NSG -5052

(NASA-CR-146846) COMPARISON OF RATE
ONE-HALF, EQUIVALENT CONSTRAINT LENGTH 24,
BINARY CONVOLUTIONAL CODES FOR USE WITH
SEQUENTIAL DECODING ON THE DEEP-SPACE
CHANNEL (Notre Dame Univ.) 41 p HC $4.00 G3/32

N76-21372

Unclas
25177

Department of

ELECTRICAL ENGINEERING

UNIVERSITY OF NOTRE DAME, NOTRE DAME, INDIANA

Comparison of Rate One-Half, Equivalent Constraint Length 24, Binary Convolutional Codes for Use with Sequential Decoding on the Deep-Space Channel*

James L. Massey
Freimann Professor of
Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

Technical Report No. EE-762

April 9, 1976

ABSTRACT

A "constraint length" for a convolutional code to be used with sequential decoding is usually specified by the allocation of a certain number of bits to be used in the "tail" of an encoded frame. The constraint length, K, of the code has conventionally been chosen to match this tail parameter; this report shows that several other options are available. Codes in various options which require the same tail allocation are said to have the same "effective constraint length", K_E, regardless of their actual constraint lengths.

Virtually all previously-suggested rate 1/2 binary convolutional codes with $K_E = 24$ (which is the effective constraint length specified for the International Ultraviolet Explorer spacecraft) are compared in this report. Their distance properties are given; and their performance, both in computation and in error probability, with sequential decoding on the deep-space channel is determined by simulation. Recommendations are made both for the choice of a specific $K_E = 24$ code as well as for codes to be included in future coding standards for the deep-space channel.

A new result given in this report is a method for determining the statistical significance of error probability data when the error probability is so small that it is not feasible to perform enough decoding simulations to obtain more than a very small number of decoding errors. This result should be of general usefulness in the efficient design of decoding simulation experiments.

* This research was supported by the National Aeronautics and Space Administration under NASA Grant NSG 5025 at the University of Notre Dame in liaison with the NASA Goddard Space Flight Center.

I. INTRODUCTION

In this report, we give the results of comparing, by their performance on a simulated deep-space channel, virtually all of the rate one-half, binary, convolutional codes of rate $R = 1/2$ and "equivalent constraint length" $K_E = 24$ which have been proposed for use with sequential decoding on the deep-space channel. These values of R and K_E are those selected for the International Ultraviolet Explorer (IUE) spacecraft. This report supersedes an earlier report [1] which gave a similar, but less extensive, comparison of $K_E = 32$ codes; $K_E = 32$ having earlier been considered for use in the IUE spacecraft.

Section II contains the definitions of terms used herein and the descriptions of the various design options for the convolutional encoder/sequential decoder (CE/SD) system. In Section III, we describe the simulation procedures employed and list the various codes which were compared. We then give, in Section IV, our recommendations for the code to be used in the IUE CE/SD system. We give consideration also in Section IV to many factors which should be considered in the development of future coding standards for deep-space communications.

Because of the very small decoding error probability for sequential decoding, even at the rather short $K_E = 24$ constraint length, it was necessary to give careful consideration to the statistical significance of the data obtained. A novel method to characterize this statistical significance was developed and is described in Section III of this report.

II. CODE PARAMETERS AND DESIGN OPTIONS

A. Convolutional Coding Terminology

We give here a brief description of a convolutional encoder for use with sequential decoding. Whenever possible, we use the same nomenclature as in the preliminary coding standard [2] developed at the Goddard Space Flight Center.

Figure 1 gives the general block diagram of a rate R = 1/2, binary, convolutional encoder. The encoder is completely specified by its

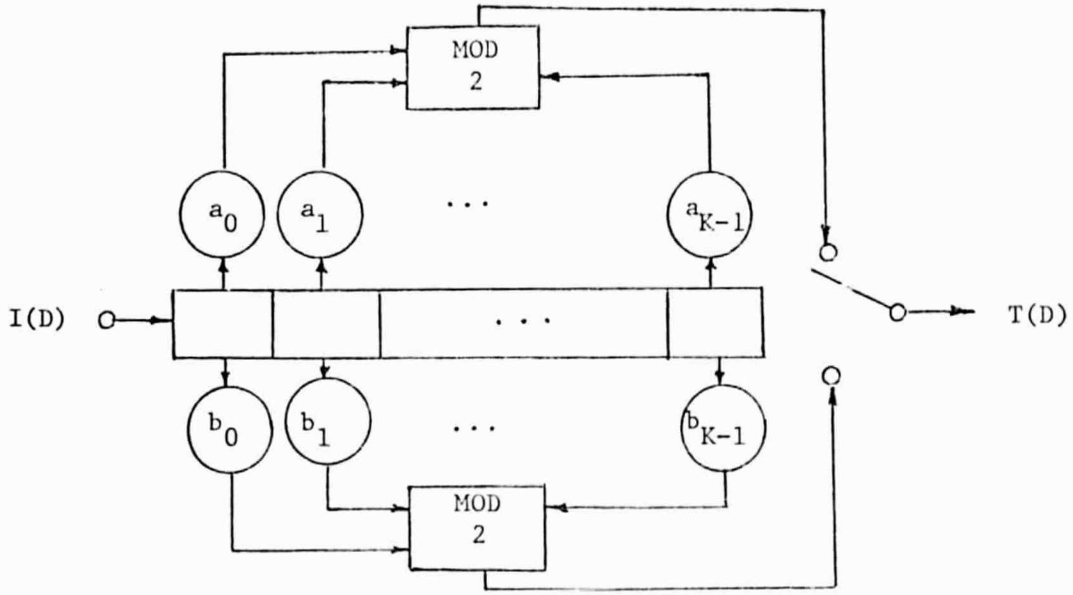

Fig. 1. A General R = 1/2 Binary Convolutional Encoder.

generator polynomials $G_1(D)$ and $G_2(D)$, where

$$G_1(D) = a_0 + a_1D + \dots + a_{K-1}D^{K-1}$$

and

$$G_2(D) = b_0 + b_1D + \dots + b_{K-1}D^{K-1}.$$

Each binary digit a_i (or b_i) is 1 or 0, respectively, according as to whether or not there is a connection from the corresponding shift-register stage to the corresponding modulo-two adder. For example, $a_1 = 1$ would signify a connection from the second shift-register stage to the upper modulo-two adder in Fig. 1, whereas $b_1 = 0$ would signify the absence of a connection from the second shift-register stage to the lower modulo-two adder in Fig. 1. It is also customary to speak of the generator functions which are the binary K-tuples

$$G_1 = a_0 a_1 \dots a_{K-1}$$

$$G_2 = b_0 b_1 \dots b_{K-1}.$$

The constraint length, K, is the length of the shift-register in Fig. 1, i.e., the number of information bits in the span which, at any clock instant, determine the two encoded bits formed by the modulo-two adders. The information sequence to be encoded is represented by the polynomial

$$I(D) = i_0 + i_1 D + \dots + i_{L-1} D^{L-1}$$

where i_t is the information bit present in the leftmost shift-register stage at clock instant t. Note that, at each clock instant, one information bit enters the encoder shift-register but two encoded bits are formed. These encoded bits are multiplexed into one stream to form the encoded sequence which is denoted by T(D) in Fig. 1. The quantity L is the frame length in information bits.

Whenever sequential decoding is used, the information sequence is followed by a tail of T zeroes so that a total of L + T clock instants are used by the encoder to form the entire encoded frame. This segmentation into "frames" is required to permit independent processing of frames so that the sequential decoder can move on to the processing of the next frame whenever it encounters a received frame which causes excessive decoding computation; such frames thus result in deletion of the frame rather than decoding errors.

