, if the received signal is changed by a factor of $(1 + \epsilon)$, the new equivalent bias, β', is:

$$\beta' = \beta - \epsilon .$$

However, the correlation metric decoder's performance as a function of β has already been shown in Figure 20 for an E_B/N_0 of 4.0 db. It follows that over a span of bits in which the signal amplitude

is low by 6 percent, the probability of error will increase by about two orders of magnitude. In addition, when the signal amplitude is high by 6 percent, the average computations per bit more than triple.

It follows that the correlation metric is unsatisfactory for a practical system.

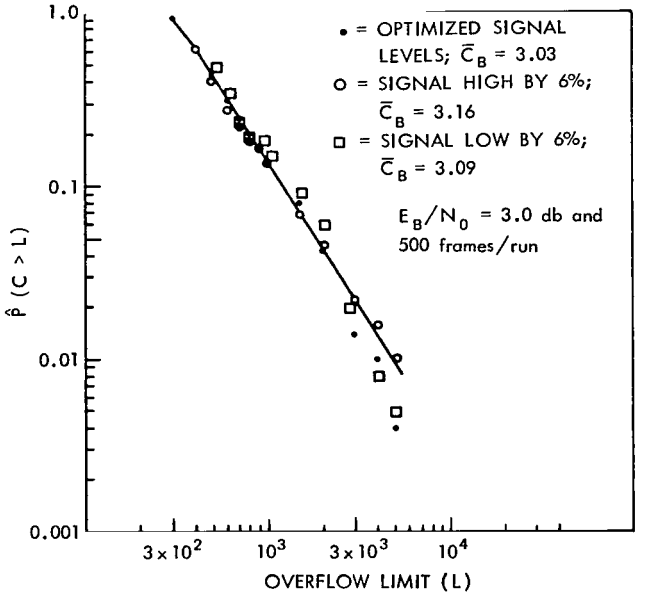

Figure 28—LAP metric signal amplitude sensitivity tests.

To help determine the sensitivity of the LAP metric to slow amplitude fluctuations, two additional computer runs, of 500 frames each, were made. In one run, the received-signal-plus-noise voltage was made high by 6 percent relative to the LAP-metric optimized values (shown previously in Figure 12). Similarly, the other run was made with the signal low by 6 percent. The overflow distributions of these two runs are compared with a previous run with optimum parameters in Figure 28. In all three runs the *same* pseudorandom noise and data sequences were used, so their *relative* performance should be accurate. All three runs gave virtually the same overflow distributions, although the average computational load did increase moderately for the two mismatched amplitude runs. Apparently no deterioration of the bit error probability occurred. No errors occurred in the two mismatched amplitude runs, while four bit errors did occur in one frame of the optimized signal level run. These are inadequate data, however, to indicate any trend. None the less, it is clear that the LAP metric is far less sensitive to signal amplitude fluctuations than is the correlation metric. Furthermore, for slow amplitude fluctuations of 6 percent or less, the performance degradation is negligible.

Additional Studies

Obviously, a more complete signal-amplitude sensitivity, error probability, and overflow specification is desirable for a practical system. It would also be useful and interesting to investigate more thoroughly the effect of Δ_η on the break in the overflow distributions. The LAP metric in these simulations was optimized at an E_B/N_0 of 3.0 db. Investigating the effect on the Pareto exponent of continuously matching the metric to the channel SNR would be useful.

The slowly changing signal amplitude treated above implies burst-type noise. A sequential decoder is especially sensitive to (large) burst noise. One way of combatting this problem is to use extensive symbol scrambling-descrambling to "spread out" the bursts. However, as noted by Jacobs (Reference 18) and others, it may be more efficient and effective to combine algebraic burst-error-correlation coding with sequential decoding to handle this problem. Investigation of these techniques promises not only diminished burst-noise sensitivity but also communications closer to channel capacity.

CONCLUSIONS

The primary purpose of this paper has been to determine the sensitivity of a sequential decoder's performance to the choice of a metric. The computer simulations of a rate 1/2 code have shown the log-a-posteriori probability (LAP) metric to be superior to a correlation metric in every way. In particular:

1. For a specified overflow probability, the correlation meter decoder required a 1.5 to 2.5 decibel higher signal-to-noise ratio than the LAP metric decoder;

2. For a given computational load (near threshold), the correlation metric decoder had an error rate nearly 5 times as high; and

3. When subjected to a slowly varying signal amplitude ($\pm$ 0.5 db), the correlation metric decoder gave an unacceptably high error rate or computational load; whereas the LAP metric decoder was negligibly degraded.

