Simulations for Full Unit-Memory and Partial Unit-Memory Convolutional Codes With Real-Time Minimal-Byte-Error Probability Decoding Algorithm

Q. D. Vo
Communications Systems Research Section

This report describes a program which was written to simulate Real-Time Minimal-Byte-Error Probability (RTMBEP) decoding of full unit-memory (FUM) convolutional codes on a 3-bit quantized AWGN channel. This program was used to compute the symbol-error probability of FUM codes and to determine the signal-to-noise (SNR) required to achieve a bit error rate (BER) of 10^{-6} for corresponding concatenated systems. A (6, 6/30) FUM code, 6-bit Reed-Solomon code combination was found to achieve the required BER at a SNR of 1.886 dB. The RTMBEP algorithm was then modified for decoding partial unit-memory (PUM) convolutional codes. A simulation program was also written to simulate the symbol-error probability of these codes.

I. Introduction

To achieve reliable communications over a very noisy channel with relatively small coding complexity, concatenated coding systems using a convolutional code as the inner code and a Reed-Solomon code as the outer code are usually suggested (Fig. 1). In our baseline system, the inner code is a (7, 1/2) convolutional code with Viterbi decoding (Ref. 1) while the outer code is an 8-bit Reed-Solomon code. The overall system achieves a bit error rate (BER) of 10^{-6} with a signal-to-noise ratio (SNR) of 2.53 dB. Our goal is to obtain a 2-dB improvement in SNR (i.e., to find an inner code with a corresponding decoding algorithm to achieve a BER of 10^{-6} at a SNR of 0.53 dB). It was found by Lin-nan Lee (Refs. 2 and 3) that the use of byte-oriented full unit-memory (FUM) convolutional codes in conjunction with real-time minimal-byte-error probability (RTMBEP) decoding algorithm provides an improvement of about 0.3 dB compared to regular bit-oriented convolutional codes with same state complexity. In this report, we simulated the RTMBEP decoding of FUM codes on a 3-bit quantized additive white Gaussian noise (AWGN) channel (as shown by the dashed box in Fig. 1). This simulation program is used as a tool to compute symbol-error probability for FUM codes. We then modified the RTMBEP algorithm for decoding a subclass of FUM codes called partial-unit-memory (PUM) codes (Ref. 4). A simulation program was also written for this modified algorithm to simulate symbol-error probability of PUM codes.

II. Simulation of FUM Codes With RTMBEP Decoding Algorithm

A general (*lo, ko/no*) unit-memory convolutional encoder is shown in Fig. 2. Here, we have *ko* bits of input to be encoded, *lo* bits of delayed input, and *no* bits of encoded output. For $lo = ko$ we have FUM convolutional codes. For $lo < ko$ we have PUM convolutional codes.

A (*ko, ko/no*) FUM code may be written as

$$\mathbf{b}_t = \mathbf{a}_t G_o + \mathbf{a}_{t-1} G_1; \quad t = 1, 2, \ldots$$

where $\mathbf{a}_t$ is the k_o-bit byte of information to be encoded at time t, $\mathbf{b}_t$ is the corresponding encoded no-bit byte, and G_o and G_1 are $ko \times no$ encoding matrices (by convention, $\mathbf{a}_o = \mathbf{0}$). The sequence $\{\mathbf{b}_t\}$ is sent over a 3-bit quantized AWGN channel and the corresponding sequence $\mathbf{r}_t$ is received. For convenience, we denote $\mathbf{a}_{[t,t^1]}$ to be $[\mathbf{a}_t, \mathbf{a}_{t+1}, \ldots, \mathbf{a}_{t^1}]$ and similarly for $\mathbf{b}_{[t,\, t^1]}, \mathbf{r}_{[t,\, t^1]}$.