The choices $a_0 = b_0 = 1$ are always made, so that there is always a connection from the first shift-register stage to both modulo-two adders in

Fig. 1. The code is said to be systematic when $a_1 = a_2 = \ldots = a_{K-1} = 0$, i.e., when the encoded sequence from the upper modulo-two adder in Fig. 1 is just the information sequence itself (the upper adder is then, in fact, not necessary). Because the information sequence appears directly in alternate positions of the encoded sequence, it can be trivially recovered from the latter in the absence of errors. In a non-systematic code, such simple recovery of the information sequence is not possible. The code is said to be quick look-in (QLI) if $G_2(D) = D + G_1(D)$, i.e., if $a_i = b_i$ for $i \neq 1$ but $a_1 \neq b_1$. For a QLI code [3], $G_1(D) + G_2(D) = D$ so that the information sequence can be recovered from the encoded sequence, except for an unimportant delay of one clock unit, by the simple circuit shown in Fig. 2.

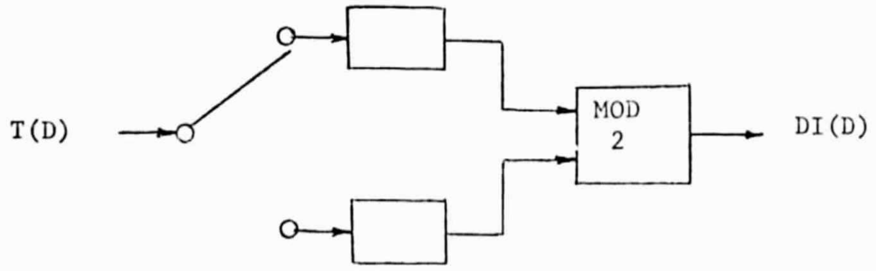

Fig. 2. Recovery of the Information Sequence from the Error-Free Encoded Sequence for a QLI code.

Simple recovery of the information sequence from the transmitted sequence is desirable both for encoder check-out and for extracting the engineering data from the "hard-decisioned" received sequence without the necessity of full decoding. The probability of error in the latter case is the minimum possible, viz. the error probability in the hard-decisions themselves, for systematic codes. For QLI codes, this error probability is at most twice this absolute minimum [3], and is the minimum for nonsystematic codes.

B. Implementation Options and Equivalent Constraint Length

It is not generally realized that there are several options available to the CE/SD system designer as to the choices of K and T even after the frame parameters have been frozen. To see this, we show in Fig. 3 a conceptual view of the complete encoded frame in an R = 1/2 CE/SD system. (We neglect the "synch patterns" which in general are appended to the encoded frame to form the entire transmitted frame. The active portion of the encoded frame consists of the 2L encoded digits formed by the modulo-two adders in Fig. 1 during the L clock instants in which information bits enter the shift-register. The passive portion of the encoded frame consists of P further encoded bits formed during the T clock instants in which the tail of zeroes is inserted into the encoder. The total encoded frame length, F, is then

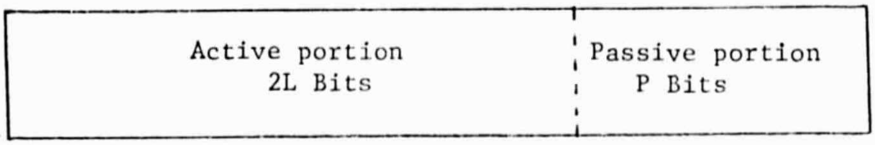

Fig. 3. Structure of a Complete Encoded Frame for R = 1/2

$$F = 2L + P.$$

We now consider the options still available to the CE/SD designer after L and P have been frozen.

1. Conventional Option

We first describe the manner in which virtually all past CE/SD systems have been implemented, which we shall call the "conventional option." This option is:

Conventional option: Choose $T = K - 1 = P/2$.

In this option, the passive portion of the frame is just the 2T encoded bits formed during the T clock instants in which the tail of zeroes enters the encoder. This option has the advantageous feature that the encoder may immediately begin encoding the next frame after completing the last because when the first information bit of the next frame enters the encoder, it automatically finds only 0's residing in the remaining $T = K - 1$ shift register stages.

We now define the <u>equivalent constraint length</u>, K_E, of any CE/SD option to be

$$K_E = P/2 + 1.$$

Our rationale for this definition is that

$$K_E = K \quad \text{(conventional option)},$$

and that, since P completely determines the actual "overhead" in the encoded frame needed to effect segmentation of frames, only P should be used as the measure of "constraint length" within the frame.

2. <u>Δ-Undersized Tail Option</u>

There is no reason to choose T so large as $K - 1$ except for the convenience of the "automatic initialization" (as just described for the conventional option) for encoding of subsequent frames. Thus, we define:

Δ-Undersized tail option: Choose $T = K - 1 - \Delta = P/2$.

(Δ, some positive integer).

In this option, the passive portion of the frame again consists of the 2T encoded digits formed as the tail of T zeroes is inserted into the encoder. From the definition of K_E, we see that

$$K_E = K - \Delta \quad \text{(Δ-undersized tail option)}.$$

This option has the minor disadvantage that, in order to begin encoding of the next frame, it is necessary to "stuff" zeroes into the Δ rightmost stages of the encoder shift-register immediately upon conclusion of the encoding of the frame. It has, however, two major advantages over the conventional option. First, the frame decoding error probability, P_e, with sequential decoding is in general proportional to the frame length in information bits, L, when $K = T + 1$ as in the conventional option. However, fixing P (and thus also fixing T in the Δ-undersized tail optio"), we have shown elsewhere [4], [5] that P_e decreases rapidly as K is increased up to the point where $\Delta \approx \log_2 L$ where no further improvement results. Moreover, for

$$\Delta > \log_2 L,$$

the resulting P_e is independent of L. We can now see a second advantage in the possibility to standardize encoder design. By choosing one good code with a large value of K, we will have a near-optimal CE/SD system over a very wide range of telemetry formats, viz., all those such that

$$K \geq P/2 + \log_2 L + 1.$$

3. Systematic Partial Tail-Suppression Option

This option can be used only with systematic codes and its possibility was first reported by us quite recently [6]. It rests upon the obvious fact that, since the upper adder in the circuit of Fig. 1 for a systematic code emits only zeroes as the tail of T zeroes is inserted into the encoder, it is both unnecessary and wasteful to multiplex these known zeroes into the encoded sequence. Thus, in this option, the passive portion of the encoded frame consists of only the $P = T$ bits from the lower adder in Fig. 1 during the T clock instants that the tail of zeroes is inserted.

Systematic partial tail-suppression option: Choose $T = K - 1 = P$. For this opcion, we see from the definition of K_E that

$$K_E = (K - 1)/2 + 1.$$

In this option, we again have $T = K - 1$ so that automatic initialization for successive encoded frames is achieved as in the conventional option.

The main advantage over the conventional option, when nonsystematic codes are used with the latter, is the simpler and more reliable recovery of the information sequence from the hard-decisioned received sequence, although this advantage is only a factor of 2 when QLI nonsystematic codes are considered. There are two minor disadvantages of this option compared to the conventional option. First, because only one encoded digit is formed at each encoder clock instant during the passive portion of the frame, the encoder clock must double its speed when it enters the passive portion of the frame. Secondly, the frame decoding error probability, P_e, may be slightly increased. It has long been known [3] that, for the same K, the best nonsystematic code gives a far smaller P_e than the best systematic code. At R = 1/2, however, the K in this option for the same K_E (i.e., same P) is related to the K in the conventional option as

$$K_{sys} - 1 = 2(K_{nonsys}-1)$$

or

$$K_{sys} \approx 2K_{nonsys} .$$

The available evidence indicates that, when their constraint lengths are so related, the best systematic code gives about the same P_e as the best nonsystematic code [3], but with a slight advantage for the nonsystematic code [7].