Although the optimum decoder functions equally well with a probability or correlation decision metric, it is clear that a sequential decoder's performance is quite sensitive to the choice of its metric.

Another purpose of this paper has been the determination of good system parameters and the resulting system performance. These results are contained in the last two sections, "The Rate 1/2 Code Simulations" and "Peformance Comparison of Decoder Metrics." Some important conclusions made for the LAP metric system are:

1. A rate 1/2 coded data system gives good performance to within 1/3 of channel capacity on the gaussian channel,

2. The decoder overflow probability is the primary system signal-to-noise ratio limitation, and

3. Choosing a decoder overflow limit greater than 100 times the number of bits per frame gains a negligible reduction in signal-to-noise ratio threshold.

ACKNOWLEDGMENT

The author wishes to express his gratutide to Mr. G. Badagliacca for his extensive computer programming assistance, without which the large simulation program and detailed investigations would still be incomplete.

Goddard Space Flight Center
National Aeronautics and Space Administration
Greenbelt, Maryland, June 28, 1968
125-23-02-96-51

REFERENCES

1. Shannon, C. E., and Weaver, W., *Mathematical Theory of Communication,* Urbana, Ill.: University of Illinois Press, 1949.

2. Viterbi, A. J., "On Coded Phase-Coherent Communications," IRE Transactions on Space Electronics and Telemetry, SET-7, Mar. 1961.

3. Rochelle, R. W., "Pulse Frequency Modulation Telemetry," NASA Technical Report R-189, Jan. 1964.

4. Jaffe, R. M., "Digilock Telemetry System for The Air Force Special Weapons Center's Blue Scout Jr.," IRE PG-SET, SET-8 (1), Mar. 1962.

5. Saliga, T. V., "An 8-Bit Bi-orthogonal Telemeter with Flexible and Efficient Synchronization," Proceedings of 1967 National Telemetering Conference, San Francisco, Calif., May 1967.

6. Wozencraft, J. M., and Reiffen, B., *Sequential Decoding,* Cambridge, Mass.: M.I.T. Press, 1961.

7. Fano, R. M., "A Heuristic Discussion of Probabilistic Decoding," *Transactions IEEE,* PTGIT IT-9, 64, 1963.

8. Blustein, G., and Jordan, Jr., K. L., "An Investigation of the Fano Sequential Decoding Algorithm by Computer Simulation," Group Report 62G-5, Lincoln Laboratory, M.I.T., DDC 412632, H-525, July 12, 1963.

9. Jacobs, I., "Probabilities of Overflow and Error in Coded Phase-Shift-Keyed Systems with Sequential Decoding," JPL Space Programs Summary No. 37-33, IV, June 30, 1965.

10. Elias, P., "Coding for Noisy Channels," *IRE Convention Report,* 4:37-46, 1955.

11. Bussgang, J. J., "Some Properties of Binary Convolutional Code Generators," *IEEE Transactions on Information Theory,* IT-II, Jan. 1965.

12. Lin, S., and Lyne, H., "Some Results on Binary Convolutional Code Generators," *IEEE Transactions on Information Theory,* IT-13 (1), Jan. 1967.

13. Massey, J. L., "Convolutional Coding Techniques for Data Protection," NASA Grant NGR-15-004-026, Jan. 1968.

14. Jacobs, I. M., "Sequential Decoding for Efficient Communication from Deep Space," *IEEE Transactions on Communication Technology,* COM-15 (4), Aug. 1967.

15. Hutchinson, D. W., "A New Uniform Pseudo-Random Number Generator," *Communications of the ACM,* 9(6), June 1966.

16. Wozencraft, J. M., and Jacobs, I. M., *Principles of Communication Engineering,* New York: J. Wiley and Sons, 1967.

17. Savage, J. E., "The Computation Problem with Sequential Decoding," Lincoln Laboratory, M.I.T., Technical Report 371, Feb. 1965.

18. Jacobs, I. M., and Berlekamp, E. R., "A Lower Bound to the Distribution of Computation for Sequential Decoding," *IEEE Transactions on Information Theory,* IT-13, Apr. 1967.