The RTMBEP decoding rule (Ref. 3) chooses its estimate $\hat{\mathbf{a}}_t$ to be the value of $\mathbf{a}_t$ which maximizes $P(\mathbf{a}_t|\mathbf{r}_{[1,t+\Delta]})$ where $\mathbf{r}_{[1,\,t+\Delta]}$ is the observed sequence with delay Δ. This algorithm is based on the facts that the channel is memoryless and that the code has unit memory to develop a recursive method for computing $P(\mathbf{a}_t|\mathbf{r}_{[1,\,t+\Delta]})$

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

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

$$P(\mathbf{a}_t, \mathbf{r}_{[1,t]}) = \sum_{\mathbf{a}_{t-1}} P(\mathbf{a}_{t-1}, \mathbf{r}_{[1,t-1]})$$

$$\times P(\mathbf{a}_t, \mathbf{r}_t|\mathbf{a}_{t-1}, \mathbf{r}_{[1,t-1]})$$

$$= 2^{-ko} \sum_{\mathbf{a}_{t-1}} P(\mathbf{a}_{t-1}, \mathbf{r}_{[1,t-1]})\, P(\mathbf{r}_t|\mathbf{b}_t)$$

$$P(\mathbf{r}_{[t+i,t+\Delta]}|\mathbf{a}_{t+i-1}) = \sum_{\mathbf{a}_{t+1}} P(\mathbf{r}_{[t+i+1,t+\Delta]}|\mathbf{a}_{t+i})$$

$$\times P(\mathbf{a}_{t+i}, \mathbf{r}_{t+i}|\mathbf{a}_{t+i-1})$$

$$= 2^{-ko} \sum_{\mathbf{a}_{t+i}} P(\mathbf{r}_{[t+i+1,t+\Delta]}|\mathbf{a}_{t+i})$$

$$\times P(\mathbf{r}_{t+i}|\mathbf{b}_{t+i})$$

The last recursion is initialized with $i = \Delta$

$$P(\mathbf{r}_{t+\Delta}|\mathbf{a}_{t+\Delta-1}) = \sum_{\mathbf{a}_{t+\Delta}} P(\mathbf{r}_{t+\Delta}, \mathbf{a}_{t+\Delta}|\mathbf{a}_{t+\Delta-1})$$

$$= 2^{-ko} \sum_{\mathbf{a}_{t+\Delta}} P(\mathbf{r}_{t+\Delta}|\mathbf{b}_{t+\Delta})$$

and computed backward with $i = \Delta - 1, \ldots, 1$. Here, we assume all information sequences are equally likely (i.e., $P(\mathbf{a}_t) = 2^{-ko}$). We modified the algorithm slightly so that it is computer compatible. Let

$$f(\mathbf{a}_t) = P(\mathbf{a}_t, \mathbf{r}_{[1,t]})$$

$$h(\mathbf{a}_{t+i-1}) = P(\mathbf{r}_{[t+i,\,t+\Delta]}|\mathbf{a}_{t+i-1})$$

At time $t = 0$ (initialization) the whole system is set up as follows:

(1) Set up coder matrix which gives $\mathbf{b}_t$ for each $\mathbf{a}_t$ and $\mathbf{a}_{t-1}$

(2) Simulate $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_\Delta$ on a 3-bit quantized AWGN channel and compute probability matrices $P(\mathbf{r}_i|\mathbf{b}_i)$

(3) Set $f(\mathbf{a}_o = \mathbf{0}) = 1$ and $f(\mathbf{a}_o \neq \mathbf{0}) = 0$

For each time t (main loop; $t = 1, 2, \ldots$) the following steps are taken:

Step 0: Simulate $\mathbf{r}_{t+\Delta}$ and compute $P(\mathbf{r}_{t+\Delta}|\mathbf{b}_{t+\Delta})$ matrix

Step 1: Compute

$$f(\mathbf{a}_t) = \sum_{\mathbf{a}_{t-1}} f(\mathbf{a}_{t-1})\, P(\mathbf{r}_t|\mathbf{b}_t)$$

and normalize

$$f(\mathbf{a}_t) = f(\mathbf{a}_t)/f(\mathbf{0})$$

Note that the normalization is needed since, for large t, $f(\mathbf{a}_t)$ tends to go to zero.

