Soft-decision threshold decoders.

R.M.F. Goodman, B.Sc., Ph.D.,
University of Hull,
England.

Summary

Coding system designers are interested in threshold decoding for convolutional codes because of the hardware simplicity of the decoder. Unfortunately, majority-decision threshold decodable codes are sub-optimum, and this involves a loss in coding gain. In this paper a new method for implementing soft-decision threshold decoding is introduced, enabling some of the loss in coding gain to be recovered without too great a sacrifice in hardware simplicity. Decoders for constraint length 2 and 12 segments are described, and their performance in Gaussian noise evaluated. The soft-decision technique used can also be applied to block codes with similar improvements in coding gain, and methods of implementing this are discussed.

1. Introduction

Binary convolutional codes have been shown to exhibit extremely good error-control properties under both Gaussian and burst noise conditions. In the case of the additive white Gaussian channel, there are several powerful convolutional decoding schemes (sequential decoding, Viterbi decoding) that yield high coding gains (5dB at a sink bit error rate of 10^{-5}). Unfortunately, the hardware complexity of such schemes is high, as the decoders are essentially large special-purpose computers. In addition, the burst-noise performance of these powerful schemes tend to be disappointing in comparison with convolutional code systems designed specifically for burst-error correction.

The system designer is therefore often interested in convolutional decoding schemes that sacrifice a few dB of coding gain in order to achieve low hardware complexity with reasonably good burst and random error performance. Threshold decoding is one method of achieving this aim. Majority-decision threshold decoding (ref. 1), is in terms of hardware, one of the simplest convolutional decoding schemes possible, and is applicable to a wide range of time-varying and fading channels. However, because the scheme is not optimum, some coding gain is lost.

In this paper we present a soft-decision majority threshold decoding scheme (ref. 2) that improves on the performance achievable with existing hard-decision decoders, thereby making up some of the lost coding gain, whilst still retaining the inherent hardware simplicity of threshold decoding. It has been shown (ref. 3) that the maximum increase

in coding gain that can be achieved by using soft-decision is about 2dB for infinite-level quantisation, and that the degradation involved in using equal-spacing 8-level quantisation (as assumed in this paper) is only 0.2dB. We therefore expect a maximum improvement of about 1.8dB for soft-decision majority threshold decoding when compared with existing hard-decision decoders.

In this paper we firstly outline hard-decision majority threshold decoding and then introduce our soft-decision scheme using a simple constraint length 2 code as an example. Next we describe our general method for soft-decision decoding of multiple error-correcting codes, using a constraint length 12 code as an example, and also present performance results. Finally, methods of applying the technique to majority-decodable block codes are discussed.

2. Hard-decision majority threshold decoding

A single-generator systematic convolutional code is one in which each information digit is encoded into V code digits (giving a message throughput rate of 1/V), the first of which is the unchanged information digit. In general, such a code is generated by a K segment generator sequence $g = g(1)\ g(2)\ g(4)\ \ldots\ g(2^{K-1})$, where K is the constraint length of the code in segments, and each segment contains V digits. For simplicity, we restrict our discussion in this paper to rate one-half codes.

Let us consider a rate one-half systematic code with constraint length K=2 segments, to review the basic hard-decision majority threshold decoding technique. The encoder for this simple code is shown in Fig. 1,

and consists of only a single one-bit delay element and a single modulo-2 adder (exclusive-OR gate). Given a sequence of information digits $x = x_{t-1}\ x_t\ x_{t+1} \cdots$, where t denotes the time unit of the information digit x_t, each information digit is encoded into two code digits c'_t and c''_t. $c'_t = x_t$ is the unaltered information digit x_t, and $c''_t = x_{t-1} \oplus x_t$ is a parity check sum based on the present information digit x_t and the K-1 = 1 previous information digits. For serial transmission the coded digits are sent to the channel in order $c'_t\ c''_t$ by appropriate action of the switch. The encoder/decoder configuration for this code is shown in Fig. 2. On the left of the diagram, the information digit x_t is encoded into c'_t and c''_t; in the middle, two noise digits n'_t and n''_t corrupt the coded digits c'_t and c''_t respectively; on the right is the decoder which realises the (hard-decision) single-error-correction capability of the code. The decoding action is explained with reference to the six points, a, b, c, S_1, S_2, and n'_{t-1}. The six points are interpreted as follows:

$$a = x_t \oplus n'_t$$

$$b = x_{t-1} \oplus n'_{t-1}$$

$$c = x_t \oplus x_{t-1} \oplus n''_t$$

$$S_1 = a \oplus b \oplus c = (x_t \oplus n'_t) \oplus (x_{t-1} \oplus n'_{t-1}) \oplus x_t \oplus x_{t-1} \oplus n''_t)$$

$$S_2 = (x_{t-1} \oplus n'_{t-1}) \oplus (x_{t-2} \oplus n'_{t-2}) \oplus (x_{t-1} \oplus x_{t-2} \oplus n''_{t-1})$$

$$\tilde{n}'_{t-1} = 1 \text{ if } S_1 = S_2 = 1$$

$$= \text{otherwise}$$

by cancelling information digits, S_1 and S_2 become:

