Coding and its application in space communications

Once regarded as purely academic, coding theory has turned out to be eminently practical for the modern applications of space channels

G. David Forney, Jr. *Codex Corporation*

Between 1948—when Shannon first proposed his basic theorems on information theory—and the start of the space age, little practical application developed from the lessons of coding theory. This article presents an overview of the Shannon theorem, interesting practical codes, and their application to the space channel. It turns out that a simple encoder in combination with a decoder of modest complexity placed into an uncoded communications system can increase the data rate by a factor of four or more depending on the coding scheme and the allowable error rate. Use of a convolutional code with sequential decoding has proved to be the outstanding scheme for these applications. It appears that, in the future, coding will find a place in most new digital space communication systems.

Coding theory has a history no doubt unique among engineering disciplines: the ultimate theorems came first, practical applications later. For many years after Shannon's announcement of the basic theorems of information theory in 1948, the absence of any actual realization of the exciting improvements promised by the theory was a source of some embarassment to workers in the field. A standard feature of IEEE Conventions in this period was a session entitled "Progress in Information Theory," or something similar, in which the talks purporting to show that the theory was approaching practical application tended instead to confirm the prejudices of practical men that information theory would do nothing for them.

In retrospect, there were two principal reasons for this lag. First, Shannon's coding theorems were existence theorems, which showed that within a large class of coding schemes there existed some schemes—nearly all, actually—that could give arbitrarily low error rates at any information rate up to a critical rate called channel capacity. The theorems gave no clue to the actual construction of such schemes, however, and the search for coding techniques capable of remotely approaching the theoretical capacity proved so difficult that a folk theorem was proposed: "All codes are good, except those we can think of."

Second, the channels of practical interest—telephone lines, cable, microwave, troposcatter, and HF radio—proved not to have anything like the statistical regularity assumed in the proof of the coding theorems. In fact, most theorems are based on the assumption of statistical independence in the noise affecting each transmitted symbol, whereas on the channels just cited disturbances tend to be manifested in bursts spanning many bits. This is to say nothing of other anomalies that arise in practice, such as a channel described at a recent information theory symposium as "a very good channel, with errors predominantly due to a noisy Coke machine near the receiver."

Over the past decade, the situation has improved tremendously. The problem of finding workable coding schemes has been recognized to be fundamentally a problem of finding decoders of reasonable complexity. The solution has been sought in considering classes of

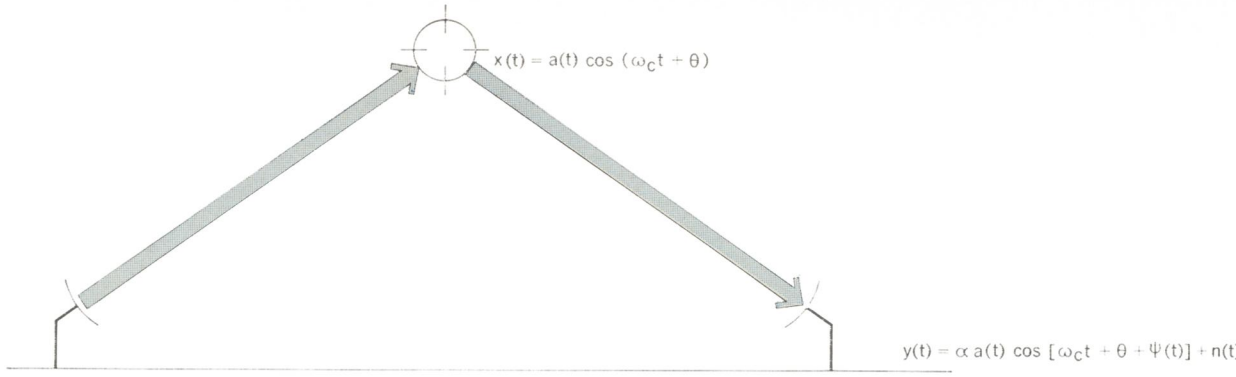
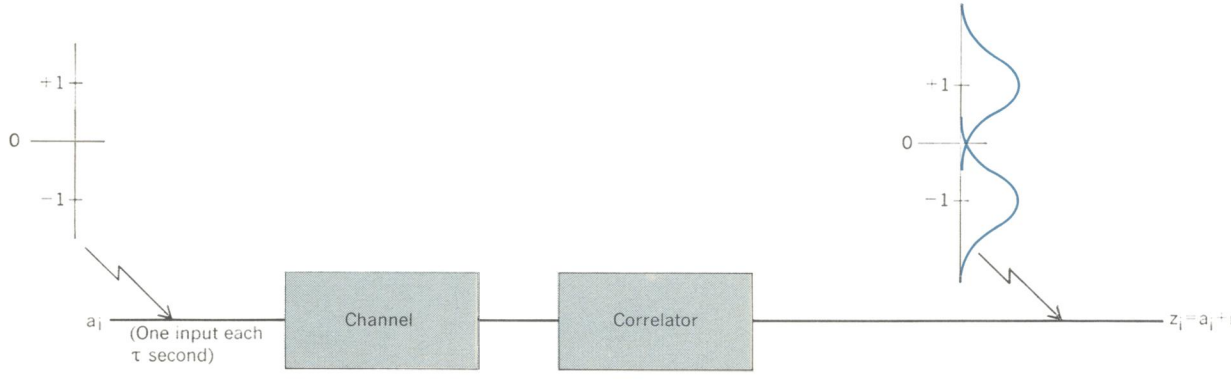


FIGURE 1. Amplitude modulation on a satellite channel.

FIGURE 2. Discrete-time channel model.



codes so structured that efficient decoding becomes feasible (but not so much structured that the codes themselves are no good). The most popular approach has been to use the structures of abstract algebra to generate classes of good, decodable block codes. A second approach uses linear sequential circuits to generate a class of codes that are called convolutional; at least for the applications to be discussed here, convolutional codes seem to have better balance between structure and randomness than is capable with the perhaps too-structured block codes.

A second major development of the last decade has been the emergence of the space channel into practical importance, both in the requirements of NASA for efficient transmission from deep-space probes, and in the proliferation of earth-orbiting communications satellites. The remarkable characteristic of the space channel is that, within the sensitivity of tests performed to date, it appears to be accurately modeled as a white-Gaussian-noise channel. Anyone who has ever taken a statistical subject knows that white Gaussian noise is the archetype of statistically regular, nonbursty noise, and as such is the theorist's dream. Consequently, in considering possible schemes for the space channel, one may use the most profound theorems, the most subtle analyses, and the most accurate simulations. One is also able to propose the most sophisticated and powerful decoding procedures, and predict performance to the accuracy of a fraction of a decibel. The initial successes of coding on the space channel have led to its incorporation in all space-system designs (of which the author is aware) in the last two years or so. For this reason, as well as the pedagogical neatness of the white-Gaussian-noise channel, this article uses the space channel for both orientation and motivation. We shall say little about the literally more mundane channels mentioned earlier, for although applications of coding have also been increasing in those environments, the schemes used are much more *ad hoc*, and more than qualitative predictions about behavior on real channels rarely can be made.

The space channel

The model of the space channel that we shall use reflects all the significant characteristics of the channel, without some details important only in practice; it is illustrated in Fig. 1. An amplitude-modulated carrier

$$x(t) = a(t) \cos (\omega_c t + \theta)$$

is generated aboard a satellite and transmitted to an earth antenna. (Frequency and phase modulation are also used, but not as often as AM, and offer no advantage in principle.) The model still applies when the signal actually originates at another ground station and the satellite is only a repeater, since the power available on the ground is so much greater than that aboard the satellite that the uplink may be considered perfect in most cases. The received signal

$$y(t) = \alpha a(t) \cos [\omega_c t + \theta + \psi(t)] + n(t)$$

is subject to several principal disturbances:

1. Simple attenuation α due to distance (assumed perfectly linear). The received signal power is denoted P.

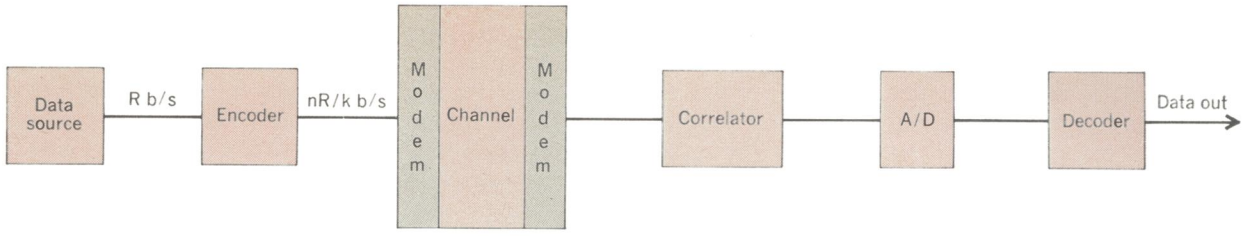


FIGURE 3. System block diagram.