Step 2: Set

$$h(\mathbf{a}_{t+\Delta-1}) = \sum_{\mathbf{a}_{t+\Delta}} P(\mathbf{r}_{t+\Delta} \mid b_{t+\Delta})$$

and normalize

$$h(\mathbf{a}_{t+\Delta-1}) = \frac{h(\mathbf{a}_{t+\Delta-1})}{h(\mathbf{0})}$$

Step 3: For $i = \Delta - 1$ to $i = 1$, compute

$$h(\mathbf{a}_{t+i-1}) = \sum_{\mathbf{a}_{t+i}} h(\mathbf{a}_{t+i})\, P(\mathbf{r}_{t+i} \mid \mathbf{b}_{t+i})$$

and normalize

$$h(\mathbf{a}_{t+i-1}) = \frac{h(\mathbf{a}_{t+i-1})}{h(\mathbf{0})}$$

Step 4: Choose the estimate $\hat{\mathbf{a}}_t$ such that

$$f(\hat{\mathbf{a}}_t)\, h(\hat{\mathbf{a}}_t) \geq f(\mathbf{a}_t)\, h(\mathbf{a}_t) \quad ; \quad \text{for all } \mathbf{a}_t\text{'s}$$

and compute the probability

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

Note that this probability could be used as a reliability indicator for soft decision (including erasure) decoding of the convolutional code. Since large Δ requires more computation, it is desirable to keep Δ as small as possible. It is shown by Lee (Ref. 3) that $\Delta = 8$ gives virtually the same performance as $\Delta = \infty$. We checked the simulation with several small FUM convolutional codes and ran it for the maximal d_{free} (6, 6/18), (6, 6/24) codes found by Lin-nan Lee (Ref. 2) and a (6, 6/30) code found by Pil J. Lee. These codes are given in Table 1 in hexadecimal format. Simulation results are given in Table 2 for different levels of signal-to-noise ratio, E_b'/No (E_b' is the input bit energy at the inner encoder and No is the one-sided noise power spectral density.) The symbol (byte) error probability P_s, which is calculated based on 8000-byte decoding simulation, is plotted versus E_b'/No in Fig. 3 for the three FUM codes. These FUM codes are concatenated with various 6-bit rate k/n ($n = 2^6 - 1 = 63$) Reed-Solomon codes. Given that the interleaving is perfect, the overall bit-error probability can be computed as

$$P_b = \frac{n+1}{2n} \sum_{i=t+1}^{n} \frac{i}{n} \binom{n}{i} p^i (1-p)^{n-i}$$

where t is the error correcting capability ($t = n - k/2$) and p is the symbol-error probability at the inner decoder output. The required p to achieve a BER of 10^{-6} is shown in Table 3 for several 6-bit Reed-Solomon codes. From Fig. 2, the requirement in p specifies a requirement in E_b'/No and E_b/No ($E_b/No = nE_b'/kNo$) as also shown in Table 3. In Fig. 4, we plot the signal-to-noise ratio (E_b/No) required to achieve a BER of 10^{-6} versus the Reed-Solomon code rate for the three FUM codes. An improvement of 0.64 dB is obtained with the (6, 6/30) code compared to the baseline (7, 1/2) convolutional code, Reed-Solomon code combination. To achieve a 2 dB improvement in required E_b/No, FUM codes with higher state space complexity will be required. Unfortunately, it takes a long time to search for and to simulate such codes. For example, it takes 16 hours to simulate one point for a (7, 7/28) FUM code on a fast computer. Therefore, we consider next a subclass of FUM codes called partial-unit-memory (PUM) codes (Ref. 4) which may provide similar performance with less complexity.