4. Systematic Partial Tail-Suppression with Δ-undersized Tail Option.

This option is that obtained by combining the two unconventional features of the two previous options, viz. removal of known zeroes from the encoded sequence and use of a tail shorter than that required to automatically initialize the encoder between frames. This option can thus be described as:

Systematic partial tail-suppression
with Δ-undersized tail option: Choose $T = K - 1 - 2\Delta = P$
(Δ, some positive integer)

We see for this option that

$$K_E = (K + 1)/2 - \Delta.$$

Because this option is a combination of the "unconventional" features of the two previous options, it shares both the advantages and disadvantages of each. On the minus side, it requires stuffing of zeroes to initialize the encoder between frames and also requires the encoder clock rate to double in the tail. One would expect also that, for the same K_E and Δ, the frame decoding error probability, P_e, would be slightly greater than for option 2 with a comparably good nonsystematic code; however, this drawback is probably not significant. [It should be noted, however, that the constraint lengths K in these two options, for the same K_E, are related as

$$K_{sys} = 2K_{nonsys} - 1.$$

As will be seen in Subsection C below, this rules out option 4 as a viable option because the required K_{sys} exceeds the value of K for which good systematic codes are known at the present time.] On the positive side for option 4, the resulting P_e would be less than in option 3 and independent of the frame length in information bits, L. Moreover, one encoder could be standardized and used for all applications where its constraint length K satisfies

$$P + 2 \log_2 L + 1$$

Finally, because the encoder is systematic, the information sequence can be simply extracted, with maximum reliability, from the hard-decisioned received sequence.

C. Summary, by Option, of the $K_E = 24$ Codes Compared in This Report.

The equivalent constraint length, K_E, has been specified at $K_E = 24$ for the IUE spacecraft CE/SD system, i.e., $P = 2(K_E - 1) = 46$ bits have been allocated for the passive portion of the encoded frame.

In the conventional option 1, the specification $P = 46$ requires that a code with $K = 24$ be chosen.

In the Δ-undersized tail option, the specification $P = 46$ requires that a code with $K = 24 + \Delta$ be chosen. We have specified further that $\Delta = 24$ for several reasons. First, the resulting $K = 48$ is about as large a constraint length as that for which good nonsystematic convolutional codes are presently known. Moreover, the corresponding $\Delta = 24$ is large enough to accomodate as wide a range of frame lengths, L, in information bits and allocated passive frame length, P, as is likely to be considered in any future deep-space applications, so that a standardized $K = 48$ encoder is robust enough for all likely future applications. Finally, $K = 48$ is a convenient constraint length for software sequential decoding as it is well-matched to most computer wordlengths. Hence, we consider hereafter only $K = 48$ codes for option 2.

In the systematic partial tail-suppression option 3, the specification $P = 46$ requires that a systematic code with $K = 47$ be chosen.

In the systematic partial tail-suppression with Δ-undersized tail option 4, the specification $P = 46$ requires that a systematic code with $K = 47 + 2\Delta$ be chosen. The same considerations as just discussed for option 2 suggests that $\Delta = 24$ would be a sensible choice. However, the resulting $K = 95$ is far beyond the range for which good systematic codes are presently known. In fact, the systematic codes reported earlier [8] with $K \leq 61$ are the longest good codes presently known. The $K = 61$ code would allow the choice $\Delta = 7$ in the IUE spacecraft CE/SD, but this rather small Δ would accomodate a very small additional range of frame parameters so that standardization at this value would be unwise. Because such standardization is perhaps the most attractive feature of option 4 vis-a-vis option 3, its impossibility with the present state of the art in code construction has led us to rule out option 4 as presently unviable.

In Table I, we summarize the types of convolutional codes, by option, which we shall consider as candidates for use in the IUE spacecraft CE/SD system.

Option No.	Option Name	K_E	K	Remarks
1	Conventional	24	24	Viable
2	(Δ=24)-Undersized Tail	24	48	Viable
3	Systematic Partial Tail-Suppression	24	47	Viable
4	Systematic Partial Tail-Suppression with (Δ-24) Undersized Tail	24	95	Unviable at Present

Table I: Required constraint length K, for convolutional codes with K_E = 24, by implementation options.

III. SIMULATION RESULTS AND COMPARISONS

A. Codes Selected for Comparison

We now list, by option, all of the codes used in the simulations to be reported here. These codes comprise all of the known convolutional codes, with the required co.straint lengths, that appear attractive for use in deep-space communications.

1. Codes for the conventional option.

a. The Massey-Costello (MC) quick-look-in (QLI) code with $K = 24$. (MCQLI-24 code).

This code is obtained by shortening the $K = 48$ QLI code given by Massey and Costello [3]. This $K = 24$ code is the recommended code for this length in the preliminary coding standard [2].

b. The first Johannesson (J) QLI code having an optimum distance profile (ODP) with $K = 24$. (1JQLIODP-24 code).

This is the first of two $K = 24$ QLI codes found by Johannesson [9] having the optimum distance profile (ODP) property which will be discussed further in Subsection B below.

c. The second Johannesson QLI code having an ODP with $K = 24$. (2JQLIODP-24 code).

d. The Bahl-Jelinek (BJ) $K = 24$ complementary code (BJ-24 code).

Complementary codes were defined by Bahl and Jelinek [10] as codes for which $a_0 = b_0 = a_{K-1} = b_{K-1} = 1$, but $a_i \neq b_i$ for $1 \leq i < K - 1$. This is the best $K = 24$ code in this class as found by exhaustive searching [10].

e. The $K = 24$ quadratic residue code. (QR-24 code).

This is a code which was found by Massey, Costello and

Justesen [11] using a technique for constructing convolutional codes from known cyclic block codes. This code actually has K = 23 since $a_{23} = b_{23} = 0$.

f. The K = 24 optimally-truncatable code (OT-24 code).

This code, which was found during the progress of the research reported here, will be described more fully in Subsection B below.

2. Codes for the (Δ=24)-undersized tail option.

a. The Massey-Costello QLI code with K = 48. (MCQLI-48 code).

This K = 48 code is the recommended code for this length in the preliminary coding standard [2]. (However, its use as here at $K_E = 24$ in the undersized tail option is not considered in the preliminary standard.)

b. The Johannesson QLI ODP code with K = 48. (JQLIODP-48 code).

This code was reported in [6] which contains extensions of Johannesson's earlier work [9].

3. Codes for the systematic partial tail-suppression option.

a. The adjoint [12] of Bussgang's optimal systematic code [12] as extended by Lin and Lyne [13], and later Forney [14], to a good, but sub-optimum, code at K = 47. (BLLF-46 code).

This is the K = 48 systematic code recommended in the preliminary coding standard [2]. Its use by shortening to K = 47 actually gives a code with K = 45 because $b_{46} = b_{45} = 0$ in the generator.

b. Johannesson's systematic ODP code with K = 47 (JSODP-47 code).

This code was reported in [6] in the extension given there of Johannesson's earlier work. There are no other K = 47 systematic codes known which appeared promising enough to be considered in this comparison.

In Table II, we summarize for easy reference the ten $K_E = 24$ codes that were used in the simulations to be reported here. The generators are listed in octal form in the manner that G_1 = 111, 011, 011, 101, 011, 011, 110, 111 (binary, with commas shown to indicate segmentation for conversion to octal) is shown as G_1 = 73353367. Also shown in Table II is the free distance, d_{free}, of the code which is the minimum Hamming distance between two complete encoded sequences caused by different information sequences. The minimum distance, d_{23}, which is the minimum Hamming distance between two encoded sequences during the first K_E = 24 encoder clock instants caused by information sequences that have different values of i_0, is also given in Table II.