Appendix A

Statistics at Output of a PCM Matched Filter

It is well known that a matched filter and a cross-correlator give identical output signal to noise ratios when the correlator's local signal is a replica of the transmitted waveform. For this derivation, a synchronous subcarrier, PCM signal source is assumed. In addition, a white, additive gaussian noise channel and a loss-less symbol correlator are assumed. Figure A-1 is a diagram of the communications system model. A loss-less, linear RF modem is assumed but is not shown explicitly. It is not necessary to include the subcarrier modem, since it does not affect the results. However, it is included just to emphasize this point.

The following definitions apply to this appendix as well as to the body of the paper:

$x(t)$ = The serial NRZ-PCM binary digit source signal;

f_s = the subcarrier clock, which is synchronous with $x(t)$;

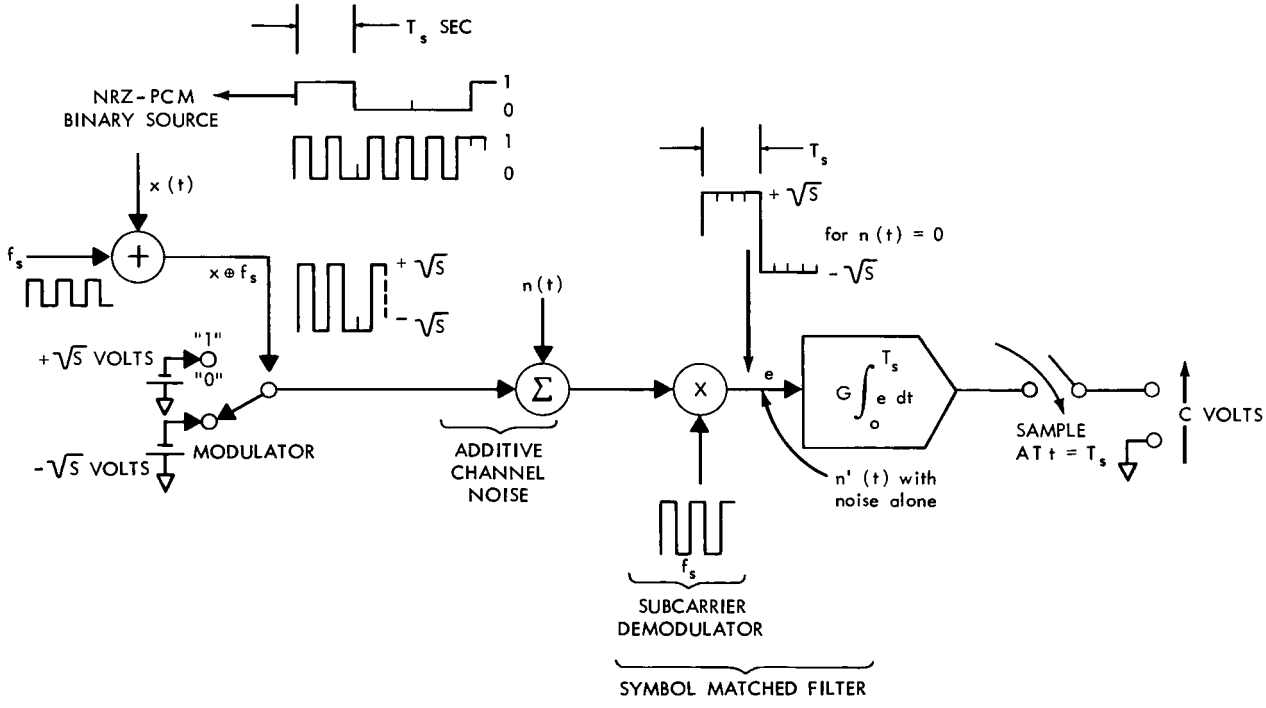

Figure A1—Signal, channel, and matched-filter models.

T_s = duration of a binary symbol, x, in seconds;

S = the signal power in watts;

n(t) = the additive channel noise; it is stationary, white, and gaussian with spectral density N_0 watts per Hz, and has zero mean;

G = the integrator's gain constant in $sec.^{-1}$; and

C = the symbol correlation voltage at $t \geq T_s$, in volts.

Because the noise is additive and the filter is linear, the signal and noise effects on C may be found separately and superposition applied.