III. Simulation of PUM Codes With Modified RTMBEP Decoding Algorithm

Let $\mathbf{a}_t = \tilde{\mathbf{a}}_t : \hat{\mathbf{a}}_t$ (the colon denotes concatenation of $k_o - l_o$-bit byte $\tilde{\mathbf{a}}_t$ with l_o-bit byte $\hat{\mathbf{a}}_t$) be the k_o-bit byte of information to be encoded at time t and $\mathbf{b}_t$ be the corresponding encoded n_o-bit byte. A (l_o, k_o/n_o) PUM code may be written as (see Fig. 2)

$$\mathbf{b}_t = \mathbf{a}_t G_o + \hat{\mathbf{a}}_{t-1} G_1; \quad t = 1, 2, 3, \ldots$$

where G_0 and G_1 are encoding matrices with dimensions $k_o \times n_o$ and $l_o \times n_o$, respectively. The state complexity is defined to be l_o, which is the number of delay cells required to realize the encoder. Let $\tilde{G}_0$ be the first $k_o - l_o$ rows of G_0 and $\hat{G}_0$ be the remaining l_o rows, then

$$\tilde{\mathbf{b}}_t = \tilde{\mathbf{a}}_t G_o + \hat{\mathbf{a}}_t \hat{G}_o + \hat{\mathbf{a}}_{t-1} G_1$$

We want to find the estimate $\mathbf{a}_t^o$ which maximizes $P(\mathbf{a}_t \mid \mathbf{r}_{[1,\, t+\Delta]})$ where $\mathbf{r}_{[1,\, t+\Delta]}$ is the received sequence with delay Δ. As in Section II, we have

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

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

$$P(\mathbf{a}_t, \mathbf{r}_{[1,t]}) = 2^{-k_o} \sum_{\mathbf{a}_{t-1}} P(\mathbf{a}_{t-1}, \mathbf{r}_{[1,t-1]}) P(\mathbf{r}_t \mid \mathbf{b}_t)$$

However, taking into account that $\mathbf{b}_t$ just depends on $\hat{\mathbf{a}}_{t-1}$, we can write

$$P(\mathbf{a}_t, \mathbf{r}_{[1,t]}) = 2^{-k_o} \sum_{\hat{\mathbf{a}}_{t-1}} P(\mathbf{r}_t \mid \mathbf{b}_t) \sum_{\tilde{\mathbf{a}}_{t-1}} P(\mathbf{a}_{t-1}, \mathbf{r}_{[1,t-1]})$$

$$= 2^{-k_o} \sum_{\hat{\mathbf{a}}_{t-1}} P(\mathbf{r}_t \mid \mathbf{b}_t) P(\hat{\mathbf{a}}_{t-1}, \mathbf{r}_{1,t-1]})$$

where

$$P(\hat{\mathbf{a}}_{t-1}, \mathbf{r}_{[1,t-1]}) = \sum_{\tilde{\mathbf{a}}_{t-1}} P(\mathbf{a}_{t-1}, \mathbf{r}_{[1,t-1]})$$

could be computed outside of the $\mathbf{a}_t$ loop. Since the dimension of $\hat{\mathbf{a}}_{t-1}$ (i.e., l_o) is less than the dimension of $\mathbf{a}_{t-1}$ (i.e., k_o), this results in a saving of computation time. For example, with $k_o = 6$, $l_o = 6$ (regular FUM codes) it requires 4096 multiplications compared to 2048 multiplications for the case $k_o = 6$, $l_o = 5$ or 1024 multiplications for the case $k_o = 6$, $l_o = 4$.