2. Additive white Gaussian noise $n(t)$ arising in the receiver front end, with single-sided spectral density N_o.

3. Phase variations $\psi(t)$ due to imperfect tracking, uncompensated Doppler shifts, an unstable carrier oscillator, and so forth. In the applications with which the author is familiar, with the carrier ω_c in S band, the phase variations are the only important departure from the ideal white-Gaussian-noise model, and make themselves felt at low data rates by frustrating perfectly coherent demodulation. On a NASA mission with a terminal of the Goldstone type, phase variations can be kept to a few hertz or less, and are unimportant unless the bit rate is of the order of 10 bits per second or less. However, in some military applications where the receiver is aboard a plane, ship, jeep, or other moving platform, "low" data rates may be as high as 75 to 2400 b/s. We shall assume hereafter that we are at high enough rates that essentially perfect phase tracking and coherent demodulation can be maintained.

It will also be assumed that the information to be transmitted is already in digital form, leaving totally aside the kind of coding (source coding) that is concerned with efficient representation of the information in bits. (The gains from efficient source coding may be expected to equal or exceed those claimed in the following for efficient channel coding. The best techniques of the infant field of data compression are, however, even more *ad hoc* than those for channel coding on bursty channels.) The information rate will be denoted as R b/s.

When a communications system can pass R information bits per second over a white Gaussian channel on which the received power is P and the noise density N_o, with some acceptable quality, we say that the system is operating at a *signal-to-noise ratio per information bit* $E_b/N_o = P/N_oR$. This dimensionless parameter then serves as a figure of merit for different coding and modulation schemes. Note that it incorporates any effective power loss due to coding redundancy. A system designer who simply wants to select a communications scheme to get the most data rate for a given power and receiver noise temperature, or to use the least power for a fixed data rate, will pick the scheme that can operate at the lowest E_b/N_o with adequate quality (if he can possibly afford it).

An appropriate modulation technique, and the only one we shall consider, is pure time-discrete, N-level amplitude modulation. By this we mean that the modulating waveform $a(t)$ can only change at discrete intervals τ seconds apart, and during any τ-second period, sometimes called a baud, it can take on one of N discrete values, usually equally spaced. We let a_i be the value in the ith interval. If N is a power of two, say 2^m, then the *signaling rate* is $1/\tau$ symbols (bauds) per second, and the *transmitted rate* m/τ bits per second. Ideally, the bandwidth occupied is $W = \frac{1}{2}\tau$ hertz, but this is only an approximation (and a lower bound) to the practical bandwidth. By far the most common scheme of this class is the binary ($N = 2$) case, with $a(t) = \pm 1$; this is commonly called PSK or phase-shift keying, the terminology arising from a viewpoint in which $a(t)$ has constant magnitude 1 and the phase θ is modulated to the two values $\pm \pi/2$.

With white Gaussian noise, and perfect phase tracking, it is appropriate to use a correlation or matched filter receiver. Mathematically, in the ith baud such a receiver forms the integral

$$z_i = \int_{i\tau}^{i\tau+\tau} y(t) \cos [\omega_c t + \theta + \psi(t)]\, dt$$

It is easily shown that $z_i = a_i + n_i$, where a_i is the modulation amplitude (scaled) in the ith baud and n_i is the noise, a Gaussian random variable centered on 0 and independent from baud to baud. (This assumes perfect synchronization of the timing intervals, which can be approached as closely as desired in practice.) Furthermore, no information is lost in the correlation operation, in the sense that any decision on what was sent that is based on the correlator outputs z_i can be just as good as the information based on the complete received waveform. Thus we have replaced our continuous-time model with a discrete-time model, illustrated in Fig. 2 for PSK. Every τ seconds, a level a_i (one of N) is sent, and a correlator output z_i is received.

In the absence of coding, a *hard decision* is made on the correlator output as to which level was actually sent. For example, with binary PSK, a positive z_i leads to a decision of $+1$, and negative to -1. With coding, it is usually desirable to keep an indication of how reliable the decision was; this can range from establishing a null zone around 0, which is treated as no decision or an *erasure*, to retaining essentially all the information in the correlator output by sufficient finely quantized analog-to-digital conversion (normally three bits), called a *soft* (or *quantized*) *decision*. Schematically, any of these possibilities will be represented by a box following the correlator output labeled A/D.

We can now lay out the complete block diagram of a system that includes coding (Fig. 3). Information bits arrive at a rate of R b/s. An encoder of code rate k/n inserts $n - k$ redundant bits for every k information bits, giving a transmitted bit rate of nR/k b/s. These bits are taken m per baud into the modulator; at the receiver, a noisy correlator output is developed for each baud and A/D converted. The resulting hard decisions, soft deci-

sions, or whatever, enter the decoder, which uses the redundancy in the data as well as (with soft decisions) the reliability of the received information to estimate which information bits were actually sent. When the signal-to-noise ratio is specified, this is a well-defined mathematical model, and it makes sense to ask the question: How much information can we transmit through this channel, and what do we put in the encoder and decoder boxes to do it? The surprising fact upon which we commented at the beginning of this article is that the answer to the first question was announced long before anyone had the remotest idea how to answer the second.

Channel-capacity statements

Shannon's original work[1] showed that the capacity of the communication system blocked out in Fig. 3 is

$$C = \tfrac{1}{2} \log_2 (1 + P/N_oW) \quad \text{bits/baud}$$

$$\text{or } W \log_2 (1 + P/N_oW) \quad \text{bits/second}$$

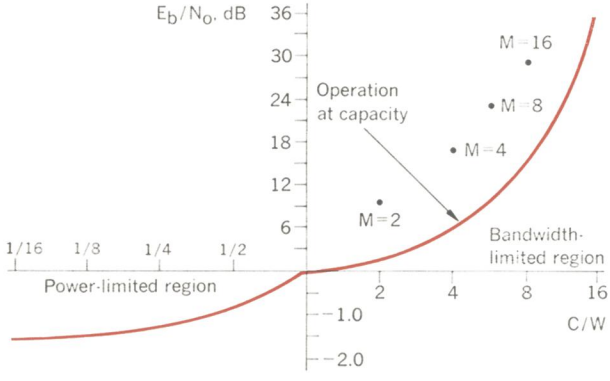
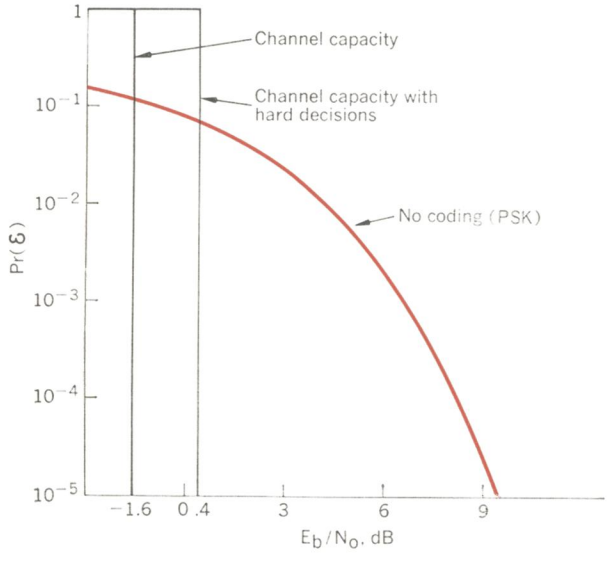


FIGURE 4. E_b/N_o as a function of C/W in bandwidth-limited and power-limited regions (note change of scale), with operation at capacity contrasted with M-level amplitude modulation [Pr (ε) = 10^{-5}].