$$S_1 = n'_t \oplus n''_t \oplus n'_{t-1}$$

$$S_2 = \qquad n'_{t-1} \oplus n''_{t-1} \oplus n'_{t-2} \qquad \ldots\ldots (1)$$

and it can be seen that the two parity check equations S_1, S_2 are orthogonal on the noise digit n'_{t-1}. Thus if a single error occurs anywhere in the 5 digit span covered by the orthogonal check sums, the only case when $S_1 = S_2 = 1$ is when $n'_{t-1} = 1$. In the decoder, the AND gate sends an estimate $\tilde{n}'_{t-1}$ to cancel the noise digit n'_{t-1} from the received digit $(x_{t-1} \oplus n'_{t-1})$, and thus produce an estimate $\tilde{x}_{t-1}$. From equation (1) it can be seen that if more than one error occurs in the 5 digit span covered by $\{S_1, S_2\}$, then the error correction capability of the code is exceeded and the decoded digit $\tilde{x}_{t-1}$ may be in error.

The decoder described above can be improved by the use of feedback. This is because if we are concerned with decoding x_{t-1} at the present moment, then x_{t-2} has already been decoded. We therefore have available an estimate of the noise digit n'_{t-2} before we decode x_{t-1}. Therefore S_2 can be simplified by feeding back $\tilde{n}'_{t-2}$ to cancel n'_{t-2} in equation (1). We may then replace S_2 with $\tilde{S}_2 = S_2 \oplus \tilde{n}'_{t-2} = n'_{t-1} \oplus n''_{t-1} \oplus n'_{t-2} \oplus \tilde{n}'_{t-2}$. If the estimate $\tilde{n}'_{t-2}$ is correct, that is $\tilde{n}'_{t-2} = n'_{t-2}$, then $S_2 = n'_{t-1} \oplus n''_{t-1}$. This means that provided the previously decoded digit was correct, the decoder check sums $\{S_1, S_2\}$ only span 4 digits, and can therefore correct a single error anywhere in 4 digits as opposed to 5 digits in the previous case. A decoder that makes use of past decisions to simplify S_2 to $\tilde{S}_2$ is called a feedback decoder, whilst a decoder that does not use past decisions is called a definite decoder.

In general, if it is possible to form a set of 2e parity check equations which are orthogonal on a specified noise digit, then it is possible to

build a hard-decision majority threshold decoder which can correct any combinations of e or fewer errors over one constraint span. Figure 3 shows the encoder/decoder arrangement for a triple error-correcting rate one-half (24,12) majority decoder which has K=12, and an effective constraint length of 24 digits within which 3 or fewer errors can be corrected. This decoder can achieve a coding gain of 1.85dB at a sink bit error rate of 10^{-5} on the binary symmetric channel (which is comparable to the (23,12) perfect Golay code), and can be built with only 16 standard integrated circuits (which is much less than that required to decode the Golay code).

3. Soft-decision majority threshold decoding

In this section we introduce our new method for soft-decision majority threshold decoding. Our basic approach is to derive a modified set of orthogonal check sums S_i^* which can be used to estimate each noise digit in the soft-decision sense.

Firstly, let us assume that each received digit is quantised into Q = 8 levels, and can therefore be expressed as a 3 digit binary number, or the BCD equivalent. For example, [000] = 0, [001] = 1, [010] = 2, ... [111] = 7. The x_t are therefore expressed as [000] when x_t = 0, or [111] when x_t = 1, in the soft-decision sense. The noise digits are expressed in a similar manner but can take any intermediate value between 0 and 7, that is, $0 = [000] \leq [n'_{t-j}] \leq [111] = 7$, where the square brackets indicate a quantised or soft-decision noise digit. Note that the most significant digit of a quantised digit is the hard decision digit itself.

For example, $[n'_{t-j}] = [010]$ implies $n'_{t-j} = 0$, and $[n'_{t-j}] = [110]$ implies $n'_{t-j} = 1$. Similarly, received digits are given by $[r'_{t-j}] = [x_{t-j}] \oplus [n'_{t-j}]$, and can take any value between 0 and 7. Let us define d_h to be the hard-decision minimum distance between the two halves of the initial code tree. The guaranteed error-correcting capability of the code over K segments is then e_h digits where e_h is the largest integer satisfying $e_h \leq (d_h - 1)/2$. The simple code used in section 2 has $d_h = 3$, and is therefore a single error-correcting code. In the soft-decision sense, the minimum distance of a code is given by $d_s = (Q-1) \times d_h$ levels, and its error correction capability is e_s soft-decision levels, where e_s is the largest integer satisfying $e_s \leq (d_s - 1)/2$. The simple example code therefore has $d_s = (8-1) \times d_h = 21$, and $e_s = 10$. We can now estimate the theoretical improvement to be gained by using soft-decision. In the hard-decision sense an error occurs when sufficient noise is added to a transmitted digit to form a received digit which lies on the opposite side of the 0/1 decision boundary. For example, if we transmit [000] (hard zero) and the noise is such that we receive [101] an 'error' in the hard-decision sense has occurred. Similarly, with transmitting [111] (hard one) and receiving [011]. Now, the minimum number of soft level errors required to cause an error in the hard-decision sense is 4. For example, transmit [000] receive [100]. As the simple code has a level correcting power of 10 levels, this indicates that integer $[\frac{10}{4}]$ = 2 'hard' errors can now be corrected. Thus if two hard errors occur and the total number of level errors amongst the 4 digits involved in

the decoding is ≤ 10, double error correction can be performed. Asymptotically, at high signal-to-noise ratios, soft-decision decoding therefore doubles the effective 'hard' correcting power of a code.