On the other hand,

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

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

since $\mathbf{r}_{[t+1,t+\Delta]}$ is independent of $\tilde{\mathbf{a}}_t$. A backward recursion is developed to compute $P(\mathbf{r}_{[t+1,t+\Delta]} \mid \hat{\mathbf{a}}_t)$:

$$P(\mathbf{r}_{[t+i,t+\Delta]} \mid \hat{\mathbf{a}}_{t+i-1}) = \sum_{\mathbf{a}_{t+i}} P(\mathbf{a}_{t+i}, \mathbf{r}_{[t+i,t+\Delta]} \mid \hat{\mathbf{a}}_{t+i-1})$$

$$= \sum_{\mathbf{a}_{t+i}} P(\mathbf{r}_{[t+i+1,t+\Delta]} \mid \hat{\mathbf{a}}_{t+i}) \times P(\mathbf{a}_{t+i}, \mathbf{r}_{t+i} \mid \hat{\mathbf{a}}_{t+i-1})$$

$$= \sum_{\mathbf{a}_{t+i}} P(\mathbf{r}_{[t+i-1,t+\Delta]} \mid \hat{\mathbf{a}}_{t+i}) \times P(\mathbf{r}_{t+i} \mid \mathbf{a}_{t+i}, \hat{\mathbf{a}}_{t+i-1}) P(\mathbf{a}_{t+i})$$

$$= 2^{-k_o} \sum_{\mathbf{a}_{t+i}} P(\mathbf{r}_{[t+i-1,t+\Delta]} \mid \hat{\mathbf{a}}_{t+i}) \times P(\mathbf{r}_{t+i} \mid \mathbf{b}_{t+i})$$

$$= 2^{-k_o} \sum_{\hat{\mathbf{a}}_{t+i}} P(\mathbf{r}_{[t+i-1,t+\Delta]} \mid \hat{\mathbf{a}}_{t+i}) \times \sum_{\tilde{\mathbf{a}}_{t+1}} P(\mathbf{r}_{t+1} \mid \mathbf{b}_{t+1})$$

This recursion is initialized with $i = \Delta$:

$$P(\mathbf{r}_{t+\Delta} \mid \hat{\mathbf{a}}_{t+\Delta-1}) = 2^{-k_o} \sum_{\mathbf{a}_{t+\Delta}} P(\mathbf{r}_{t+\Delta} \mid \mathbf{b}_{t+\Delta})$$

For $k_o = 6$, $l_o = 5$ this recusion requires 1048 multiplications compared to 4096 multiplications of the regular FUM codes ($k_o = 6$, $l_o = 6$). Let

$$f(\mathbf{a}_t) = P(\mathbf{a}_t, \mathbf{r}_{[1,t]})$$

$$h(\hat{\mathbf{a}}_{t+i-1}) = P(\mathbf{r}_{[t+i,t+\Delta]} \mid \hat{\mathbf{a}}_{t+i-1})$$

Then the modified algorithm for PUM codes is summarized as follows: At time $t = 0$, the system is initalized as follows:

(1) Set up coder matrix which gives $\mathbf{b}_t$ for each $\mathbf{a}_t$ and $\hat{\mathbf{a}}_{t-1}$

(2) Simulate $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_\Delta$ on a 3-bit quantized AWGN channel and compute probability matrices $P(\mathbf{r}_i|\mathbf{b}_i)$

(3) Set $f(\mathbf{a}_o = \mathbf{0}) = 1$ and $f(\mathbf{a}_o \neq \mathbf{0}) = 0$

At time t ($t = 1, 2, \ldots$) the following steps are taken:

Step 0: Simulate $\mathbf{r}_{t+\Delta}$ and compute $P(\mathbf{r}_{t+\Delta} \mid \mathbf{b}_{t+\Delta})$ matrix.

Step 1: Compute

$$f(\hat{\mathbf{a}}_{t-1}) = \sum_{\tilde{\mathbf{a}}_{t-1}} f(\mathbf{a}_{t-1})$$

For each $\mathbf{a}_t$, compute

$$f(\mathbf{a}_t) = \sum_{\hat{\mathbf{a}}_{t-1}} f(\hat{\mathbf{a}}_{t-1})\, P(\mathbf{r}_t \mid \mathbf{b}_t)$$

$$f(\mathbf{a}_t) = f(\mathbf{a}_t)/f(\mathbf{0})$$

Step 2: Set

$$h(\hat{\mathbf{a}}_{t+\Delta-1}) = \sum_{\mathbf{a}_{t+\Delta}} P(\mathbf{r}_{t+\Delta} \mid \mathbf{b}_{t+\Delta})$$

$$h(\hat{\mathbf{a}}_{t+\Delta-1}) = \frac{h(\hat{\mathbf{a}}_{t+\Delta-1})}{h(\mathbf{0})}$$