Signal Only

If n(t) = 0, then C is deterministic for each x. From the model in Figure A-1, it is clear that, for the 2-cycles-per-symbol subcarrier shown or any "n"-cycles-per-symbol PCM subcarrier, "e" at the integrator input will always be of the same form. (n = 1, 2, 3, . . .). Thus

$$C]_{x=1} = G \int_0^{T_s} \sqrt{S}\, dt = G\sqrt{S}\, T_s \qquad \text{(A-1)}$$

and,

$$C]_{x=0} = G \int_0^{T_s} -\sqrt{S}\, dt = -G\sqrt{S}\, T_s . \qquad \text{(A-2)}$$

Noise Only

The autocorrelation function of white, gaussian noise will be needed in the derivation below. It is well known that the autocorrelation function and the power spectral density are a Fourier Transform pair. Therefore

$$\phi(\tau) = \frac{1}{2} \int_{-\infty}^{\infty} P(f)\, \epsilon^{+jwt}\, df , \qquad \text{(A-3)}$$

where

$\phi(\tau)$ = the autocorrelation function of some random signal, and

P(f) = the power spectral density of the random signal.

$\phi(\tau)$ is here defined as

$$\phi(\tau) = E\left[n(t)\, n(t-\tau)\right], \tag{A-4}$$

where

$n(t)$ is the random signal and $E(\ )$ denotes mathematical expectation over an ensemble.

The autocorrelation function of $n(t)$ may be found from Equation A-3, since $P(f) = N_0$ is known. Thus

$$\phi_n(\tau) = \frac{1}{2}\int_{-\infty}^{\infty} N_0\, \epsilon^{+jwt}\, df$$

$$= \frac{N_0}{2}\,\delta(\tau), \tag{A-5}$$

where $\delta(t)$ is the Dirac delta function.

Now it is desired to calculate the probability density function for the "noise alone" matched-filter output. The effects of the subcarrier multiplier will first be considered.

The effect of the subcarrier multiplier on $n(t)$ can be found formally, using the convolution theorem. However, it can be simply observed that, since the multiplier multiplies only by +1 and -1 and $n(t)$ is an infinite bandwidth signal, the amplitude statistics of e, the product, are unchanged. The power spectral density, N_0, is also unchanged. The noise time-sequence output from the multiplier, $n'(t)$, is not $n(t)$, but its amplitude statistics and power spectrum are unchanged.

The effects of the integrator may now be treated. Since the integrator is a linear summation device and the input noise, $n'(t)$, has gaussian amplitude probability density, then the output, C, must also be gaussian. Knowing the mean and variance of C will then completely specify its statistics.

Define C_N = the matched-filter output with noise alone, $n'(t)$, as input. Then

$$\text{mean}\,(C_n) \triangleq E(C_n)$$

$$= E\left(G\int_0^{T_s} n'(t)\,dt\right),$$

and, since the expectation is computed over an ensemble,

$$\text{mean}\,(C_n) = G\int_0^{T_s} E\left[n'(t)\,dt\right] = 0 \tag{A-6}$$

where E( ) = mathematical expectation over an ensemble of events. The variance of C_N is, by definition,

$$\mathrm{VAR}(C_N) = E\left[(C_N - \mathrm{mean})^2\right]$$

$$= E(C_N^2)$$

here, and

$$\mathrm{VAR}(C_N) = E\left[G\int_0^{T_s} n'(t)\,dt \cdot G\int_0^{T_s} n'(t)\,dt\right].$$

The integrals may be combined provided the integrand parameters are made distinct. So

$$\mathrm{VAR}\ C_N = E\left[G^2\int_0^{T_s}\int_0^{T_s} n(u)\,n(t)\,du\,dt\right]$$

$$= G^2\int_0^{T_s}\int_0^{T_s} E\left[n(u)\,n(t)\right]du\,dt,$$

but, from Equations A-4 and A-5,

$$E\left[n(t)\,n(t-\tau)\right] = \frac{N_0}{2}\,\delta(\tau).$$

Then

$$\mathrm{VAR}(C_N) = G^2\int_0^{T_s}\int_0^{T_s}\frac{N_0}{2}\,\delta(t-u)\,du\,dt$$

$$= \frac{G^2 N_0}{2}\int_0^{T_s} dt = \frac{G^2 N_0 T_s}{2}. \qquad \text{(A-7)}$$