We now outline the soft-decision technique. Consider the orthogonal check sums for the example code (with feedback):

$$S_1 = n'_t \oplus n''_t \oplus n'_{t-1}$$
$$S_2 = \quad n'_{t-1} \oplus n''_{t-1} \qquad \ldots\ldots (2)$$

Our basic approach is to estimate, in the soft-decision sense, the value of <u>each</u> noise digit that appears in the orthogonal check sums, for two contradictory assumptions.

(a) the data bit being decoded now (r'_{t-1}) is <u>not</u> in error, that is $\tilde{n}'_{t-1} = 0$, and

(b) the data bit <u>is</u> in error, $\tilde{n}'_{t-1} = 1$. For each assumption a sum of the total number of level errors is formed. That is, $S^*_i = \sum_j [\tilde{n}_j]$ for all j involved in the decoding (= 4 in this case), and i = 0 for the assumption $\tilde{n}'_{t-1} = 0$, and i = 1 for $\tilde{n}'_{t-1} = 1$. The assumption which gives the smallest sum of estimated errors is chosen to be the correct decoding decision.

Note that the noise digits cannot be directly found but have to be derived from the received digits by a process of estimation as follows. Firstly, let us make the assumption of no error in r'_{t-1}, that is, $\tilde{n}'_{t-1} = 0$. Thus if the received digit at point b is $[r'_{t-1}] = [x_{t-1} \oplus n'_{t-1}] \leq 3$ = 'hard' zero, the estimate of $[n'_{t-1}]$, is

given by $[\tilde{n}'_{t-1}] = [x_{t-1} \oplus n'_{t-1}]$. If the received digit is $[x_{t-1} \oplus n'_{t-1}] \geq 4$, then $[\tilde{n}'_{t-1}] = 7 - [x_{t-1} \oplus n'_{t-1}]$. Corresponding values for r'_{t-1} in error are therefore $7 - [r'_{t-1}]$, and $[r'_{t-1}]$.

To estimate the remainder of the noise digits we need to know the result of each orthogonal parity check sum in the hard-decision sense. Consider first S_2. If $S_2 = 1$, that is, S_2 'fails' in the hard-decision sense, then we must assume that $\tilde{n}''_{t-1} = 1$, because we have assumed $\tilde{n}'_{t-1}=0$ in the hard-decision sense. Hence the estimate of n''_{t-1} in the soft-decision sense is $[\tilde{n}''_{t-1}] = [r''_{t-1}]$ if $[r''_{t-1}] \geq 4$, and $[\tilde{n}''_{t-1}] = 7 - [r''_{t-1}]$ if $[r''_{t-1}] \leq 3$. Conversely, if S_2 does not fail in the hard-decision sense, then $\tilde{n}''_{t-1} = 0$ and $[\tilde{n}''_{t-1}] = 7 - [r''_{t-1}]$ if $[r''_{t-1}] \geq 4$, or $[n''_{t-1}] = [r''_{t-1}]$ if $[r''_{t-1}] \leq 3$. Consider now S_1. If S_1 fails then either n'_t or n''_t is in error. We choose the assumption which gives the lowest number of level errors and then estimate the noise digits as previously. If S_1 does not fail we assume no errors in n'_t and n''_t. S^*_o is then formed by summing the noise estimates

$$S^*_o = [n'_t] + [n''_t] + [n''_{t-1}] + [n'_{t-1}].$$

The process is repeated for the assumption $\tilde{n}'_{t-1} = 1$ and S^*_1 is calculated. If $S^*_o \leq S^*_1$, then $\tilde{n}'_{t-1} = 0$, and if $S^*_o > S^*_1$, $\tilde{n}'_{t-1} = 1$.

Consider the following example. Let us assume that $x_t = x_{t-1} = 0$, that the noise digits are $[n'_{t-1}] = [101]$, $[n''_{t-1}] = [100]$, $[n'_t] = [001]$, $[n''_t] = [000]$, and that the decoder has not made any previous decoding errors. Note that as $n'_{t-1} = n''_{t-1} = 1$, a hard decision decoder would decode $\tilde{x}_{t-1} = 1$, thus giving a decoding error. Using the above soft-

decision procedure, however, x_{t-1} can be correctly decoded.

(1) Assume $\tilde{n}'_{t-1} = 0$. Hence the received digit $[r_{t-1}] = [x_{t-1} \oplus n'_{t-1}] = [101]$ is not in error and $[\tilde{n}'_{t-1}] = 7 - [101] = 2$ levels.