Step 3: For $i = \Delta - 1$ to $i = 1$, compute for each $\hat{\mathbf{a}}_{t+i-1}$

$$h(\hat{\mathbf{a}}_{t+i-1}) = \sum_{\hat{\mathbf{a}}_{t+i}} h(\hat{\mathbf{a}}_{t+i}) \sum_{\tilde{\mathbf{a}}_{t+i}} P(\mathbf{r}_{t+i} \mid \mathbf{b}_{t+i})$$

$$h(\hat{\mathbf{a}}_{t+i-1}) = \frac{h(\hat{\mathbf{a}}_{t+i-1})}{h(\mathbf{0})}$$

Step 4: Choose the estimate $\mathbf{a}_t^o$ such that (recall that $h(\mathbf{a}_t) = h(\hat{\mathbf{a}}_t)$)

$$f(\mathbf{a}_t^o)\, h(\mathbf{a}_t^o) \geq f(\mathbf{a}_t)\, h(\mathbf{a}_t)$$

for all $\mathbf{a}_t$'s and compute as a reliability indicator

$$P(\mathbf{a}_t = \mathbf{a}_t^o \mid \mathbf{r}_{[1,t+\Delta]}) = \frac{f(\mathbf{a}_t^o)\, h(\mathbf{a}_t^o)}{\sum_{\mathbf{a}_t} f(\mathbf{a}_t)\, h(\mathbf{a}_t)}$$

Again, this reliability indicator could be used for soft decision (including erasure) decoding of the convolutional code.

We checked the simulation with some small PUM codes given in Table 4. The simulated symbol-error probability for each code is also given in Table 4. Good PUM codes with higher state complexity are needed to achieve the required performance.

IV. Summary

A program is written to simulate the RTMBEP decoding of FUM codes on a 3-bit quantized AWGN channel. This program is used to compute the symbol-error probability for various FUM codes. From this, the SNR required to achieve a BER of 10^{-6} can be determined for a concatenated system. A 6, 6/30) FUM code is found to achieve the required BER at a SNR of 1.886 dB. Finally the RTMBEP algorithm is modified for decoding a subclass of FUM codes called PUM codes. A simulation program is also written to simulate symbol-error probability of PUM codes.

References

1. Viterbi, A. J., and Omura, J. K., *Principles of Digital Communication and Coding*, McGraw-Hill, N.Y., 1979.
2. Lee, Lin-nan, "Short Unit-Memory Byte-Oriented Binary Convolutional Codes Having Maximal Free Distance," *IEEE Transactions on Information Theory*, Vol. IT-22, pp. 349-352, May 1976.
3. Lee, Lin-nan, *Concatenated Coding Systems Employing Unit-Memory Convolutional Codes and Symbol-Oriented Optimal Decoding Algorithms*, Ph.D. Dissertation, Department of Electrical Engineering, University of Notre Dame, IN., May 1976.
4. Lauer, G. S., "Some Optimal Partial Unit Memory Codes," *IEEE Transactions on Information Theory*, Vol. IT-25, pp. 240-243, March 1979.

Table 1. Some UM convolutional codes

Rate	no	ko	G_0	G_1	d_{Free}
1/3	18	6	38D30	031CB	16
			1C698	06396	
			0E34C	0C72C	
			07986	18E19	
			234C3	30C72	
			31A61	218E5	
1/4	24	6	83EEB0	3C52D3	24
			41F758	78A5A6	
			22FB8C	F14B0D	
			1375C6	E6865A	
			0BBAC3	CD1CB4	
			07DD61	9E2969	
1/5	30	6	20FBAC1C	0F14B4C1	29
			107DD60E	1E296982	
			08BEE307	3C52C344	
			04DD71A3	39A19688	
			02EEB0F1	33472D10	
			01F75878	278A5A60	