FIGURE 5. Bit-error probability as a function of signal-to-noise ratio per information bit for situations involving no coding and coding at capacity.



where P is the received signal power, N_o the single-sided noise spectral density, and W the nominal bandwidth $\frac{1}{2}\tau$. Shannon showed that whenever the information rate R is less than C, then there exists some coding and modulation scheme with as low a decoded error probability as you like; whereas if $R > C$, then the error probability cannot approach zero and more coding generally only makes things worse. Finally, it can be shown that the same results apply when the special modulation assumptions of Fig. 3 are removed, and any signaling scheme whatsoever is allowed.

At one time, this classic formula fell into disrepute, after it had been used loosely by all sorts of coarse fellows who applied it promiscuously to channels not remotely characterized by the white-Gaussian-noise model. With the advent of the space channel, however, it is time to rehabilitate it for the insight it provides.

Suppose we could actually transmit at capacity; the signal-to-noise ratio per information bit would then be $E_b/N_o = P/N_oC$. The number of bits per cycle of bandwidth under the same conditions would be C/W. The capacity formula is usefully rewritten as a relation between these two dimensionless parameters:

$$C/W = \log_2 [1 + (P/N_oC)(C/W)]$$

This relation is plotted in Fig. 4. We see that, for a fixed power-to-noise ratio P/N_o, more and more efficient communication is possible as the bandwidth is increased, and that with no bandwidth limitations, E_b/N_o approaches a limit of ln 2 ($\approx$0.69, or −1.6 dB), called the Shannon limit. To date, space communication has been characterized by severe power limitation and bandwidth to burn, so that this so-called power-limited case has been the regime of interest. We note that, although the E_b/N_o limit is reached only for infinite bandwidth, at ½ bit per cycle of bandwidth (or a code rate of about ¼ with PSK) we are practically there.

Let us now see what coding has to offer in the power-limited case. Figure 5 is a more standard curve of error probability versus E_b/N_o in decibels. The no-coding curve is that for ideal PSK, which is representative of what was in fact used in the years B.C. (before coding), as in the Mariner '64 system that returned the first pictures from Mars. We see that an E_b/N_o of 6.8 dB is required to obtain a bit error probability of 10^{-3} and 9.6 dB to obtain 10^{-5}. On the other hand, the capacity theorem promises essentially zero error probability whenever E_b/N_o exceeds −1.6 dB. This means that potential coding gains of 8 to 11 dB (a factor of 6 to 12) are possible, which is rather exciting in an environment where the cost of a decibel is frequently measured in millions of dollars. Since, in the power-limited region, R is directly proportional to P, this gain may be taken either as reduced power or as increased data rate.

Another curve of parenthetical interest is included in Fig. 5, the capacity curve when the A/D box of Fig. 3 makes hard decisions. It turns out that this costs a factor of $\pi/2$ or 2 dB. We remark on this loss here because it seems to be one of the universal constants of nature: regardless of the coding scheme, use of hard decisions rather than soft in the power-limited region always costs about 2 dB.

The situation is quite different when the channel is bandwidth-limited rather than power-limited. The following simple argument shows that, in this region, coding

no longer offers such dramatic gains. Referring back to the capacity formula, we see that for $P/N_0W \gg 1$, with fixed N_0 and W, each increase by a factor of four in P leads to an increase of 1 bit/baud in channel capacity. On the other hand, consider what is required to increase the transmission rate in conventional multilevel amplitude modulation by 1 bit/baud. To double the number of signal levels while maintaining the same level separation and therefore the same probability of error requires increasing the amplitude span of the levels by a factor of two, as in Fig. 6, or the average power P by a factor of about four (this rapidly becomes exact as $N = 2^m$ increases). Thus, if R_{AM} is the rate achievable with amplitude modulation and C the capacity for some power P, then as P increases by k factors of four, we have

$$P \to 4^k P$$

$$R_{AM} \to R_{AM} + k$$

$$C \to C + k$$

$$\frac{R_{AM}}{C} \to \frac{R_{AM} + k}{C + k} \to 1 \quad \text{as } k \to \infty$$

Thus we can nearly achieve capacity without coding as we get deeper into the bandwidth-limited region. In Fig. 4, we plotted the first few AM points for $\Pr(\mathcal{E}) = 10^{-5}$ to show how rapidly R_{AM} approaches C. It therefore may be anticipated that as communications satellites achieve greater and greater effective radiated power the attractiveness of coding will diminish. One also suspects that this argument partially explains why, despite the fact that much outstanding early work on coding, including Shannon's, came out of the Bell Telephone Laboratories, to date there has been negligible operational use of coding on telephone circuits, which are engineered to be high signal-to-noise ratio, narrow-bandwidth lines. Comsat, by inheriting telephone-type tariffs that require its bandwidth to be offered in narrow slices, has been hobbled in the same way.

Maximum-length shift-register codes

In the remaining sections, we will discuss different types of codes and decoding methods, in an attempt to give an impressionistic feel for what they involve, with particular reference to performance on the power-limited space channel. We begin with block codes, which were the first to be studied and have the most well-developed theory. The maximum-length shift-register (or pseudo-random or simplex) codes are a class of codes that make a good introduction to algebraic block codes. Their properties are interesting and easy to derive, and serve as an easy entrée to the mysteries of finite fields, upon which further developments in block codes depend. Furthermore, they are actually useful in space applications and in noncoding areas as well. The number and quality of the pictures of Mars returned from the recent Mariner probes depended on the use of codes like these.

Consider first a digital feedback circuit such as the one depicted in Fig. 7; i.e., an m-bit shift register whose serial input is the modulo-2 (exclusive-or) sum of two or more of the bits in the shift register. In Fig. 7, $m = 4$ and the two bits are the rightmost b_1 and the leftmost b_4, so that the input b_{in} is expressed mathematically as

$$b_{in} = b_1 + b_4 \quad \text{modulo 2} \qquad (1a)$$

or, using the notation $\oplus$ for modulo-2 addition,

$$b_{in} = b_1 \oplus b_4 \qquad (1b)$$

When we say "shift register," we imply that whenever the circuit is pulsed by a clock pulse (not shown), b_{in} enters the left end, all other bits shift one place to the right, and the rightmost bit b_1 is lost.

It is well to be absolutely solid on the properties of modulo-2 arithmetic before striding off into the woods of algebraic coding theory.* Only two quantities occur in the arithmetic, 0 and 1. They may be added and multiplied as though they were ordinary integers, except that $1 \oplus 1 = 0$. This leads to the curious property that any number (0 or 1) added to itself in this arithmetic "cancels," i.e., equals zero, so that each number can be regarded as the negative of itself, and addition and subtraction are indistinguishable. (For example, if $a = b \oplus c$, then $b =$

* In general, the operations of modulo-N arithmetic (N equal to any integer) are the same as those of ordinary arithmetic after every number is reduced to its remainder when divided by N. For example, 8 modulo 3 is 2.

I. Modulo-2 arithmetic

Addition			Multiplication		
+	0	1	×	0	1
0	0	1	0	0	0
1	1	0	1	0	1

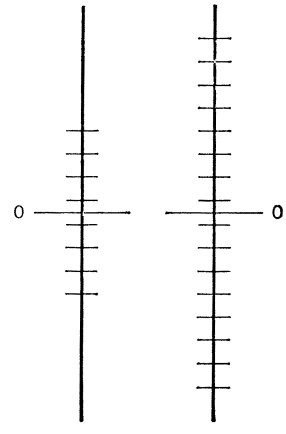
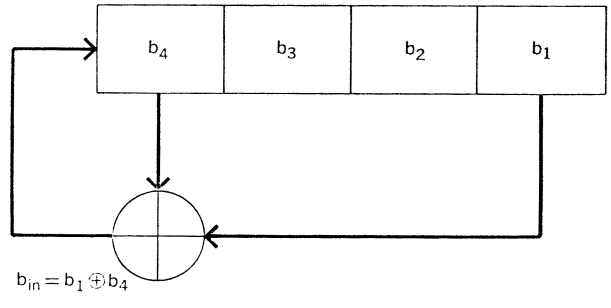


FIGURE 6. Doubling the number of levels with the same level spacing requires quadrupling the power in pulse amplitude modulation.