(2) $S_2 = 1 \oplus 1 = 0$ = no fail. We therefore assume $\tilde{n}''_{t-1}$ is not in error. Hence $[\tilde{n}''_{t-1}] = 7 - [r''_{t-1}] = 7 - 4 = 3$ levels.

(3) $S_1 = 0 \oplus 0 \oplus 1 = 1$ = fail. Hence we assume either n'_t or n''_t is in error. If we assume n'_t is in error than $[\tilde{n}_t] = 7 - [r'_t] = 7 - [001] = 6$ levels, as $[r'_t] \leq 1 \leq 3$. Also, n''_t is assumed not in error, giving $[\tilde{n}''_t] = 0$ levels, and a total of 6 level errors. If we assume the converse, n''_t in error, then $[\tilde{n}'_t] = 1$ level, and $[\tilde{n}''_t] = 7$ levels, giving a total of 8 level errors, we therefore assume that of the two $[\tilde{n}'_t]$ is more likely to be in error.

(4) We now calculate $S^*_o = 2 + 3 + (6+0) = 11$.

(5) Now assume that $\tilde{n}'_{t-1} = 1$, and that r'_{t-1} is in error. Hence $[\tilde{n}'_{t-1}] = [r'_{t-1}] = 5$.

(6) S_2 does not fail. Therefore as $\tilde{n}'_{t-1} = 1$, $\tilde{n}''_{t-1} = 1$ to cause this. Hence $[\tilde{n}''_{t-1}] = [r''_{t-1}] = 4$.

(7) S_1 fails. Therefore n'_t and n''_t must be assumed correct. Hence $[\tilde{n}'_t] = 1$, and $[n''_t] = 0$.

(8) $S^*_1 = 5 + 4\ (1+0) = 10$.

(9) Therefore $S^*_1 < S^*_o$ and we assume $\tilde{n}'_{t-1} = 1$, that is, $r'_{t-1} = 1$ is in error, and x_{t-1} is correctly decoded as $\tilde{x}_{t-1} = 0$.

4. Soft-decision multiple error threshold decoding

In this section the approach used for the constraint length 2 code is

generalised to deal with multiple-error correcting convolutional codes, using a constraint length 12 code as an example.

Figure 3 shows a triple-error-correcting hard-decision threshold decoding system. It is possible to form 2e = 6 check sums orthogonal on the noise digit n'_{t-11} as follows.

$$S_{t-11} = n'_{t-11} \oplus n''_{t-11}$$

$$S_{t-5} = n'_{t-11} \oplus n'_{t-5} \oplus n''_{t-5}$$

$$S_{t-4} = n'_{t-11} \oplus n'_{t-10} \oplus n'_{t-4} \oplus n''_{t-4}$$

$$S_{t-2} = n'_{t-11} \oplus n'_{t-9} \oplus n'_{t-8} \oplus n'_{t-2} \oplus n''_{t-2}$$

$$S_t \oplus S_{t-3} \oplus S_{t-7} = n'_{t-11} \oplus n'_{t-6} \oplus n'_{t-3} \oplus n'_t \oplus n''_{t-3} \oplus n''_{t-7} \oplus n''_t$$

$$S_{t-1} \oplus S_{t-8} \oplus S_{t-10} = n'_{t-11} \oplus n'_{t-7} \oplus n'_{t-1} \oplus n''_{t-1} \oplus n''_{t-8} \oplus n''_{t-10}$$

Our basic approach is now to estimate the algebraic sum of a set of 2e+1 soft-decision noise sums, one for each orthogonal check sum, and one for $[\tilde{n}'_{t-11}]$, and compare this to a fixed threshold value T = (Q-1)(2e+1)/2. Then if $S^*_o > T$ we decode $\tilde{n}'_{t-11} = 1$, and if $S^*_o \leq T$, $n'_{t-11} = 0$.

The method outlined here differs from the scheme detailed in the last section in that only one noise digit per orthogonal check sum is estimated in the soft-decision sense. This digit is always the 'worse' digit (that is the one nearest the 0/1 boundary) in each orthogonal sum, excluding the digit on which all sums are orthogonal. The reason for doing this can be seen with reference to the example given in the last section. In that example the estimated value of n''_t did not change for

the two assumptions $\tilde{n}'_{t-1} = 0$, and $\tilde{n}'_{t-1} = 1$. The value $\tilde{n}''_t$ therefore played no part in deciding which sum is the greater S^*_o or S^*_1, and can therefore be omitted. In general then, the only noise estimate which will change for a given orthogonal check sum result is the 'worst' digit in the check sum set.

Also, as a consequence of the above, and neglecting estimates which do not change, it can be seen that the noise estimates for each orthogonal check sum are complements of each other for the two assumptions $\tilde{n}'_{t-1} = 0$, and $\tilde{n}'_{t-1} = 1$, that is, they add to $(Q-1) = 7$. For the triple-error correcting code this means that the two sums S^*_o and S^*_1 add to $(Q-1)(2e+1) = 49$, and hence $T = 24$. It is therefore only necessary to compute S^*_o, and compare its value to $(Q-1)(2e+1)/2$, because $S^*_1 = (Q-1)(2e+1) - S^*_o$.