Filter Output Statistics With Signal and Noise

With combined signal and noise entering the matched filter, then

$$\text{mean (C)} = \text{mean (signal)} + \text{mean (noise)}$$

$$= \begin{cases} G\sqrt{S}\,T_s & \text{for} \quad x = 1 \\ -G\sqrt{S}\,T_s & \text{for} \quad x = 0 \end{cases} . \qquad \text{(A-8)} \quad \text{(A-9)}$$

The density of C is of course gaussian with variance given by (A-7). A useful additional result is the mean to standard deviation ratio of C:

$$\frac{\mu_1}{\sigma} = \frac{G\sqrt{S}\,T_s}{\sqrt{\frac{G^2 N_0 T_s}{2}}} = \sqrt{2\frac{ST_s}{N_0}} \qquad \text{(A-10)}$$

where

μ_1 = mean of C, given $x = 1$, and

σ = standard deviation of C = $[\text{VAR(C)}]^{1/2}$.

Appendix B

The Sequential Decoder Computer Subroutine

To complete the definition of the sequential decoder algorithm actually used in the simulations, a listing of the FORTRAN IV sequential decoder subroutine is given here. Most of the program is self-explanatory except for some of the FORTRAN variable and array names. The routine was written to handle rate 1/2, 1/3, or 1/4 codes with up to 1024 bits per data frame. In the following definitions, references will be made to Figure 16 as the decoder model.

LDATA = the number of bits per frame.

IV = the number of transmitted symbols per encoder input bit.

LFRAME = the number of nodes, of IV symbols each, in a frame of data. This includes the tail symbols.

IQ = the received symbol memory array. It is the Q array shown in Figure 16.

IOFLOW = the overflow limit set for the move-forward and error loops.

ITAPS = the array describing the encoder generator matrix. It is composed of 0 and 1's.

KSRL = the equivalent of the temporary data-storage shift register in Figure 16.

IP = the bit number or current node "pointer."

XHAT = the estimated data bit equivalent to $\hat{D}_i$ in Figure 16.

ACCUM = the accumulated node metric.

T = the running threshold.

IDELTA = the threshold spacing increment.

Programming KSRL to perform as a shift register in FORTRAN is inefficient. Instead, an appropriate 32-bit window is established in KSRL for the local encoder by indexing alone. Hence the use of the following:

$$\text{INDEX} = \text{LFRAME} - \text{IP} + 1.$$

Thus KSRL (LFRAME) contains the first data bit hypothesis, etc.

The program listing is given in Figure B-1.

```
      SUBROUTINE SEQDEC (LDATA, LFRAME, IV, IDELTA, IOFLOW, ITAPS, KSRL,
     1 KRAP, LOOP 1, LOOP 2, IQ, ICZERO, ICONE, LAMBDA)
      COMMON IP
      COMMON LFRAME
      INTEGER XHAT, T
      DIMENSION IQ (1024, 4), KSRL (1024), ITAPS (4, 32),
     1          METRIC (1024, 2), LAMBDA (16, 2)
C     ******************************************************************************
C     THE FOLLOWING SETS UP THE DECODER AT THE BEGINNING OF A FRAME
C     ******************************************************************************
      IP = 1
      T = 0
      IACCUM = 0
C     ******************************************************************************
C     KSRL IS THE LOCAL WORKING REGISTER, 1024 BIT FIXED LENGTH
C     ******************************************************************************
      DO62 I = 1,1024
   62 KSRL (I) = 0
      LOOP 1 = 0
      LOOP 2 = 0
C     ******************************************************************************
C     THE DECODER IS NOW READY TO SEARCH FORWARD ON FIRST BRANCH
C     ******************************************************************************
C
    1 INDEX = LFRAME - IP + 1
C     ******************************************************************************
C     DATA BITS IN KSRL ARE ACCESSED FOR METRIC LOOK UP BY INDEXING
C     THROUGH KSRL
C     ******************************************************************************
      KSRL (INDEX) = 0
      IF (IP .GT. LFRAME) GO TO 500
    4 CALL NODMET (IP, IQ, KSRL, IV, ITAPS, LAMBDA, ICZERO, ICONE, LFRAME)
C     ******************************************************************************
C     THE NODMET SUBROUTINE PERFORMS THE FOLLOWING FUNCTIONS
C         (A) GENERATES THE LOCAL CODE FOR EITHER HYPOTHESIS
C         (B) BASED ON EACH SYMBOL QUANTILE IN IQ AND THE LOCAL CODE
C          IT ACCESSES THE LAP METRIC IN LAMBDA
C         (C) EACH OF THE IV SYMBOL METRICS ARE THEN ADDED TO GIVE
C          THE NODE METRIC
C         (D) THE NODE METRIC FOR EACH BIT HYPOTHESIS IS THEN OUTPUT,
C          NAMELY ICONE AND ICZERO
C         (E) WHEN IP = LAST DATA BIT, THE METRIC FOR IT AND THE
C          ENTIRE TAIL IS USED
C     ******************************************************************************
      IF (ICZERO .GT. ICONE) GO TO 999
      XHAT = 1
      NODVAL = ICONE
      METRIC (IP, 1) = ICONE
      METRIC (IP, 2) = ICZERO
      GO TO 14
  999 XHAT = 0
      NODVAL = ICZERO
      METRIC (IP, 1) = ICZERO
      METRIC (IP, 2) = ICONE
C
C     ******************************************************************************
C     THE PATH WITH THE LARGEST METRIC IS COMPARED TO THRESHOLD
C     ******************************************************************************
   14 IACFWD = IACCUM + NODVAL
      IF (IACFWD - T) 2, 2, 3
C
```