Code No.	Code Name	G1 (octal)	G2 (octal)	d_{23}	d_{free}
1a	MCQLI-24	73353367	53353367	9	17
1b	1JQLIODP-24	74042417	54042417	11	18
1c	2JQLIODP-24	74041567	54041567	11	19
1d	BJ-24	51202215	66575563	10	24
1e	QR-24	77441232	54502376	10	20
1f	OT-24	75105323	55105323	10	19
2a	MCQLI-48	73353367 67373553	53353367 67373553	9	$\geq$19
2b	JQLIODP-48	74042402 07121635	54042402 07121635	11	$\geq$19
3a	BLLF-47	40000000 00000000	71547370 13174650	9	19
3b	JSODP-47	40000000 00000000	67114545 75564666	11	17

Table II: Description of K_E = 24 chosen for comparison in simulations. (Where d_{free} is unknown, the best known lower bound is given, but these lower bounds are not expected to be tight. For instance, it is likely that d_{free} is at least 30 for codes 2a and 2b.)

B. Distance Properties of the Selected Codes

The distance properties of a convolutional code can be used to make a fairly reliable estimate of how that code will perform with sequential decoding. We now consider the distance properties of the above codes with a view to correlating their performance on the deep-space channel with these properties.

The i-th order column distance, d_i $(i = 0,1,2,\ldots)$ is defined to be the minimum Hamming distance between two encoded sequences over the first $i + 1$ time instants [$2(i+1)$ digits for rate 1/2 codes] resulting from information sequences with different values of i_0. Because of the linearity of the code, d_i is equal to the minimum Hamming weight of an encoded sequence over the first $i + 1$ time instants resulting from an information sequence with $i_0 = 1$. The free distance, d_{free}, may be expressed as

$$d_{free} = \lim_{i\to\infty} d_i$$

and is well-known to be the main determiner of error probability where sequential decoding is used [3]. The distance profile, $\underline{d} = [d_0,d_1,d_2,\ldots d_M]$, has been shown [9] to be the property of interest as far as computational performance with sequential decoding is concerned. The distance profile $\underline{d}$ is said to be superior to the distance profile $\underline{d}'$ if $d_i > d_i'$ for the smallest i such that $d_i \neq d_i'$; we denote this superiority by writing $\underline{d} > \underline{d}'$. We define the distance index i_j of a code to be the smallest integer i such that $d_i = j$. Because $d_0 = 2$ for all $R = 1/2$ codes having $a_0 = b_0 = 1$, i_2 is the first distance index of interest. Because $d_i \leq d_{free}$ for all i, $i_{d_{free}}$ is the largest distance index of interest. The distance indices i_j for $j = 2,3,\ldots,d_{free}$ of course uniquely determine the distance profile $\underline{d}$, but in general they also provide further information, viz. that about the column distances d_i for $i > M$.

In Table III, we show the distance indices for all of the codes to be compared in our simulations of $K_E = 24$ codes. For codes 1d, 2a and 2b, the distance indices are shown wherever we have found their value, but the time requirements for computing higher distance indices were so great as to prohibit generation of the full set of distance indices. All of the distance indices shown were obtained by using a sequential decoding program, as was first suggested by Forney [14], which was programmed in assembly language for the IBM 370/158 computer. In this procedure, for a given threshold $T = j-1$, one explores all encoded sequences beginning with $i_0 = 1$ until their Hamming weight exceeds T. If the longest path of weight $T = j-1$ is $i + 1$ branches in length, then $d_j = i$. The program also checks the encoder state after each extension of an encoded path whose Hamming weight is $T = j-1$. If the all-zero state is ever encountered, then $d_{free} = j-1$ and the program is stopped. Otherwise, the threshold T is increased by 1 and the search continued until either a zero state is reached or the predetermined limit on CPU time is exceeded. The values of d_{free} given in Table II were all found in this manner, except that for the BJ-24 code whose d_{free} was found in [10]. The CPU time required for each code in Table II with $d_{free} \geq 19$ was approximately 15 minutes.

It is interesting to compare the codes in Table II in terms of their distance properties. In terms of their free distance, the ordering by quality of the codes is

$$(2b) \overset{?}{\geq} (2a) \overset{?}{>} (1d) > (1e) > (1c) = (3a) = (1f) > (3b) = (1a)$$

[ordering by free distance]

where the codes are shown in order of decreasing quality. In terms of their distance profile (where we treat all codes as having M = 47, the memory of the code with greatest memory), the ordering by quality of the codes is

Code No.	1a	1b	1c	1d	1e	1f	2a	2b	3a	3b
Code Name	MCQLI-24	1JQLIODP-24	2JQLIODP-24	BJ-24	QR-24	OT-24	MCQLI-48	JQLIODP-48	BLLF-47	JSODP-47
i_2	0	0	0	0	0	0	0	0	0	0
i_3	1	1	1	1	1	1	1	1	1	1
i_4	3	3	3	3	3	3	3	3	3	3
i_5	6	5	5	5	6	5	6	5	5	5
i_6	10	7	7	8	10	8	10	7	7	7
i_7	12	10	10	10	15	11	12	10	10	10
i_8	15	12	12	14	17	15	15	12	14	12
i_9	20	16	16	19	20	17	20	16	16	16
i_{10}	25	19	19	21	23	22	26	19	20	19
i_{11}	29	23	23	24	28	28	30	23	24	23
i_{12}	32	29	28	27	34	32	36	27	27	27
i_{13}	37	35	35	33	36	34	40	30	31	30
i_{14}	39	38	37	37	40	39	43	34	35	34
i_{15}	44	43	43	39	43	42	45	38	40	38
i_{16}	49	46	46	44	49	45	48	42	43	42
i_{17}	55	50	51	51	54	49	58	46	53	46
i_{18}	-	54	56	58	64	53	61	52	55	-
i_{19}	-	-	58	60	67	60	64	58	59	-
i_{20}	-	-	-	63	70	-	≥ 69	≥ 63	-	-
i_{21}	-	-	-	71	-	-	?	?	-	-
d_{free}	17	18	19	24	20	19	≥ 19	≥ 19	19	17

Table III: Distance properties of the codes chosen for comparison in simulations. (i_j is the smallest i such that $d_i = j$.)

(2b) = (3b) > (1c) > (1b) > (3a) > (1d) > (1f) > (1a) > (2a) > (1e)
[ordering by distance profile].

As we shall see below, these two orderings substantially agree with the ordering in performance with respect to decoding error probability and computation,respectively, for these codes on the simulated deep-space channel.

C. Description of the Simulation

The channel chosen for simulation was a white additive Gaussian noise channel (i.e., a deep-space channel) with an energy per transmitted bit, E, to one-sided noise power spectral density, N_o, ratio of $E/N_o = 1$ or 0 db. Because the code rate is 1/2, this corresponds to an energy per information bit, E_b, to N_o ratio of $E_b/N_o = 2$ or 3 db. The received digits were quantized to 3 bits using the quantization scheme suggested by Jacobs [15]. The result of this quantization is to convert the Gaussian channel to a 2-input, 8-output discrete memoryless channel (DMC) whose transition probabilities are as given in Table IV. This DMC is the actual channel that was simulated, the various transition probabilities being obtained by use of a standard random number routine. The cut-off rate, R_o, of the Gaussian channel (which is sometimes also denoted as R_{comp}) was .5481, whereas that of the resulting DMC is .4913 so that the loss due to quantization is 0.48 db. Note that $R_o = .4913$ is slightly less than the code rate R = .5000, which indicates that this DMC is actually somewhat noiser than the channels on which the CE/SD system would actually be used. This "noisier than usual" situation was chosen for simulation so that the probability of a decoding error would be large enough so that the number of decoded frames required to determine the decoding error probability would correspond to practical amounts of computing time.

output irput	0	0'	0''	0'''	1'''	1''	1'	1
0	.434	.197	.167	.111	.058	.023	.008	.002
1	.002	.008	.023	.058	.111	.167	.197	.434

Table IV: Transition probabilities for the DMC obtained by 3-bit quantization of the deep-space channel with $E_b/N_o = 3$ db for R = 1/2 coding.