FIGURE 7. Maximum-length shift-register sequence generator with m = 4 stages.



$a \ominus c = a \oplus c$.) Addition and multiplication tables are given explicitly in Table I. It is easy to verify that all the ordinary rules of arithmetic—i.e., $a + b + c = c + b + a$, $a(b + c) = ab + ac$, etc.—apply in modulo-2 arithmetic, so that we can manipulate symbolic expressions freely, just as though they involved ordinary numbers, with the additional rule that $a + a = 0$.

Return now to the feedback circuit of Fig. 7. What happens when it is shifted a number of times? The answer clearly depends on what its initial contents are. If all stages initially contain zeros, then the input will be zero, so that a shift will leave the register in the all-zero state. There are 15 other initial states; if we pick one of them, say 0001, and use Eq. (1), we find that 15 shifts cycle the register through all nonzero states and return the register to the starting point. The state diagram is shown in Fig. 8; it consists of two cycles: the one-state all-zero cycle, and the 15-state nonzero cycle. The name "maximum-length shift register" is given to this circuit since, given that 0000 must go to 0000, the 15-state cycle is the maximum length possible.

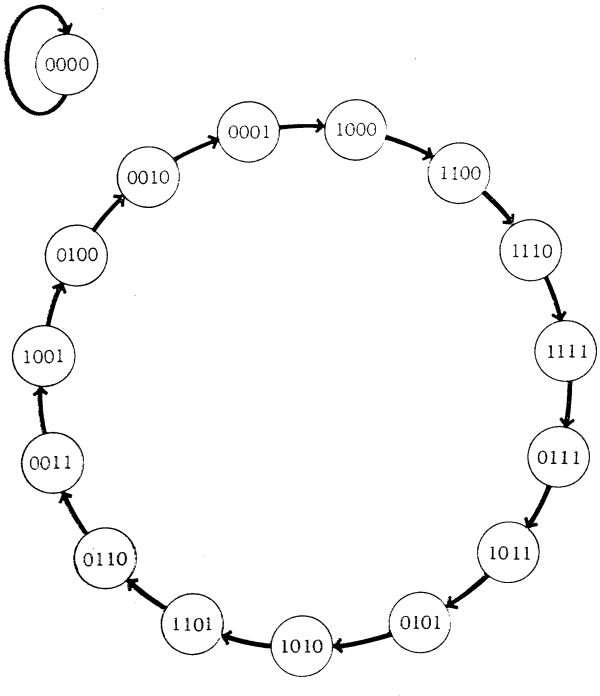
It is a nontrivial result of algebra that for any number of stages m we can always find a circuit like Fig. 7 with a state diagram like Fig. 8. The input is always a modulo-2 sum of certain stages of the register, so the all-zero state always gives a zero input, and the zero state always goes into the zero state on a shift. The remaining $M - 1$ states form a maximum-length cycle, where $M = 2^m$. Table II specifies input connections to the modulo-2 adder that will give a maximum-length shift register for $1 \leq m \leq 34$.

A block code using the circuit of Fig. 7 as an encoder operates as follows: The message to be transmitted, assumed to be a sequence of bits, is segregated into 4-bit segments. Each segment is loaded into the 4-bit shift register, and the register is shifted 15 times. The 15 bits coming out of the rightmost stage of the register are transmitted as a block, or code word. Table III gives the 15-bit code words corresponding to each 4-bit information segment.

This code is called a (15, 4) code, since code words have 15 bits for each 4 information bits. By using registers of different lengths m, we can create $(M - 1, m)$ codes. Since $M = 2^m$, as m gets large, the ratio of information bits to transmitted bits (the code rate) becomes very small, which limits the usefulness of these codes for coding purposes; in other applications, however, the fact that a very long nonrepeating sequence can be generated with a short register is the feature of interest.

We can quickly determine some properties of the $(M - 1)$-bit sequences generated by these registers. First, the bits in these sequences are the rightmost bits of the $M - 1$ nonzero state sequences of length m. Since exactly half of all m-bit sequences end in "1," precisely $M/2$ 1's occur in any maximum-length sequence (for example, 8 bits out of the 15 in the sequence of the example). In a long sequence, if we look at the output at a random time, the probability of seeing a "1" is $(M/2)/(M - 1)$, or just

FIGURE 8. State diagram of feedback circuit in Fig. 7.



II. Connections for MLSR generators

m	Stages Connected to Modulo-2 Adder	m	Stages Connected to Modulo-2 Adder
1	1	18	1, 12
2	1, 2	19	1, 15, 18, 19
3	1, 3	20	1, 18
4	1, 4	21	1, 20
5	1, 4	22	1, 22
6	1, 6	23	1, 19
7	1, 7	24	1, 18, 23, 24
8	1, 5, 6, 7	25	1, 23
9	1, 6	26	1, 21, 25, 26
10	1, 8	27	1, 23, 26, 27
11	1, 10	28	1, 26
12	1, 7, 9, 12	29	1, 28
13	1, 10, 11, 13	30	1, 8, 29, 30
14	1, 5, 9, 14	31	1, 29
15	1, 15	32	1, 11, 31, 32
16	1, 5, 14, 16	33	1, 21
17	1, 15	34	1, 8, 33, 34

III. Code words in a (15, 4) code

Information Bits	Code Word
0000	000000000000000
0001	000111101011001
1000	100011110101100
0100	010001111010110
0010	001000111101011
1001	100100011110101
1101	110010001111010
0110	011001000111101
1011	101100100011110
0101	010110010001111
1010	101011001000111
1101	110101100100011
1110	111010110011001
1111	111101011001000
0111	011110101100100
0011	001111010110010

about ½. Furthermore, since all m-bit sequences except the all-zero sequence occur somewhere in the maximum-length sequence, the probability of seeing a "1" given any $m - 1$ or fewer preceding bits is still nearly one half. These and other statistical properties make a maximum-length sequence difficult to distinguish from a sequence generated truly randomly, as by flipping a coin, yet these sequences are easy to generate and repeatable. Thus they are commonly used to generate pseudorandom bits.

The class of maximum-length shift-register codes is representative of the major classes of algebraic block codes, in that such codes have the properties of being

1. Systematic; that is, the information bits are transmitted unchanged as part of the code word. In the example (Table III), the first four bits of each code word are the information bits.

2. A parity-check code; that is, each of the noninformation (parity) bits is a parity check on (modulo-2 sum of) certain information bits. This can be proved inductively; for example, in the example code, the fifth bit is the modulo-2 sum of the first and fourth; the sixth is the sum of the second and fifth, but this is the same as the second plus the first plus the fourth; in general, the nth bit is some modulo-2 sum of previous bits, which are themselves each modulo-2 sums of information bits, so the nth bit is also some modulo-2 sum of information bits. (In fact, in the maximum-length shift-register codes, the parity bits consist of all possible different parity checks on the information bits.)

3. Cyclic; that is, the end-around shift of any code word is another code word.

The parity-check property can be used to prove the most important single result concerning parity-check codes (the group property), which is that if we form the modulo-2 sum of two code words, we get another code word.

The modulo-2 sum of two n-bit code words is defined as the bit-by-bit modulo-2 sum; that is, if x_i and y_i, $1 \leq i \leq n$ are the bits in the two original code words, then the bits z_i in their sum are

$$z_i = x_i \oplus y_i$$

Thus the information bits in z are the modulo-2 sum of the information bits in x and y. The parity bits in z are what we get when we put the modulo-2 sum of the information bits in x and y into our 4-bit register and shift 15 times; it is not hard to see that they are the modulo-2 sum of the parity bits in x and y, since the shift-register connection is itself a modulo-2 sum. In other words, the two circuits in Fig. 9 have identical outputs.