Consider the following example for the constraint length 12 code. We assume $x_t = x_{t-1} = \ldots\ldots = x_{t-11} = 0$ and that no previous decoding error has been accepted. Also, $[n'_{t-11}] = [110]$, $[n'_{t-5}] = [101]$, $[n'_{t-10}] = [100]$, $[n''_{t-4}] = [011]$, $[n'_{t-9}] = [100]$, $[n'_{t-8}] = [001]$, $[n'_{t-1}] = [010]$, and all other noise digits are [000]. Note that this gives 4 hard decision errors. The estimation of S^*_o is performed as follows.

(1) $[\tilde{n}'_{t-11}] = 7 - [r'_{t-11}] = 7 - [110] = 1$, because we assume $\tilde{n}'_{t-11} = 0$.

(2) $S_{t-11} = 1 \oplus 0 = 1$. Hence $\tilde{n}''_{t-11} = 1$ and $[\tilde{n}''_{t-11}] = 7 - [r''_{t-11}] = 7 - [000] = 7$.

(3) $S_{t-5} = 1 \oplus 1 \oplus 0 = 0$. Hence 'worst' not in error. That is, $\tilde{n}'_{t-5} = 0$ and $[\tilde{n}'_{t-5}] = 7 - [r'_{t-5}] = 7 - [101] = 2$.

(4) $S_{t-4} = 1 \oplus 1 \oplus 0 = 0$. Hence 'worst' not in error. That is, $\tilde{n}''_{t-4} = 0$ and $[\tilde{n}''_{t-4}] = [r''_{t-4}] = [011] = 3$.

(5) $S_{t-2} = 1 \oplus 1 \oplus 0 \oplus 0 \oplus 0 = 0$. Hence, 'worst' not in error. That is, $\tilde{n}'_{t-9} = 1$ and $[n'_{t-9}] = 7 - [r'_{t-9}] = 7 - [100] = 3$.

(6) $S_t \oplus S_{t-3} \oplus S_{t-7} = 1 \oplus 0 \oplus 0 \oplus 0 \oplus 0 \oplus 0 = 1$. Hence 'worst' is in error. That is, $\tilde{n}'_t = 1$ and $[\tilde{n}'_t] = 7 - [r'_t] = 7 - [000] = 7$.

(7) $S_{t-1} \oplus S_{t-8} \oplus S_{t-10} = 1 \oplus 0 \oplus 0 \oplus 0 \oplus 0 \oplus 0 = 1$. Hence, 'worst' is in error. That is $\tilde{n}'_{t-1} = 1$ and $[\tilde{n}'_{t-1}] = 7 - [010] = 5$.

(8) Therefore, $S^*_o = 1 + 7 + 2 + 3 + 3 + 7 + 5 = 28 > T = 24$ and hence $\tilde{n}'_{t-11} \neq 0$ but $\tilde{n}'_{t-11} = 1$, giving a correct decoding $\tilde{x}_{t-11} = 0$.

5. Decoder Design

The increase in complexity required to implement the soft-decision algorithm is not excessive. Essentially, the only complex items in the circuitry are a BCD adder capable of adding the (2e+1) noise estimates, BCD comparators for each orthogonal check sum, and a threshold comparator.

Figure 4 shows a soft-decision decoder for the simple K=2 code. The essential items in the design are the same for this code or a multiple error-correcting code, and are as follows.

Delay bistables (denoted D): are used to store both hard-decision and soft-decision digits.

Quantizers: these provide 8 level quantization of the received digits.

Basic Soft-Error Processors (BSEP): these devices output the number of

level errors in the received digit, assuming that the received digit is not in error, in the hard-decision sense. This is therefore a simple logic device with the following truth table.

INPUT			OUTPUT		
Quantized soft-decision digit			Binary		Levels
most significant (hard) bit		LSB	MSB	LSB	
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	1	0	2
0	1	1	1	1	3
1	0	0	1	1	3
1	0	1	1	0	2
1	1	0	0	1	1
1	1	1	0	0	0

Comparators: these operate on the outputs of the basic soft-error processors in such a way that for any two BCD inputs, the greatest BCD number is output. By this means the 'worst' digit in an orthogonal check sum is identified. Note that in figure 4 only one such comparator is required. In general, however, more than two noise digits are involved in a check sum, and therefore a comparator with as many inputs as there are digits in the orthogonal check sum, minus one, are required. Such a device is easily constructed by simply iterating the basic 2-input device as many times as required.

Compliment Processors (CP): these devices operate under control of a

hard-decision orthogonal check sum, and either allow a soft-decision estimate [i] to be transmitted unalterated through them, or else compliment the estimate to 7-[i]. The device is again a simple logic element with the following truth table.