Table 2. Symbol-error probability for codes in Table 1

E'_b/No (dB)	0.50	0.75	1.00	1.25	1.50
(6,6/18) P_S		0.05925	0.0355	0.02125	0.01275
(6,6/24) P_S	0.060875	0.03775	0.02275	0.01225	0.00675
(6,6/30) P_S		0.024625	0.0125	0.007875	0.005

Table 3. Signal-to-noise ratio requirement to achieve a BER of 10^{-6}

k	45	47	49	51	53	55	57
k/n	0.7143	0.7460	0.7778	0.8095	0.8413	0.8730	0.9048
(dB)	-1.461	-1.272	-1.091	-0.918	-0.751	-0.590	-0.435
P	0.02875	0.02294	0.01764	0.01290	0.00879	0.00538	0.00277
(6,6/18)							
E'_b/No (dB)	1.104	1.212	1.342	1.494	1.681	1.920	
E_b/No (dB)	2.565	2.484	2.433	2.412	2.432	2.510	
(6,6/24)							
E'_b/No (dB)	0.887	0.995	1.109	1.233	1.388	1.587	
E_b/No (dB)	2.348	2.267	2.200	2.151	2.139	2.177	
(6,6/30)							
E'_b/No (dB)		0.758	0.846	0.968	1.169	1.392	
E_b/No (dB)		2.030	1.937	1.886	1.920	1.982	

Table 4. Simulated symbol-error probability for some PUM codes

Codes	G_0	G_1	E'_b/No (dB) and P_S				
(1,2/4)	F	6	2 dB		4 dB		6 dB
	3		0.04036		0.00665		0.00039
(3,4/8)	FF	17	2.5 dB	3.0 dB	3.5 dB	4.0 dB	4.5 dB
	OF	2D	0.00946	0.00376	0.00125	0.00034	0.00005
	33	8B					
	55						