The Fano sequential decoding algorithm [16] was used for decoding. The metrics used were as given in Table V.

rec'd trans'd	0	0'	0''	0'''	1'''	1''	1'	1
0	4	4	3	0	-8	-20	-34	-59
1	-59	-34	-20	-8	0	3	4	4

Table V: Metrics used for sequential decoding for the DMC given in Table IV.

The threshold increment Δ used in the Fano algorithm was 32. A frame size of L = 256 information bits was selected. A frame was declared to be erased when 100,000 computations were performed without completion of the decoding. This erasure limit of 390 computations per decoded information bit is well above what one would generally choose in practice, the reason for this choice being again to ensure a significant number of frames with decoding errors (rather than merely erasing most of those frames.) The number of computations used for an erroneously decoded frame was recorded so that one can determine how P_e would change if a smaller erasure limit on computation were chosen.

For all codes selected for comparison, 10,000 frames were decoded in the simulation. The same 10,000 received frames were decoded for each code so that any differences in computation and/or error probability would be due to the codes themselves.

Code No.	1a	1b	1c	1d	1e	1f	2a	2b	3a	3b
Code Name	MCQLI-24	1JQLIODP-24	2JQLIODP-24	BJ-24	QR-24	OT-24	MCQLI-48	JQLIODP-48	BLLF-47	JSODP-47
#Frame errors	4	4	2	0	0	5	1	0	1	0
#Bit errors	48	59	36	0	0	116	38	0	13	0
#Erased frames	35	24	23	25	27	25	26	22	17	24
#Frames with:										
$C \leq 400$	1058	1122	1118	1103	1037	1100	1063	1123	611	602
$C \leq 500$	3029	3283	3424*	3352*	2918	3194	2987	3288	2653	2689
$C \leq 650$	5145	5575	5552*	5426*	4961	5344	5175	5588	5135	5268
$C \leq 850$	6681	7029	7012	6965	6518	6838	6695	7013	6824	6849
$C \leq 1,200$	7852	8136	8115*	8051*	7755	7957	7860	8142	8004	8080
$C \leq 1,700$	8558	8754	8754*	8696*	8507	8627	8561	8756	8702	8744
$C \leq 2,400$	8992	9109	9119*	9074*	8971	9047	8983	9124	9098	9106
$C \leq 3,500$	9298	9385	9397*	9363*	9300	9346	9290	9402	9364	9382
$C \leq 5,000$	9497	9563	9550	9540	9507	9534	9485	9552	9553	9546
$C \leq 7,000$	9637	9671	9672*	9652*	9633	9665	9613	9678	9681	9672
$C \leq 10,000$	9737	9762	9765	9741	9745	9756	9729	9771	9774	9773
$C \leq 15,000$	9823	9826	9835*	9828*	9827	9831	9822	9833	9855	9831
$C \leq 20,000$	9859	9873	9872	9871	9864	9876	9869	9870	9881	9867
$C \leq 30,000$	9904	9916	9915	9913	9897	9913	9902	9922	9919	9920
$C \leq 50,000$	9941	9950	9948	9948	9940	9946	9947	9948	9942	9950
$C \leq 100,000$	9965	9976	9977	9975	9973	9975	9974	9978	9983	9976
C in frames with errors	36,369 12,491 933 48,065	1,185 16,346 23,410 85,658	70,218 9,740			734 62,585 37,149 2,969 3,047	49,449		17,922	

Table VI: Results of Decoding 10,000 frames of length L = 256 information bits on the simulated deep-space channel with an E_b/N_o of 3 db. (Values marked * were interpolated from nearest data points obtained in the simulation on the assumption that the computation is Pareto-distributed.)

D. Simulation Results

In Table VI, we show the results obtained from decoding the same 10,000 frames of length L = 256 on the simulated deep-space channel with an E_b/N_o of 3 db. Because no decoding frame errors were made with four of the codes, additional frames were decoded for these codes with the results shown in Table VII.

Code No	1d	1e	2b	3b
Code Name	BJ-24	QR-24	JQLIODP-48	JSODP-47
#Additional frames decoded	40,000	40,000	40,000	10,000
#Frame errors	0	2	0	2
#Bit errors	0	20	0	41
C in frames with errors		23,864 45,786		9,249 57,636

Table VII: Results of additional decoding simulations for those codes which gave no decoding errors on the first 10,000 decoded frames.

Approximately 15 minutes of CPU time on the IBM 370/158 system were required for each 10,000 decoded frames. Neither code 1d (the BJ-24 code) nor code 2b (the JQLIODP-48 code) yielded any decoding errors on a total of 50,000 decoded frames. It was not feasible to decode enough additional frames to distinguish which code actually gives the better decoding error probability.

As regards decoding error probability as determined by simulation, it can be seen from Tables VI and VII that the codes rank in quality as:

(2b) = (1d) > (1e) > (3b) > (3a) > (2a) > (1c) > (1a) > (1b) > (1f)

where we have resolved ties in the number of frames decoded in error by considering the number of information bits decoded in error. We note that this empirical ranking is substantially the same as the ranking by d_{free} given in subsection B above.

In comparing the frame decoding error probability, P_e, one should also give attention to the number of computations done in the erroneously decoded frames as this can be used to determine the change in P_e should a different frame erasure limit be used. For instance, we see from Table VI that if the (more realistic) limit of 10,000 computations for a decoded frame is used, codes 1a, 1b and 1f would yield 1, 1 and 3 erroneously decoded frames, respectively, compared to 4, 4 and 5, respectively, for the 100,000 computation limit.

As regards computational performance as determined by simulation, it can be seen from Table VI that the codes rank in quality as

(1b) > (2b) > (1c) > (1d) > (3b) > (1f) > (3a) > (2a) > (1a) > (1e)

where we have, somewhat arbitrarily taken the number of frames decoded with 850 or fewer computations as the indicator of quality. We note that this ranking is in substantial agreement with the ranking by distance profile, $\underline{d}$, given in subsection B above. In particular, we see the excellent computational performance of all of the optimum distance profile (ODP) codes.

A word should be said about the apparently poor computational performance of the two systematic codes (codes 3a and 3b) when the computation is small. The reason for this effect is that a tail of $T = 47$ branches (with only one digit per branch) is included in the frame for these codes, compared to only $T = 23$ branches (with two digits per branch) for all the other codes.

Because "computation" is measured in terms of branches searched, the T = 47 systematic codes necessarily require a small number of additional computations per frame, compared to the T = 23 nonsystematic codes, to account for the computation done on the extra tail branches. However, as the computation per frame increases, the additional computation becomes a negligible part of the total computation (which is the reason we compared computational performance at 850 computations or less per frame rather than some smaller number.)

E. Statistical Significance of the Decoding Error Probability as Determined from the Simulations.

The very small number of frame decoding errors, between 0 and 5 inclusive for the best and the worst codes, makes it essential to consider the statistical significance of the frame error probabilities determined therefrom. We now present a new method for testing this significance, a method which it appears could be of rather general usefulness in decoding simulation studies.