INPUT			OUTPUT	
hard-decision control check sum input	soft-decision digits		binary	levels
	binary	levels		
0	00	0	000	0
0	01	1	001	1
0	10	2	010	2
0	11	3	011	3
1	00	0	111	7
1	01	1	110	6
1	10	2	101	5
1	11	3	100	4

BCD Adder: In general, an adder with (2e+1) inputs that is capable of adding input numbers in the range 0→7 is required. Finally, a threshold device which outputs a 1 if the adder output is > T, and a 0 otherwise, is required.

6. Soft-decision threshold decoding of block codes

The soft-decision decoding algorithm outlined in the previous sections can also be used to decode one-step or L-step majority-logic decodable block codes, with very little modification. Consider as an example the (15,7) cyclic double-error correcting one-step decodable code which has

a generator polynomial $g(x) = x^8 + x^7 + x^6 + x^4 + 1$. It is possible to form 2e=4 parity check sums orthogonal on the noise digital n_{14} as follows:

$$A_1 = n_{14} \oplus n_{12} \oplus n_{11} \oplus n_3$$

$$A_2 = n_{14} \oplus n_{13} \oplus n_5 \oplus n_1$$

$$A_3 = n_{14} \oplus n_6 \oplus n_2 \oplus n_0$$

$$A_4 = n_{14} \oplus n_{10} \oplus n_8 \oplus n_7$$

Figure 5 shows a Type II decoder for this code, which operates as follows.

(1) With gate 1 on and gate 2 off, the received block is read into the buffer register.

(2) The 2e checksums orthogonal on n_{14} are formed and, the threshold gate outputs $\tilde{n}_{14} = 1$ if a clear majority of inputs are one.

(3) The estimate $\tilde{n}_{14}$ is then added modulo-2 to the received digit to form the correct output digit $\tilde{x}_{14}$.

(4) The register is shifted once with gate 1 off, and gate 2 on. Hence the corrected digit $\tilde{x}_{14}$ is fed back, thereby removing n_{14} from the equations, provided that a correct bit decoding has been made. The exclusive OR gates now form 4 check-sums orthogonal on n_{13}, which is now at the right-hand end of the buffer register, and decoding is again accomplished via the threshold gate.

(5) Decoding continues on a bit-by-bit basis, until all the corrected information bits have been output.

It can be easily seen that the soft-decision algorithm can operate on this decoding scheme in exactly the same way as with convolutional codes. That is, for each information bit the sum S_o^* of (2e+1) soft-decision noise estimates is computed: one for $[\tilde{n}_{14}]$ given the assumption $\tilde{n}_{14} = 0$, and one for each orthogonal check-sum based on the 'worst' digit in the checksum. The sum of estimates is then compared with the fixed threshold T, and n_{14} is decoded. Similarly, each information digit in the block is decoded on a bit-by-bit basis by simply shifting the quantized received block in the buffer register.

The above type of soft-decision decoding is again sub-optimum in that the full 2dB of soft-decision coding gain is not achieved. It is possible however to considerably improve the decoding performance, at the expense of more complex control circuitry, by modifying the scheme as follows.

The above block code soft-decision algorithm proceeds by decoding the information bits in their natural order, that is, we estimate the noise digits in the order n_{14}, n_{13}, ..., n_{n-k+1}, n_{n-k}. However, as decoding decisions are fed-back, thus affecting future decoding decisions, it would be wise to decode all the bits in the block in order of decreasing 'confidence'. We can form an estimate of the 'confidence' of each decoding decision by comparing the sum S_o^* with the threshold value T. The further the value of S_o^* is away from the value of T, that is, the greater $|T-S_o^*|$, the greater the confidence we have of a correct bit decoding decision. A decoder operating on this scheme would therefore

calculate the sum $|T-S_o^*|$ for each bit in the block, and store a list of the order in which bits are to be decoded, based on the increasing value of $|T-S_o^*|$. The decoder then decodes the bits in the order indicated by successively shifting them into the decoding position, that is, the right-most end of the buffer register. In this way, a decoder which realises most of the 2dB soft-decision coding gain available can be built for a wide variety of majority-logic decodable codes.

7. Performance

Figure 6 shows the performance of various block and convolutional decoding schemes using the algorithm, under conditions of additive white Gaussian noise. It can be seen that useful coding gains over that achievable with hard-decision decoding are possible. Note that all curves are corrected for rate, that is, plotted versus normalised signal-to-noise ratio (energy per information bit E_b/N_o noise density), to ensure a valid comparison between uncoded and coded transmission.

8. References

1. Massey, J.L.: Threshold Decoding, M.I.T. Press, Cambridge Mass., 1963.

2. Goodman, R.M.F., and Ng, W.H.: 'Soft-decision threshold decoding of convolutional codes', I.E.R.E. Conf. Proc. No. 37, 1977.