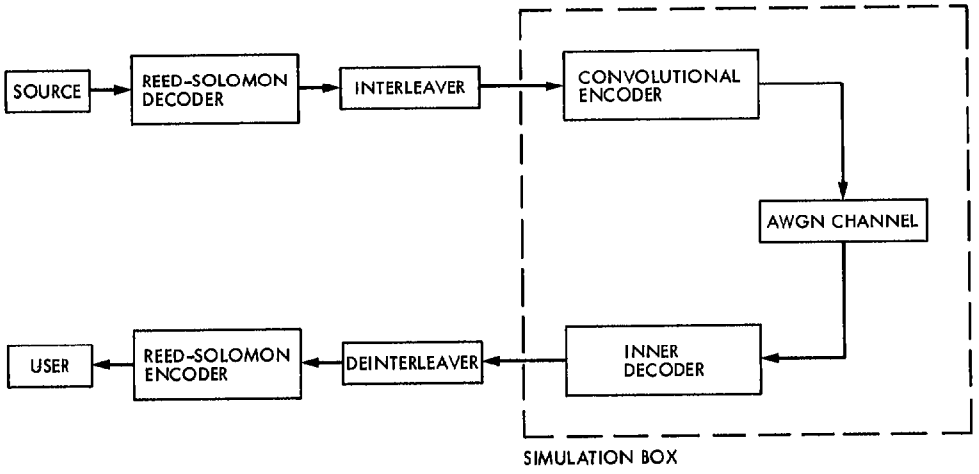

Fig. 1. A concatenated coding system

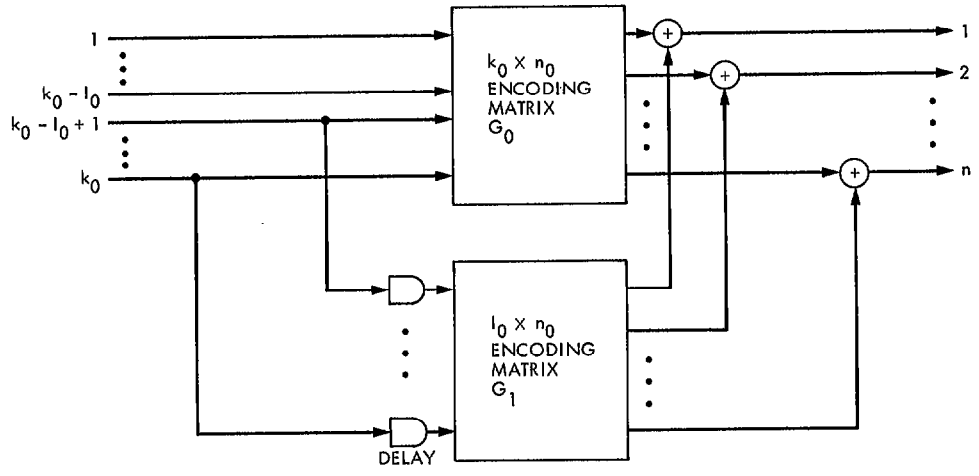

Fig. 2. A general (I_o, *ko/no*) unit-memory convolutional encoder

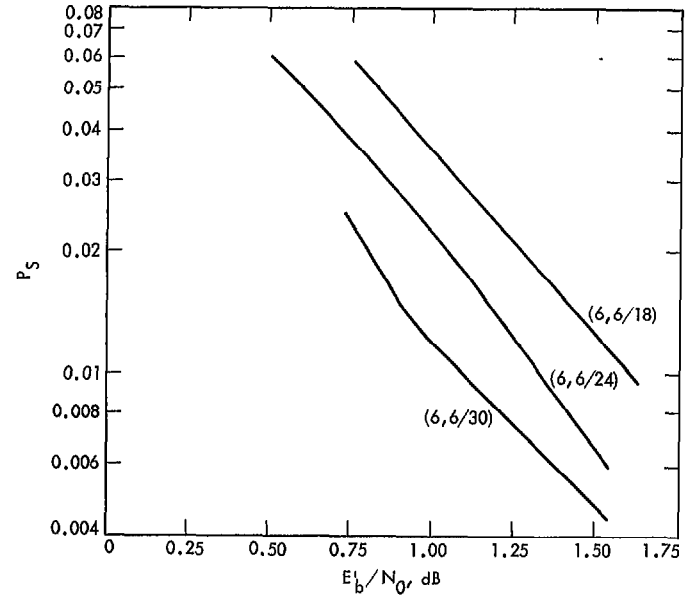

Fig. 3. Symbol error probability for three FUM codes

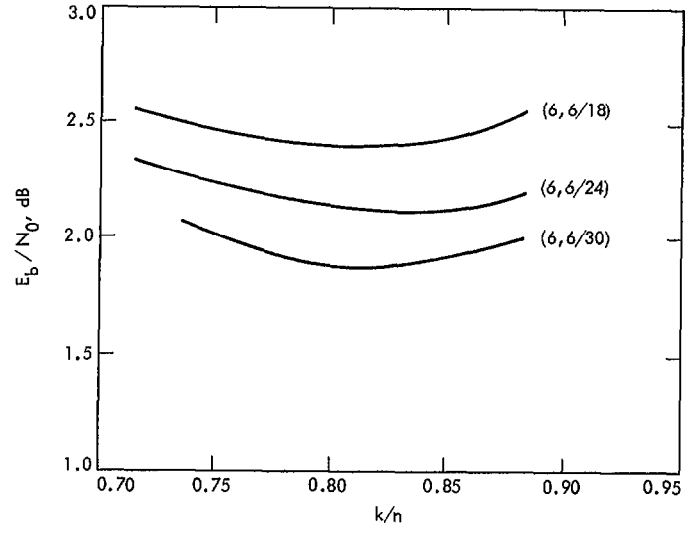

Fig. 4. Required E_b/No to achieve a BER of 10^{-6}