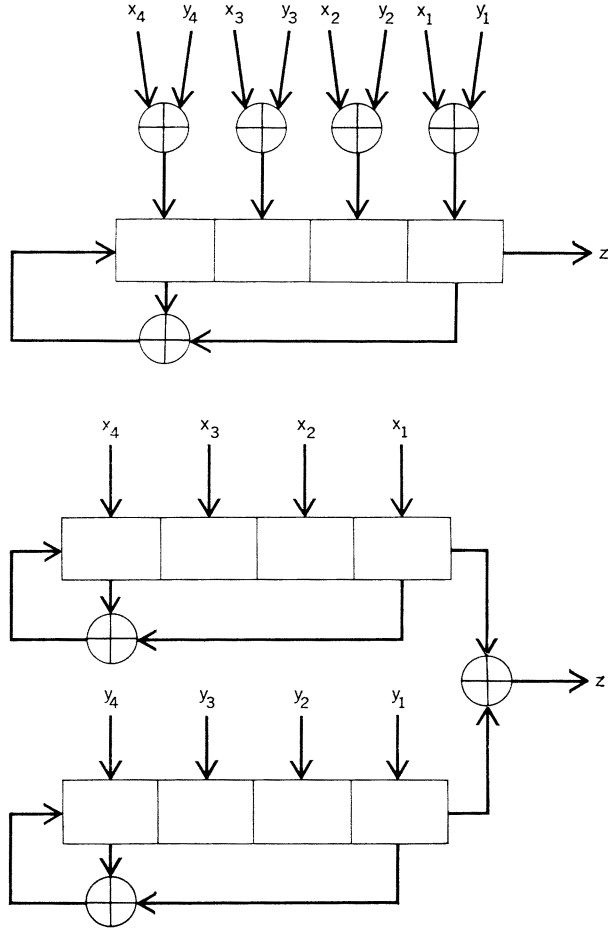
This can be verified also by taking any two of the words in Table III and forming their modulo-2 sum; the result will be another one of the cyclic shifts of the basic sequence.

The group property gives immediate answers to questions about distance or correlation between code words. The distance (Hamming distance) between two code words is defined as the number of places in which they differ. If we form the modulo-2 sum of two code words, the resulting word will have zeros in the positions in which the two code words agree, and ones where they differ; thus the distance between two code words is exactly the number of ones in their sum. But, from the group property, their sum is another code word; and in the maximum-length shift-register codes all words have the same number of ones,* namely $M/2$ (eight in our example). Thus the distance between any two words in these codes is $M/2$, or about half the code length.

The equidistant property of maximum-length shift-register codes makes them an optimum solution to the following problem in signal design: How can one construct M equal-energy signals to minimize the cross-correlation between any two signals, with no bandwidth limitations? Let us suppose that a code word is sent by PSK, so that a 0 is sent as a baud of amplitude -1 and a 1 as amplitude $+1$. The M-code words then correspond to M vectors in $M - 1$ dimensions, all of equal energy (autocorrelation) $M - 1$. The cross-correlation (inner product) of any two vectors is a sum of baud-by-baud correlations, equal to $+1$ if the vectors agree in that place, and -1 if they disagree. But we have just proved that the Hamming distance between any two code words is $M/2$, so that any two vectors disagree in $M/2$ places and agree in the remaining $M/2 - 1$. Consequently, any two vectors are anticorrelated with cross-correlation -1. This implies that as vectors in $(M - 1)$-space, the code words form a geometrical object called a simplex, which is universally believed (though it has never quite been proved) to be the distribution of equal-energy signals in signal space that minimizes the probability of in-

* Except the all-zero word, of course; this is the code word we get when we sum any code word with itself.

FIGURE 9. Two equivalent linear circuits.



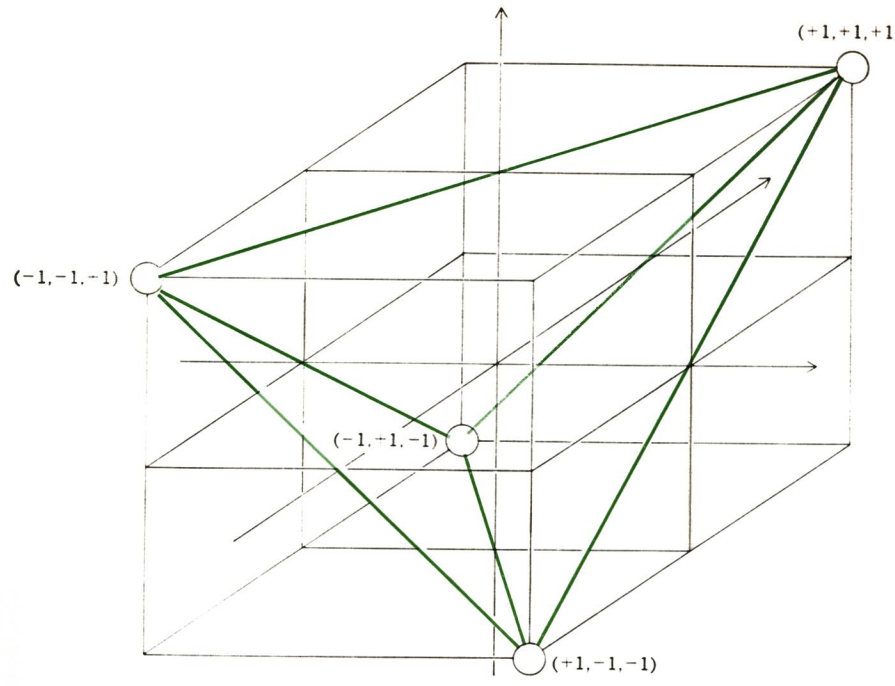


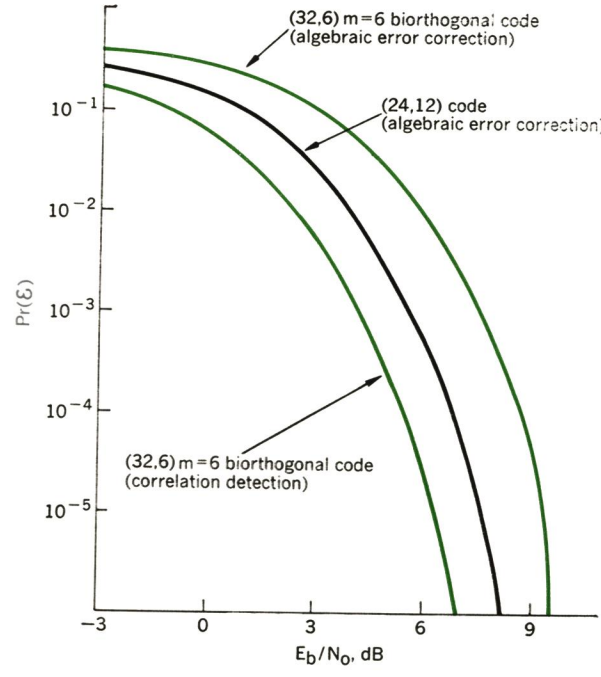
FIGURE 10. Simplex (tetrahedron) formed by m = 2, (3, 2) code in three dimensions.

correct detection. Figure 10 shows the simplex corresponding to the $m = 2$ maximum-length shift-register code, which takes the form of a tetrahedron in three dimensions. Here is an intriguing contact between algebraic coding theory and the geometry of N dimensions.