Suppose that N frames are decoded for a particular code and that the true frame decoding error probability is P_e. Because decoding of frames is done independently, and because $N >> 1$ while $P_e << 1$, the number of frame errors, X, is a Poisson-distributed random variable whose mean and variance are both equal to

$$\lambda = NP_e.$$

Let x be the observed value of X for the particular N frames actually decoded. Then x is the best estimate of both $E(X) = \lambda$ and $Var(X) = \lambda$, while x/N is the best estimate of $P_e = E(X)/N$. The problem is that, for x small (say, $x < 10$), the estimate of Var(X) is quite inaccurate so that the standard statistical "confidence level" techniques cannot be applied with any validity. We now show how this dilemma can be resolved.

We define the 100β% confidence interval, (λ_L, λ_U), for λ, given the observation x of X, in the manner that λ_L and λ_U are the largest and smallest numbers, respectively, such that

$$P(X < x) \geq \beta, \quad \text{all } \lambda \leq \lambda_L,$$

and

$$P(X > x) \geq \beta, \quad \text{all } \lambda \geq \lambda_U.$$

This is equivalent to saying that, had the code possessed an actual λ smaller than λ_L, the probability of observing fewer than the number of x of frame errors actually observed would have been at least β; and that, had the actual λ been larger than λ_U, the probability of observing more than the number of x of frame errors actually observed would have been at least β.

Because X is Poisson with mean and variance equal to λ, the frequency distribution for X, i.e., the probability of the event that x equals i, is given by

$$f_X(i) = \frac{\lambda^i}{i!} e^{-\lambda} .$$

Now, letting $F_j(\lambda)$ be the probability that $X \leq j$, we have

$$F_j(\lambda) = \sum_{i=0}^{j} \frac{\lambda^i}{i!} e^{-\lambda} .$$

It now follows, from the definitions of λ_L and λ_U, that

$$F_{x-1}(\lambda_L) = \beta$$

and

$$F_x(\lambda_U) = 1 - \beta.$$

For a given x, these equations may be readily solved, e.g., by the Newton-Raphson method, to determine λ_L and λ_U. In Table VIII, we give the 90% confidence intervals, so determined, for $0 \leq x \leq 5$.

x	λ_L	λ_U
0	0	2.30
1	.105	3.88
2	.532	5.29
3	1.107	6.65
4	1.76	7.99
5	2.52	9.06

Table VIII: 90% confidence intervals for λ given the observed value x for a Poisson random variable X.

Given the 100β% confidence interval (λ_L, λ_U) for λ, the corresponding 100β% confidence interval (P_{eL}, P_{eU}) for $P_e = E(X)/N = \lambda/N$ is just $(\lambda_L/N, \lambda_U/N)$. In Table IX, we give the 90% confidence levels for P_e as determined from the simulation results given in Tables VI and VII.

When the confidence interval for one code lies entirely to the left of that for another, then with at least 100β% confidence we can assert that the actual P_{eU} of the former code is smaller than that of the latter. In Figure 4, we have plotted the 90% confidence intervals on a logarithmic scale for all of the codes tested. We see from this figure that with at least 90%

Code No.	# frames decoded, N	# frame errors, x	best estimate of P_e, x/N	90% confidence interval (P_{eL}, P_{eU}) for P_e
1a	10,000	4	40×10^{-5}	$(18 \times 10^{-5}, 80 \times 10^{-5})$
1b	10,000	4	40×10^{-5}	$(18 \times 10^{-5}, 80 \times 10^{-5})$
1c	10,000	2	20×10^{-5}	$(5.3 \times 10^{-5}, 53 \times 10^{-5})$
1d	50,000	0	0	$(0, 4.6 \times 10^{-5})$
1e	50,000	2	4×10^{-5}	$(1.1 \times 10^{-5}, 11 \times 10^{-5})$
1f	10,000	5	50×10^{-5}	$(25 \times 10^{-5}, 91 \times 10^{-5})$
2a	10,000	1	10×10^{-5}	$(1.1 \times 10^{-5}, 39 \times 10^{-5})$
2b	50,000	0	0	$(0, 4.6 \times 10^{-5})$
3a	10,000	1	10×10^{-5}	$(1.1 \times 10^{-5}, 39 \times 10^{-5})$
3b	20,000	2	10×10^{-5}	$(2.7 \times 10^{-5}, 27 \times 10^{-5})$

Table IX: 90% confidence intervals for P_e as determined from the decoding simulations reported in Tables VI and VII.

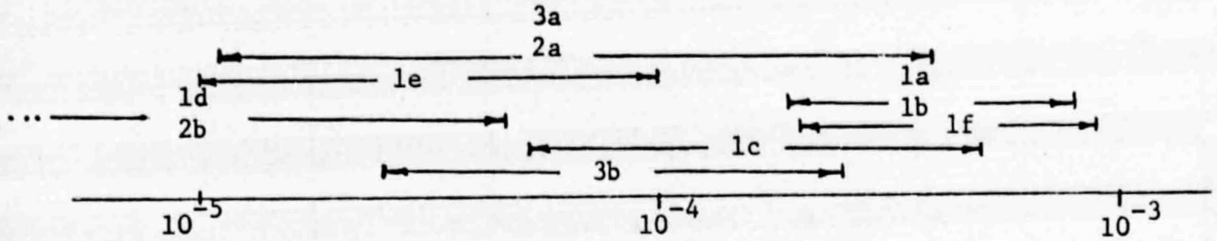

Fig. 4. Graphical portrayal of the 90% confidence intervals for P_e for all codes tested. (Codes shown by code number).

confidence we can assert that the frame error probability for codes 1d and 2b (neither of which yielded any errors in 50,000 decoded frames) is truly smaller than that of all the other codes, except codes 1e, 2a, 3a and 3b where our confidence in the superiority of 1d and 2b is somewhat less. It is a remarkable property of Poisson-distributed random variables that such a small number of error events (at most 5 for any code tested) can be so highly significant statistically.

It probably should be pointed out that, although 256 information bits are decoded in each frame so that there are 256 times as many bit decoding decisions as frame decoding decisions, one cannot assert greater statistical confidence in the observed decoding bit error probability than in the observed decoding frame error probability. The reason of course is that the decodings of bits within a frame are highly dependent so that one has no more independent bit decoding decisions from which to infer probabilities than one has independent frame decoding decisions.

IV. DISCUSSION AND RECOMMENDATIONS

A. Performance Summary for Each Code

1. Codes for the conventional option.

a. The MCQLI-24 code, (Code 1a).

Although this is the recommended K = 24 in the preliminary coding standard, its frame error probability P_e was tied for worst with 1JQLIODP-24 code among all codes tested, and significantly so (cf. Table IX and Fig. 4). Moreover, its computational performance was superior only to the QR-24 code (cf. Table VI). Perhaps, however, this is a good place to make the point that <u>only codes known to be good were tested</u>. One might say that this code is the worst of a good lot! Moreover, it does possess the desirable quick-look-in property which is worth some sacrifice in performance (but is also a feature of the distinctly better 2JQLIODP-24 code.)

Because this code is the K = 24 truncation of the MCQLI K = 48 code whose K = 32 truncation is an excellent code at that length [1], its poor K = 24 performance was somewhat unexpected. To explain this phenomenon, the free distance for all the distinct truncations of the MCQLI K = 48 code with $K \leq 28$ were computed with the results given in Table X. The value of $d_{free} = 23$ for the K = 32 code is also shown. From this table, one sees that there is indeed a "soft spot" around K = 24 in the truncations. In fact, the K = 22 code is quite likely superior in P_e to the K = 24 code because the same free distance is achieved within a smaller constraint length!

K	2	3	5	6	8	9	10	12	14	15	17	18	19	20	22	23	24	25	26	28	32
d_{free}	3	5	6	7	9	7	9	11	11	11	13	11	13	15	17	15	17	18	18	≥ 21	23

<u>Table X</u>: Free distance for truncation to constraint length K, for K = 32 and all $K \leq 28$ giving distinct truncations, of the MCQLI-48 code.