3. Wozencraft, J.M., and Jacobs, I.B.: Principles of Communication Engineering, Wiley, New York, 1965.

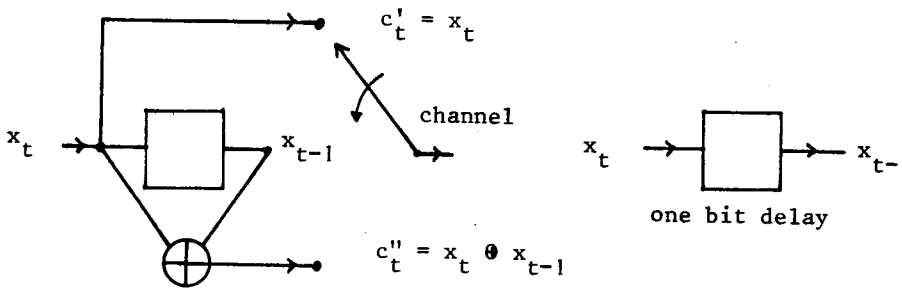

Figure 1.A simple convolutional encoder g = 11 01

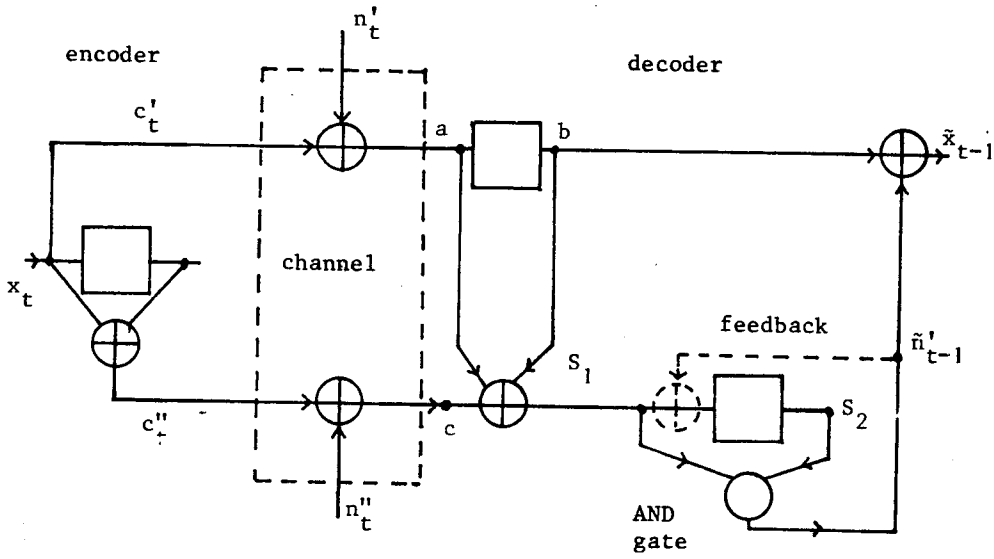