Suppose now that we use such a binary code with PSK modulation; how shall we decode it at the receiver? As in Fig. 3, we assume that we start with the $M - 1$ correlator outputs z_i that correspond to the $M - 1$ bauds required to send a code word. For definiteness, we use the code of our example in which $m = 4$ and $M - 1 = 15$. Here we shall see a distinction between the viewpoints of the signal designer and of the algebraic coding theorist. The signal designer would take the attitude that what we have here is a way of sending one of 16 signals through a white Gaussian channel, where each possible signal is made up of 15 binary chips, and thus is a vector in 15 dimensions. As in the pure binary case, the optimum detection method is to correlate the received signal against all the 16 possible transmitted signals, which can be done by simply summing the correlator outputs z_i multiplied by ± 1 according to the code word amplitude in the corresponding baud. Thus 16 computations followed by a selection of the largest correlation must be performed. (It turns out that the correlations can be done simultaneously in a special-purpose computer—called the "Green machine" at Jet Propulsion Laboratory[2]—as an M-point fast Hadamard transform, which is structurally very similar to a fast Fourier transform.) The computational load remains manageable for m less than eight or so, which is also where the bandwidth occupied by these codes begins to be absurdly large. A modified (biorthogonal) $m = 6$ code was used in the Mariner '69 expedition; its performance curve is shown in Fig. 11.[3]

An alternate approach is usually taken by the algebraic coding theorist. The first step is to make a hard decision on each correlator output to obtain a 15-bit digital word

FIGURE 11. Performance of various block codes.



called the received word. Now we are back in the realm of modulo-2 arithmetic, Hamming distance, and so forth. In the hard-decision process, a number of bit errors will usually be made. From the distance properties of the original code, one can determine that if fewer than some maximum number of errors occur, then correct decoding is guaranteed. In the example, where the Hamming distance between any two words is eight, it is easy to see that if three errors occur in the reception of any code word, then the received word will differ in three places from the correct word, but in at least five places from any other word, so that in principle decoding should be correct. In

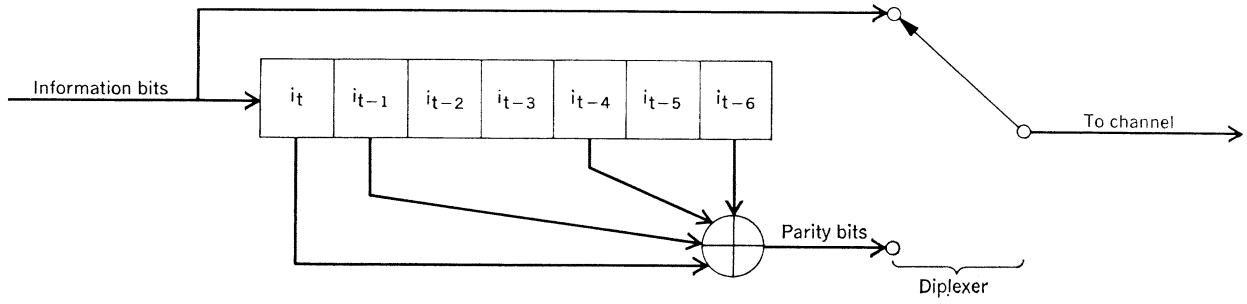


FIGURE 12. Convolutional encoder.

general, a number of bit errors equal to the greatest integer less than half the Hamming distance is guaranteed to be correctable.

One decoding method suitable for the (15, 4) example code is permutation decoding.[4] From the distance properties of this code, we know that if we can find some code word within Hamming distance three of the received word, then we should assume that word was sent, since all other code words must be at least distance five from the received word. To find such a word, we can start by simply reencoding the four received information bits, and checking whether the reencoded parity bits agree with the received parity bits in all but three or fewer places. If so, we are done. If not, then because of the cyclic property of the code, we can take any other four consecutive received bits, treat them as information bits, and generate the rest of the cycle (end-around) that makes up the code. (If this is not immediately clear, try taking any four consecutive positions of any code word in Table III, loading them into the encoder shift register, and shifting 15 times to generate the whole code word, starting at that point and cycling around past the beginning.) If there are actually three or fewer bit errors, at least one set of four consecutive positions will be received correctly, so by taking each set of four in turn, reencoding, and comparing, we will eventually find the correct code word. (This cyclic permutation scheme is also used to correct burst errors, since a correctable burst of errors will not affect at least one set of k consecutive bits.[5])

The performance of hard decisions followed by algebraic error correction is also shown in Fig. 11, for the same (32, 6), distance-16 code as in the correlation detection curve. We see that it is more than 3 dB worse for the lower error probabilities. It might therefore seem that the correlation technique is the better one; however, algebraic decoding remains feasible for much longer code lengths and numbers of information bits, where correlation detection is computationally infeasible. A curve for the (24, 12) (minimum distance eight) extended Golay code is also shown in Fig. 11; with longer codes, the hard-decision disadvantage can eventually be overcome.

Ideally, one would like a scheme whose computational complexity was like that of the algebraic decoding schemes, but would make use of all the information in the correlator output and thus achieve performance like that of correlation detection. At least two approaches (orthogonal equation decoding[6,7] and generalized minimum-distance decoding[8]) with these features are known, but they have not been extensively studied due to the existence of superior convolutional coding schemes (to be described in the next section).

Although we have studied only the maximum-length shift-register codes here, more advanced algebraic block codes involve quite similar ideas. Peterson[9] and Berlekamp[10] are the standard references of the field.

Convolutional codes

Historically, the coding world has been divided between block-code people and convolutional-code people. Although relations between these groups are perfectly amicable, block-code types tend to harp on the relatively primitive theoretical understanding and development of convolutional codes vis-à-vis block codes, whereas convolutional-code types point out that in all respects in which convolutional codes can be compared with block codes they are essentially as good in theory, and in some major respects better, while in practice they are typically simpler. The correctness of both these viewpoints will be illustrated in this section. Whereas we have considered an infinite class of good block codes, we cannot now consider such a class of convolutional codes, since classes of reasonably good codes in the block-code sense are unknown. Instead we shall consider a simple typical code and some reasonable ways of decoding it. The best of these methods will be seen to give better performance on the space channel than any block-code techniques.

Consider the linear sequential circuit illustrated in Fig. 12. Like the maximum-length shift-register generator of Fig. 7, it consists of a shift register and a modulo-2 adder connected to several shift-register stages. In this case, however, information bits are continuously entered into the left end of the register, and for each new information bit a parity bit (a parity check on the current bit and three of those in the past) is computed according to the formula

$$p_t = i_t \oplus i_{t-1} \oplus i_{t-4} \oplus i_{t-6}$$

Information and parity bits are transmitted alternately over the channel. The code generated by this encoder is called a rate-½ convolutional code: rate ½ because there are two transmitted bits for every information bit, convolutional because the parity sequence is the convolution of the information sequence with the impulse response 1,1,0,0,1,0,1, modulo-2. Like the block codes considered earlier, the code is systematic (information bits are transmitted), and is a parity-check code; therefore, it has the group property (the modulo-2 sum of two encoded se-

quences is the encoded sequence corresponding to the modulo-2 sum of the information sequences).

We shall now suppose that the encoded sequence is sent over a binary channel and that hard decisions are made at the receiver output. How do we decode? First, the decoder must establish which received bits are information and which parity, but as there are only two possibilities, trial and error is a feasible procedure. (For block codes, the comparable problem involves a choice between n phases, where n is the block length, and some special synchronization means may be required.) This done, we shall let the decoder form *syndromes*, which are defined as follows:

Take the received information sequence, and from it recompute the parity sequence with an encoder identical to that of Fig. 12. Compare these recomputed parity bits with the parity bits actually received; the outputs from the comparator (another modulo-2 adder) are called the syndromes (see Fig. 13). (The syndrome idea is equally useful with block codes.)

It is evident that if no errors occur in transmission over the channel, the recomputed parity bits will equal the received parity bits and all syndromes will be zero. On the other hand, if an isolated error occurs in the parity sequence, then a single syndrome will be equal to one at the time of the error. If an isolated error occurs in the information sequence, then the syndromes will equal one at all times when the incorrect bit is at a tapped stage of the shift register, so the syndrome sequence will be 1,1,0,0,1, 0,1,0,0 . . . , starting at the time of the error. The syndrome pattern for more than one error is just the linear superposition (modulo-2) of the syndrome patterns for each of the individual errors. Thus do the syndromes indicate the nature of the disease.

An obvious technique for correcting single isolated errors now suggests itself. Such an error will manifest itself as a syndrome pattern of 1100101 or 1000000, depending on whether it is in an information or a parity bit. The first time we see a 1 in the syndrome sequence, we know that an error has occurred; the value of the following syndrome tells us whether it was an information or parity error. Since only information errors need be corrected, an AND gate looking for two successive syndrome "ones" suffices, as illustrated in Fig. 14(A).