Beyond K = 24, the codes rapidly improve; the K = 32 code being already quite good. From the data in Table X, we conclude that it would probably be unwise to recommend truncations of the MCQLI-48 code with K < 32 in future coding standards.

b. The 1JQLIODP-24 (Code 1b).

This code gave the best computational performance of all codes tested (cf. Table VI). However, its superiority in this respect over the structurally-similar 2JQLIODP-24 code was very slight, but its inferiority to the latter in P_e (cf. Table IX and Fig. 4) is substantial. It thus appears unwise to recommend this code for any applications.

c. The 2JQLIODP-24 code, (Code 1c).

The three nonsystematic optimum distance profile codes tested (codes 1b, 1c and 2b) all gave virtually the same computational performance and were superior in this respect to all other codes tested (cf. Table VI), often quite significantly. The 2JQLIODP-24 code was also the best of the K = 24 codes with the quick-look-in feature (codes 1a, 1b, 1c and 1f) with respect to P_e (cf. Table IX and Fig. 4), and significantly so. This code is definitely the best of the K = 24 QLI codes studied.

d. The BJ-24 code, (Code 1d).

This code gave outstanding P_e performance (cf. Table IX and Fig. 4), as expected from its large d_{free} of 24. Its computational performance was slightly inferior to that of the ODP codes (cf. Table VI), but not significantly so. This code, unfortunately, does not have the quick-look-in property; but we have shown elsewhere [17] that it can be encoded almost as simply as a QLI code although, of course, the extraction of data by an encoder inverse from the hard-decisioned received sequences is much less reliable than for a QLI code. However, if one does not insist on the QLI property, this code is clearly the best of all known K = 24 codes.

e. The QR-24 code, (Code 1e).

As we noted earlier, this is actually a K = 23 code as both of its K = 24 generators end in a 0. This code is primarily interesting in that it is the only one of all the codes tested which was derived by an algebraic construction [11] as opposed to being found by a heuristically-guided computer search. Although its computational performance is distinctly inferior to the ODP codes and the BJ-24 code (cf. Table VI), its P_e performance was surprisingly good (cf. Table IX and Fig. 4). However, because it also lacks the QLI property (while being inferior in both P_e and computation) as does the BJ-24 code, this code should not be recommended for any K = 24 application. Its superiority in P_e over most of the codes tested does suggest, however, that there may well be merit in investing further effort toward finding algebraic approaches to convolutional code construction, an almost totally-neglected research area.

f. The OT-24 code, (Code 1f)

The rather uneven quality of the various truncations of the MCQLI-48 code (cf. Table X) suggested to us the advisability of devising a code which would truncate well everywhere. This led us to define the optimally-truncatable OT-K code as the QLI code obtained by the following algorithm:

```
Step 0: Set $a_0 = 1$, $b_0 = 1$, $a_1 = 1$ and $b_1 = 0$. Set $i = 2$, $d = 3$
        and $i_d = 1$.
Step 2: Set $a_i = b_i = 1$ and compute the free distance $d'$ of this code
        and the distance index $i_{d'}$.
Step 3: If $d' > d$ or if $d' = d$ and $i_{d'} < i_d$, replace $d$ by $d'$ and go
        to step 4. Otherwise, set $a_i = b_i = 0$.
Step 4: If $i = K-1$, stop. Otherwise, increase $i$ by 1 and go to step 2.
```

It should be clear that, for K' < K, the OT-K' code is the K' truncation of the OT-K code, so that these QLI codes are "nested" in the same sense as the MCQLI codes.

In Table XI, we give the free distance for all the distinct OT-K codes with $K \leq 24$. Comparison of Table XI with Table X shows indeed that the OT-K QLI codes are indeed superior to the MCQLI-K codes, but significantly so only for rather small K. However, if one desires a QLI code, in future coding standards, which truncates well for K < 24, the OT-24 code would be a good choice.

K	2	3	4	6	9	13	15	17	18	20	23	24
d_{free}	3	5	6	8	10	11	14	14	16	17	18	19

Table XI: Free distance for distinct truncations of the K = 24 optimally-truncatable code.

The penalty for both truncating well at all smaller constraint lengths and having the QLI property, however, is that one cannot do as well as an unconstrained code at some fixed large K. For instance, the OT-24 code has $d_{free} = 19$ compared to $d_{free} = 24$ for the BJ-24 code.

From Table III, we see that the OT-24 code is superior to the MCQLI-24 code in both distance profile and free distance. This explains the slightly better computational performance of the OT-24 code (cf. Table VI), but raises questions about the observed superiority in P_e for the MCQLI-24 code (cf. Table IX); however, as Figure 4 clearly shows, the observed difference in P_e for these two codes (1a and 1f) is not statistically significant.

2. Δ-undersized tail option.

a. The MCQLI-48 code, (Code 2a).

This code, which is the K = 48 code recommended in the preliminary coding standard [2], performed (as expected) slightly better computationally than its $K = 24$ truncation, code 1a (cf. Table VI). While its P_e performance is certainly good (cf. Table IX), it is somewhat surprising that even one frame was incorrectly decoded since the theoretical bounding argument [4] suggests that, because of the extra memory, P_e should be reduced by a factor of at least $L/K_E = 10.7$ over its $K = 24$ truncation for which only 4 frame decoding errors were observed. The explanation is provided by Table VIII; the actual error probability of this $K = 48$ code, for which only $x = 1$ frame decoding error was observed, has a significant probability of being smaller, by at least the theoretically-expected order of magnitude, than that observed in the simulation. However, because this code is still inferior in all respects to the JQLIODP-48 code to be described next, it would be wise to replace the MCQLI-48 code in the standard by the JQLIODP-48 code if the disruption would not be too great.

b. The JQLIODP-48 code, (Code 2b).

This code performed superbly in both computation (cf. Table VI) and in error probability (cf. Table IX and Fig. 4). The excellent computational performance, which was the best of all codes tested, results of course from its optimum distance profile over the full $K = 48$ branches. The extremely low observed P_e, which tied with that of the BJ-24 code for the best of the codes tested, is explained by the expected reduction by at least a factor of $L/K_E = 10.7$ over the P_e of its $K = 24$ truncation because of the additional memory of the $K = 48$ code.

This is probably the appropriate place to point out that a truncation of an ODP code is also again an ODP code. Thus, all truncations of the JQLIODP-48 code would perform well computationally compared to other codes at their constraint length. The JQLIODP-48 code would be a good choice for the truncatable $K = 48$ quick-look-in code in future coding standards.

3. Systematic partial tail-suppression option.

a. The BLLF-47 code, (Code 3a).

This code, which is the K = 47 truncation of the K = 48 systematic code given in the preliminary coding standard [2], had nearly as good a computational performance as the JSODP-47 code (cf. Table VI) to which it most closely compares, and is also nearly as good in error probability (cf. Table IX and Fig. 4). In fact, as can be seen from Table III, the distance profile of this code is only slightly inferior to the ODP code and, moreover, this code has a larger free distance than the ODP code (19 vs 17). [The paradoxically slightly better P_e of the ODP code derives most likely from the fact that it reaches its free distance much more quickly ($i_{17} = 46$ compared to $i_{17} = 53$ and $i_{19} = 59$ for the BLLF-47 code).] The good P_e performance of both codes confirms the conjecture, noted in Section II.B.3 above, that a good systematic code in option 3 performs about as well in P_e as does a good nonsystematic code in option 1, for the same K_E.