Figure B1—Program listing (1 of 2).

```
C       ************************************************************************
C       THIS IS THE NORMAL MOVE FORWARD UP DATE PATH
C       ************************************************************************
      3 LOOP 1 = LOOP 1 + 1
    200 IF (LOOP 1.GT.IOFLOW) GO TO 69
        KSRL (INDEX) = XHAT
        IACCUM = IACFWD
        IP = IP + 1
C       THE NEXT CARD PREVENTS TIGHTENING OF T WHEN IN THE ERROR MODE
        IF ((IACCUM-NODVAL).GT.(T + IDELTA)) GO TO 1
C       ************************************************************************
C       THE FOLLOWING CARDS TIGHTEN T BY DELTA
C       ************************************************************************
        IF (IACCUM .GT. (T + IDELTA)) GO TO 20
        GO TO 1
     20 T = T + IDELTA
        GO TO 1
C
C       ************************************************************************
C       THE FOLLOWING IS THE THRESHOLD VIOLATED LOOP
C       ************************************************************************
      2 LOOP 2 = LOOP 2 + 1
    201 IF (LOOP 2.GT.IOFLOW) GOTO 69
        IPBK = IP-1
        IF (IPBK) 17, 17, 8
     17 T = T - IDELTA
        GO TO 1
      8 IACCBK = IACCUM - METRIC (I PBK, 1)
        IF (IACCBK - T) 17, 17, 10
     10 IP = IPBK
        IACCUM = IACCBK
C       ************************************************************************
C       THIS FORCES THE CODER DOWN THE ALTERNATE PATH IN THE TREE
C       IF WE BACKED DOWN VIA THE HIGHEST METRIC PATH
C       ************************************************************************
        IF (METRIC (IP,1) - METRIC (IP, 2)) 2, 39, 39
C       ************************************************************************
C       THESE NEXT FIVE CARDS COMPLEMENT KSRL (INDEX)
C       ************************************************************************
     39 INDEX = LFRAME - IP + 1
        IF (KSRL (INDEX)) 40, 40, 41
     40 KSRL (INDEX) = 1
        GO TO 42
     41 KSRL (INDEX) = 0
     42 XHAT = KSRL (INDEX)
        NODVAL = METRIC (IP, 2)
        METRIC (IP, 1) = NODVAL
C       NEXT CARD PREVENTS SEARCHING DOWN OTHER METRIC PATH REPEATEDLY
C       IF THE NODE METRIC PAIR ARE EQUAL
        METRIC (IP, 2) = 1000
        GO TO 14
C       ************************************************************************
C       THIS ENDS THE SEQUENTIAL ALGORITHM. THERE ARE 3 EXIT PATHS
C       FROM THE ALGOR. THE NORMAL EXIT IS STATMNT 1. THE OVERFLOW EXITS
C       ARE STATMNTS 200, 201
C       ************************************************************************
     69 KRAP = 1
        RETURN
    500 KRAP = 0
        RETURN
        END
```

Figure B1—Program listing (2 of 2).

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Notes, and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546