One can correct double errors with the hardly more complicated circuit of Fig. 14(B). Here the syndromes are fed into a 7-stage shift register; a threshold circuit fires if three or four of four selected places contain ones. The selected places are those that would contain ones if there were only a single information error. A single parity error, in addition, can only disturb one input to the threshold circuit; similarly, it can be verified that with this particular code a second information error can only interfere with one input, so that if only two errors occur the threshold circuit will certainly fire at the right time. On the other hand, it can also be verified that under the assumption of only two errors the circuit will never fire at the wrong time. Finally, the complement line is included to take out the effect of a corrected error in those syndrome bits that were inverted by it, so that the decoder can handle all error patterns that do not have more than two errors in any seven consecutive pairs of received bits.

Both these decoders are examples of threshold decoders[7] (working on a self-orthogonal code[11]). Threshold

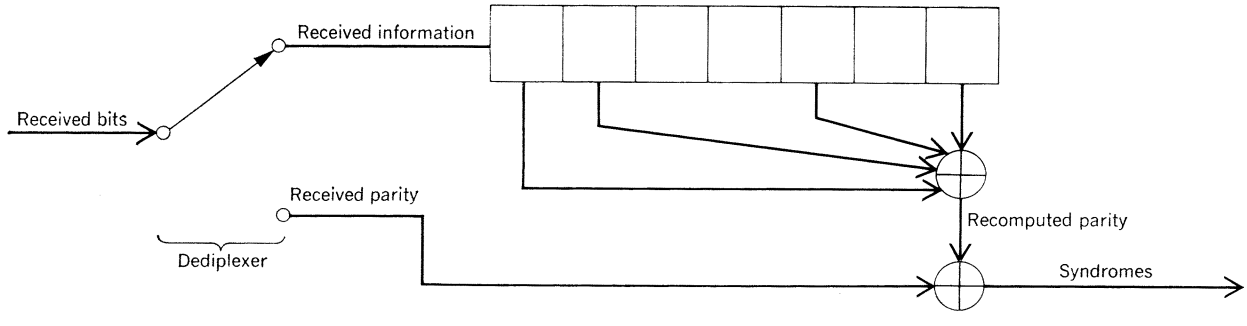
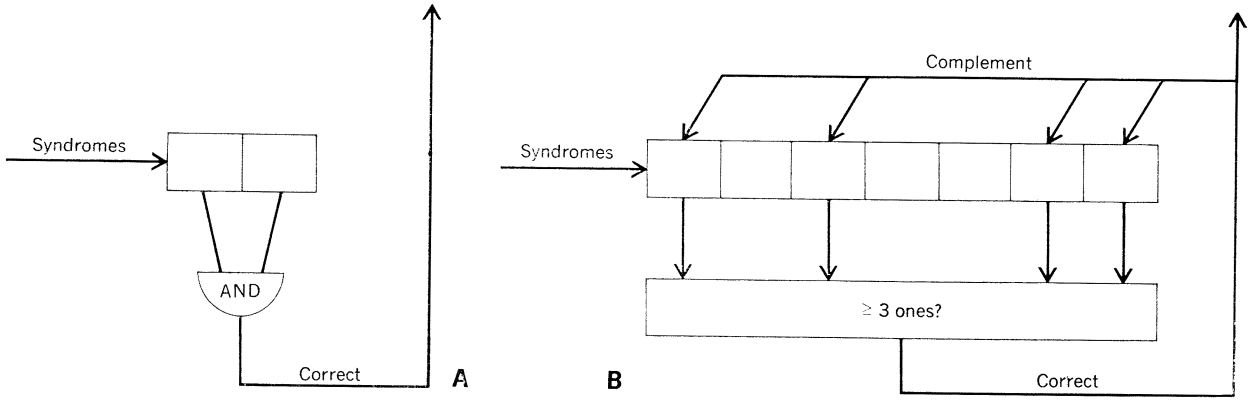


FIGURE 13. Syndrome formation at the receiver.

FIGURE 14. Simple single- and double-error-correcting threshold decoders.



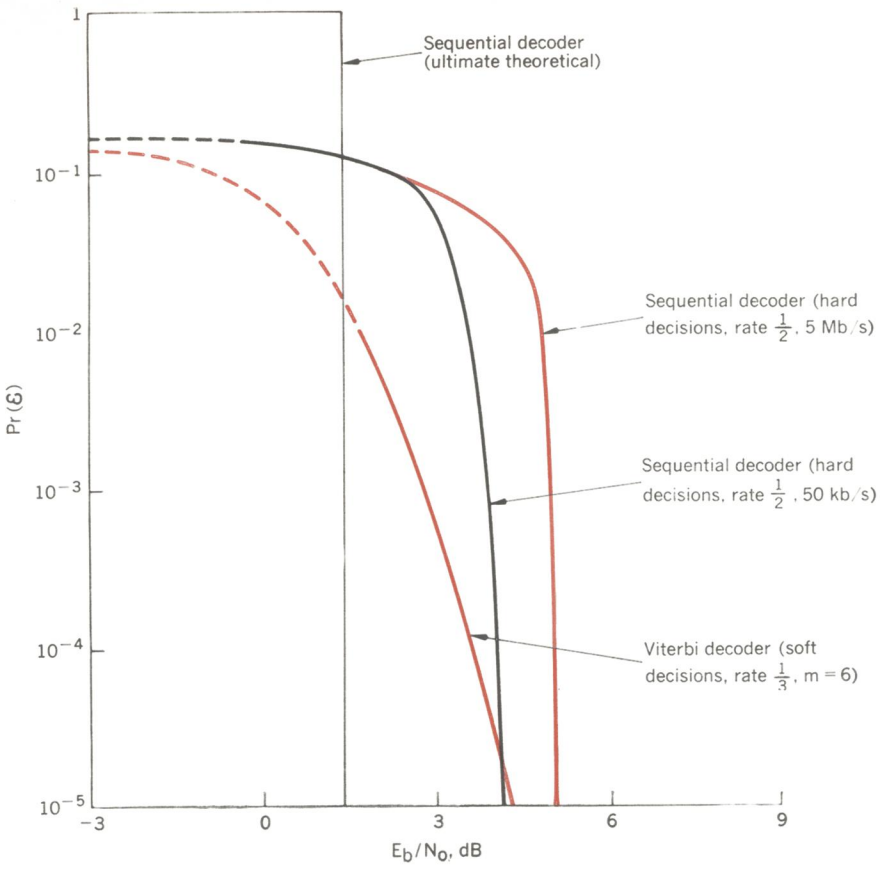


FIGURE 15. Performance of convolutional decoders.

decoding is an extremely simple technique that applies to many short codes correcting a few errors, and that is easily extended to correct bursts of errors. Its efficiency diminishes as the number of errors to be corrected becomes large, and for this reason it is not an outstanding performer on the space channel. With hard decisions, the performance of the three-error-correcting (24, 12) convolutional code (shift register length 12) is about the same (to within 0.2 dB) as that of the (24, 12) block code of Fig. 11.

Sequential decoding was invented by Wozencraft[12] in about 1957. Through a decade of improvement, analysis, and development, it has become the best-performing practical technique known for memoryless channels like the space channel, and will probably be the general-purpose workhorse for these channels in the future. Like much else in the convolutional-coding domain, it is hard to explain and analyze, but relatively easy to implement. Very crudely, a sequential decoder works by generating hypotheses about what information sequence was actually sent until it finds some that are reasonably consistent with what was received. It does this by a backward and forward search through the received data (or through syndromes). It starts by going forward, generating a sequence of hypotheses about what was sent. It checks what was received against what would have been transmitted given the hypotheses, and according to the goodness of the agreement updates a measure of its happiness called the metric. As long as it is happy, it goes forward; when it becomes unhappy, it turns back and starts changing hypotheses one by one until it can go forward happily again. A simple set of rules for doing this is called the Fano algorithm.[13-15]

It is evident even from this meager description that sequential decoding involves a trial-and-error search of variable duration. When reception is perfect, the decoder's first guess is always correct, and only one "computation" (generation of a hypothesis) is required per bit. The more noise, the more hypotheses must be generated, up to literally millions to decode a single short segment. Because of the variability of the computational load, buffer storage of the received data must be provided to permit long searches. Whenever this buffer overflows, the decoder must jump ahead and get restarted, leaving a section of data undecoded. This overflow event therefore leads to a burst of output errors; its frequency generally dominates the probability of decoding error, since the code can be made long enough that the probability the decoder is actually happy with incorrect hypotheses can be made negligible.