The "systematic" property is a highly desirable one for many purposes (such as the absolutely minimal error probability in recovering data by an encoder inverse from the hard-decisioned received sequences, and the absolutely simplest "encoder inverse".) The disadvantages in option 3 compared to option 1 are the necessity to double encoder speed in the tail, and the small amount of extra computation required because of the extra branches in the tail.

The rather slight inferiority of the BLLF-47 code to the JSODP-47 code suggests that there would be only slight advantage in replacing this code by the better JSODP-47 code in future coding standards.

b. The JSODP-47 code, (Code 3b).

As just discussed, this is the best available systematic code with respect to both computation and decoding error probability. Its ODP structure guarantees that its truncations will also be ODP, and hence good. If the changeover from the BLLF-47 code would not be too disruptive, *it would be the best available choice for the* K = 48 *truncatable systematic code in future coding standards*.

B. Recommendations for the K_E = 24 code

In the performance summaries for the K_E codes tested, we have already made a number of recommendations regarding the selection of a K_E = 24 code for the deep-space channel and, more generally, for the selection of rate R = 1/2 convolutional codes for the deep-space channel. We now particularize our recommendations to the choice of a K_E = 24 code for the IUE spacecraft.

1. The best choice for the K_E = 24 code would be the JQLIODP-48 code (code 2b) used in the (Δ = 24)-undersized tail option (option 2).

We favor this choice because (a) this code has the best computational behavior of all the codes tested, and a decoding error probability tied with the BJ-24 for the best, (b) this same code and encoder could be advantageously used at K_E = 32, or any $K_E \leq 48$, in later deep-space missions so that its adoption now would be a healthy move toward encoder standardization, (c) the lone drawback of this code (or any code used in option 2), namely that the encoding shift-register must have all its contents set to zero upon completion of encoding of the frame, appears to be very slight, and (4) this code has the desirable quick-look-in (QLI) property.

2. The next best choice would be the BJ-24 code (code 1d), provided the QLI feature is not considered essential.

If the conventional option (option 1) is chosen for the IUE convolutional coding system, this is definitely the best code to use. Its decoding error probability was equal to the best of the other codes tested, and its computational performance was nearly as good as any other. The only drawback for this code is that it does not have the quick-look-in feature.

3. If the QLI feature is considered essential, the next best choice (after the JQLIODP-48 code) would be the 2JQLIODP-24 code (code 1c) or the K = 24 truncation of the JQLIODP-48 code, although the MCQLI-24 code (code 1a) would be acceptable.

If both the conventional option (option 1) and the QLI feature are chosen for the IUE convolutional coding system, the 2JQLIODP-24 code offers the twin advantages of near-optimal computational performance and near-minimal decoding error probability. Although the MCQLI-24 code was somewhat inferior in both of these respects, the fact that various spacecraft have already used truncations of the MCQLI-48 code, so that proven encoder hardware is presently available, may make this code the practical choice for the IUE spacecraft. However, if the JQLIODP-48 code (code 2b) displaces the MCQLI-48 code (code 2a) as the truncatable QLI code in future coding standards (in keeping with our recommendation in Section IV.A.2.b), then the K = 24 truncation of this code should be used in preference to the MCQLI-24 code.

POSTSCRIPT

Following just upon completion of this report, the author received an advanced copy of the paper "Further Results on Binary Convolutional Codes with an Optimum Distance Profile" by R. Johannesson and E. Paaske to be presented at the IEEE International Symposium on Information Theory in Ronneby, Sweden, June 21-24, 1976. This paper reports their newly discovered K = 24 code, namely

G1 = 55346125 (octal)

G2 = 75744143 (octal),

which has both an optimum distance profile and a free distance $d_\infty = 25$ exceeding that of any previously known K = 24 code (including the BJ-24 code which has $d_\infty = 24$.) This code is thus undoubtedly superior both in computation and in error probability to the BJ-24, and should be recommended in place of the latter code wherever the latter code is recommended in this report. Simulations to confirm this superiority are now in progress and the results will be reported as soon as they become available.

REFERENCES

[1] J. L. Massey, "A Recommended R = 1/2, K = 32, Quick-Look-In Convolutional Code for NASA Use," Tech. Rept. No. EE-751, Department of Electrical Engineering, University of Notre Dame, Notre Dame, Ind., April 28, 1975.

[2] "Aerospace Data Systems Standards, Part 7: Coding Standards, Standard 7.1, Inner Convolutional and Block Error Control Coding Standard," Preliminary Inner Error Control Coding Standard 1975-05-27, Goddard Space Flight Center.

[3] J. L. Massey and D. J. Costello, Jr., "Nonsystematic Convolutional Codes for Sequential Decoding in Space Applications," IEEE Trans. on Comm. Tech., Vol. COM-19, pp. 806-813, October 1971.

[4] J. L. Massey, "Error Bounds for Tree Codes, Trellis Codes, and Convolutional Codes with Encoding and Decoding Procedures," Lectures presented at Summer School on Coding and Complexity, Centre International des Sciences Mechanique (C.I.S.M.), Udine, Italy, July 15-26, 1974, to appear in a future volume in the C.I.S.M. Lecture Series.

[5] R. Johannesson, "On the Error Probability of General Tree and Trellis Codes with Applications to Sequential Decoding," Tech. Rept. No. EE-7316, Department of Electrical Engineering, University of Notre Dame, Notre Dame, Ind., December 1973.

[6] J. L. Massey, "Semi-Annual Status Report, Coordinated Design of Coding and Modulation Systems," NASA Grant NSG-5025, Department of Electrical Engineering, University of Notre Dame, Notre Dame, Ind., June 25, 1975.

[7] P. R. Chevillat and D. J. Costello, Jr., "Distance and Computation in Sequential Decoding," Tech. Rept., Illinois Inst. of Tech., Chicago, Illinois, 1975.

[8] R. Johannesson, "Some Long, Rate One-Half, Binary Convolutional Codes with an Optimum Distance Profile and the Systematic versus Nonsystematic Code Question," Tech. Rept. No. EE756, Department of Electrical Engineering, University of Notre Dame, Notre Dame, Ind., August 19, 1975.

[9] R. Johannesson, "Robustly Optimal Rate One-Half Binary Convolutional Codes," IEEE Trans. Info. Th., Vol. IT-21, pp. 464-468, July 1975.

[10] L. R. Bahl and F. Jelinek, "Rate 1/2 Convolutional Codes with Complementary Generators," IEEE Trans. Info. Th., Vol. IT-17, pp. 718-727, November 1971.

[11] J. L. Massey, D. J. Costello, Jr., and J. Justesen, "Polynomial Weights and Code Constructions," IEEE Trans. Info. Th., Vol. IT-19, pp. 101-110, January 1973.

[12] J. J. Bussgang, "Some Properties of Binary Convolutional Code Generators," IEEE Trans. Info. Th., Vol. IT-11, pp. 90-100, January 1965.

[13] S. Lin and H. Lyne, "Some Results on Binary Convolutional Code Generators," IEEE Trans. Info. Th., Vol. IT-13, pp. 134-139, January 1967.

[14] G. D. Forney, Jr., "Convolutional Codes I: Algebraic Structure," IEEE Trans. Info. Th., Vol. IT-16, pp. 720-738, Nov. 1970.

[15] I. M. Jacobs, "Sequential Decoding for Efficient Communication from Deep Space," IEEE Trans. Comm. Tech., Vol. COM-15, pp. 492-501, August 1967.

[16] R. M. Fano, "A Heuristic Discussion of Probabilistic Decoding," IEEE Trans. Info. Th., Vol. IT-19, pp. 64-73, April 1963.

[17] J. L. Massey, "Quarterly Progress Report: Convolutional Coding Techniques for Data Protection," NASA Grant NGL-15-004-026, Department of Electrical Engineering, University of Notre Dame, Notre Dame, Ind., March 2, 1970.