Sequential decoding is outstandingly adaptable; it can work with soft or hard decisions and PSK, or with any modulation and detection scheme. In the four implementations for the space channel to date, the Lincoln Experimental Terminal decoder[16] works with 16-ary frequency-hopping modulation and incoherent list detection; the NASA Ames decoder for the Pioneer satellites[17] and the JPL general-purpose decoder[18] work with PSK and soft (eight-level) decisions; and the Codex decoder, built for the U.S. Army Satellite Communication Agency,[19] works with PSK (or DPSK or QPSK) and hard decisions, the choice in every case being based on system considera-

tions. Sequential decoding can even make efficient use of known redundancies in the data, as was done for some preexisting parity checks in the Pioneer data format. The one thing a sequential decoder cannot tolerate is bursts of errors, which will cause excessive computation; therefore, it cannot be applied without modification to any channel but the space channel.

The performance of sequential decoding depends both on the modulation and detection scheme with which it is used, and on the data rate relative to the internal computation rate. The theoretical limit of any sequential decoder on a white Gaussian channel is $E_b/N_o = 1.4$ dB, exactly 3 dB above the Shannon limit; this limit can be approached with PSK, soft decisions, and low-rate codes. The simplest possible sequential decoder working with rate-½ codes, PSK, and hard decisions has a theoretical limit of $E_b/N_o = 4.5$ dB; 2 dB of this loss is due to hard decisions, 1 dB to the choice of rate ½ rather than a lower rate. Actual performance depends on the data rate as well as the error rate desired, although the curves are very steep; Fig. 15 shows measured curves at 50 kb/s and 5 Mb/s for the Codex decoder,[20] which has an internal computation rate of 13.3 Mb/s.

Somehow the idea that sequential decoding is complicated to implement has achieved considerable circulation. This is undoubtedly partly due to the difficulty of the literature. Also, the first sequential decoder (SECO[21]), built at Lincoln Laboratory for telephone lines with the technology of an earlier day, was an undoubted monster, due in part to large amounts of auxiliary equipment such as equalizers. It should be emphasized that three of the four implementations just mentioned involve only a drawer of electronics with a core memory system for the buffer storage; the fourth, the Pioneer system, was actually done in software because of the low maximum bit rate (512 b/s).

We conclude by mentioning two more classes of schemes of current interest. One, the Viterbi algorithm,[22] performs optimum correlation detection of short convolutional codes much as the Green machine does of block codes. Figure 15 shows the performance of this algorithm[23] with soft decisions when the decoding complexity is comparable to that of the $m = 6$ block decoder of Fig. 11; performance is uniformly superior. This algorithm is competitive in performance with sequential decoding for moderate error rates, but cannot achieve very low error rates efficiently. On the other hand, it can be implemented in a highly parallel pipe-lined decoder capable of extremely high speeds (tens of megabits) where sequential decoders become uneconomic. It therefore may find application in high-data-rate systems with modest error requirements, such as digitized television.

The second class represents attempts to bridge the final 3-dB gap between the sequential decoding limit and the Shannon limit by combining sequential decoding with algebraic block code constraints. Recent unpublished work of Jelinek gives promise of performances between 1 and 2 dB from the Shannon limit without excessive computation. At the moment, all schemes in this class seem most suited for software implementation, and will probably be used only for low-data-rate applications where the ultimate in efficiency is desired, as in deep-space probes.

Thus do we near practical achievement of the goal set by Shannon 20 years ago.

REFERENCES

1. Shannon, C. E., "A mathematical theory of communication," *Bell System Tech. J.*, vol. 27, pp. 379–423, 623–656, 1948.
2. Green, R. R., "A serial orthogonal decoder," Jet Propulsion Lab. Space Programs Summary 37–39, vol. 4, pp. 247–252, 1966.
3. *Digital Communications with Space Applications*, appendix 4, S. Golomb, ed. Englewood Cliffs, N.J.: Prentice-Hall, 1964.
4. MacWilliams, J., "Permutation decoding of systematic codes," *Bell System Tech. J.*, vol. 43, pp. 485–506, 1964.
5. Gallager, R. G., *Information Theory and Reliable Communication*. New York: Wiley, 1968, pp. 291–297.
6. Gallager, R. G., *Low-Density Parity-Check Codes*. Cambridge, Mass.: M.I.T. Press, 1963, pp. 42–52.
7. Massey, J. L., *Threshold Decoding*. Cambridge: M.I.T. Press, 1963, pp. 59–63.
8. Forney, G. D., "Generalized minimum distance decoding," *IEEE Trans. Information Theory*, vol. IT-12, pp. 125–131, Apr. 1966.
9. Peterson, W. W., *Error-Correcting Codes*. Cambridge: M.I.T. Press, 1961.
10. Berlekamp, E. R., *Algebraic Coding Theory*. New York: McGraw-Hill, 1968.
11. Robinson, J. P., and Bernstein, A. J., "A class of binary recurrent codes with limited error propagation," *IEEE Trans. Information Theory*, vol. IT-13, pp. 106–113, Jan. 1967.
12. Wozencraft, J. M., and Reiffen, B., *Sequential Decoding*. Cambridge: M.I.T. Press, 1961.
13. Fano, R. M., "A heuristic discussion of probabilistic decoding," *IEEE Trans. Information Theory*, vol. IT-9, pp. 64–74, Apr. 1963.
14. Wozencraft, J. M., and Jacobs, I. M., *Principles of Communication Engineering*. New York: Wiley, 1965.
15. Gallager, R. G., *op. cit.*, pp. 263–286.
16. Lebow, I. L., and McHugh, P. G., "A sequential decoding technique and its realization in the Lincoln Experimental Terminal," *IEEE Trans. Communication Technology*, vol. COM-15, pp. 477–491, Aug. 1967.
17. Lumb, D. R., "Test and preliminary flight results on the sequential decoding of convolutionally encoded data from Pioneer IX," 1969 IEEE Internat'l Communications Conf. Record, Boulder, Colo., pp. 39/1–8.
18. Lushbaugh, W., "Multiple-mission sequential decoder," Jet Propulsion Lab. Space Programs Summary 37–58, vol. 2, pp. 33–36, 1969.
19. Forney, G. D., Jr., and Langelier, R. M., "A high-speed sequential decoder for satellite communications," *1969 IEEE Internat'l Communications Conf. Record*, Boulder, Colo., pp. 39/9–17.
20. "High-speed sequential decoder," Codex Corp. final rept., Contract DAAB07-69-C-0051, U.S. Army Satellite Communication Agency, Ft. Monmouth, N.J., June 6, 1969.
21. Lebow, I. L., *et al.*, "Application of sequential decoding to high-rate data communication on a telephone line," *IEEE Trans. Information Theory*, vol. IT-9, pp. 260–269, Apr. 1963.
22. Viterbi, A. J., "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Information Theory*, vol. IT-13, pp. 260–269, Apr. 1967.
23. Heller, J., "Improved performance of short constraint length convolutional codes," Jet Propulsion Lab. Space Programs Summary 37–56, vol. 3, pp. 83–84, 1969.

G. David Forney, Jr. (M) received the B.S.E. degree from Princeton in 1961, and the M.S. and Sc.D. degrees from M.I.T. in 1963 and 1965, respectively. He has been with Codex Corporation, Watertown, Mass., since 1965, and now serves as director of research. Responsible for company efforts in space communications, he has also been involved in the design and development of products in the areas of coding, multiplexing, and modems. Dr. Forney is the author of several journal articles and a book entitled "Concatenated Codes"; although the book is concerned with block codes, he is usually identified as a convolutional-code type. He is currently vice chairman of the Boston IEEE Information Theory Group Chapter and is also a member of the AAAS.