Figure 2. A K=2 convolutional coding system

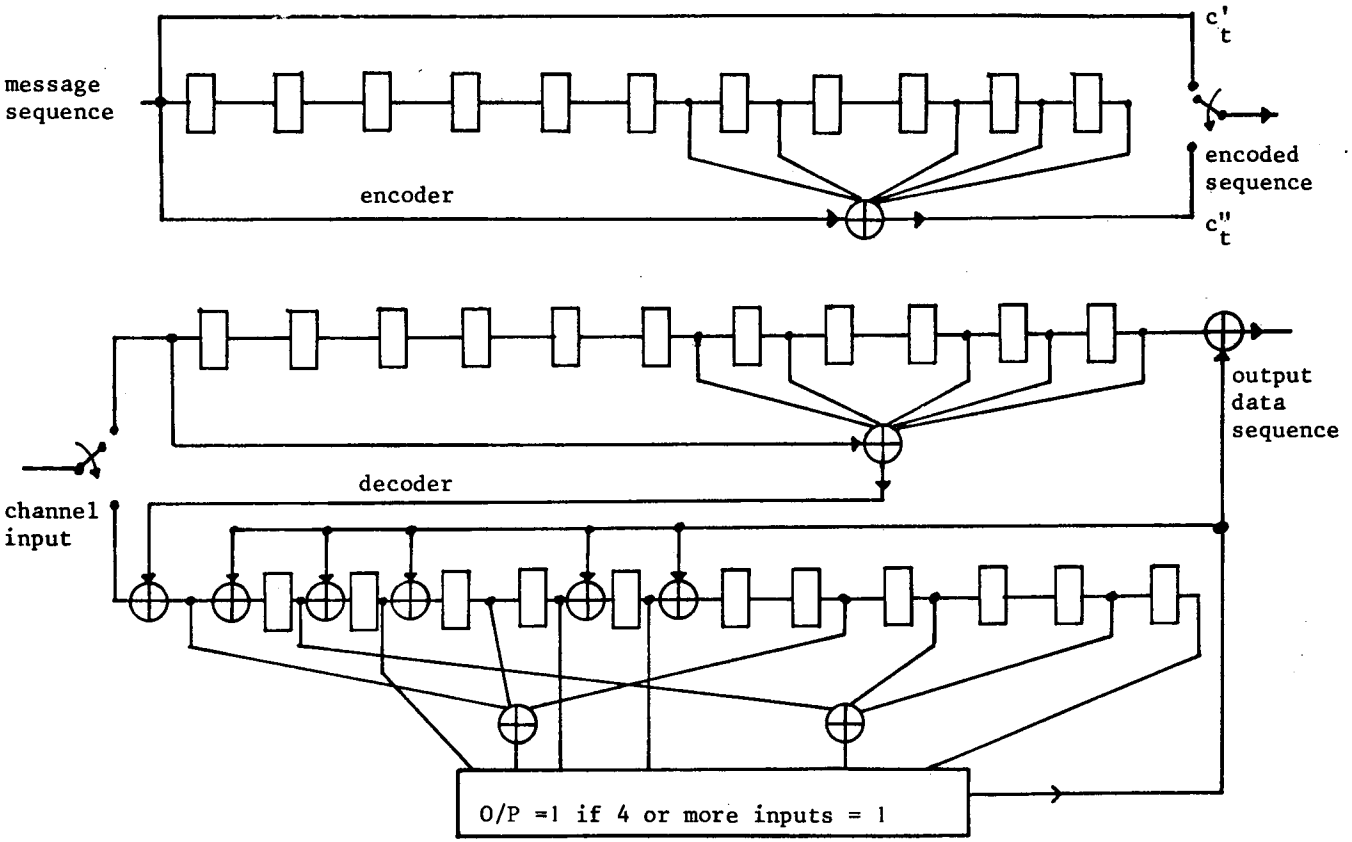

Figure 3 . A triple error correction threshold coding system

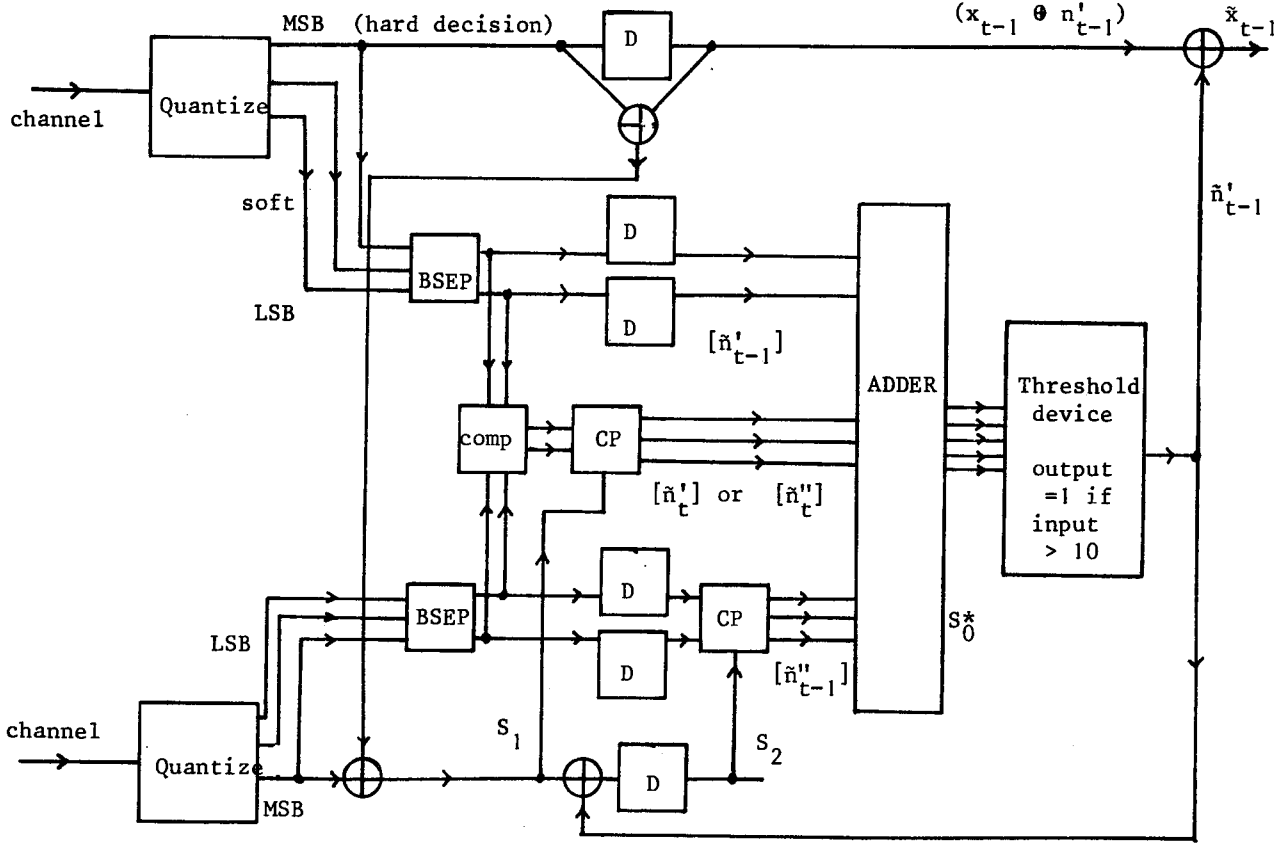

Figure 4. Soft-decision threshold decoder

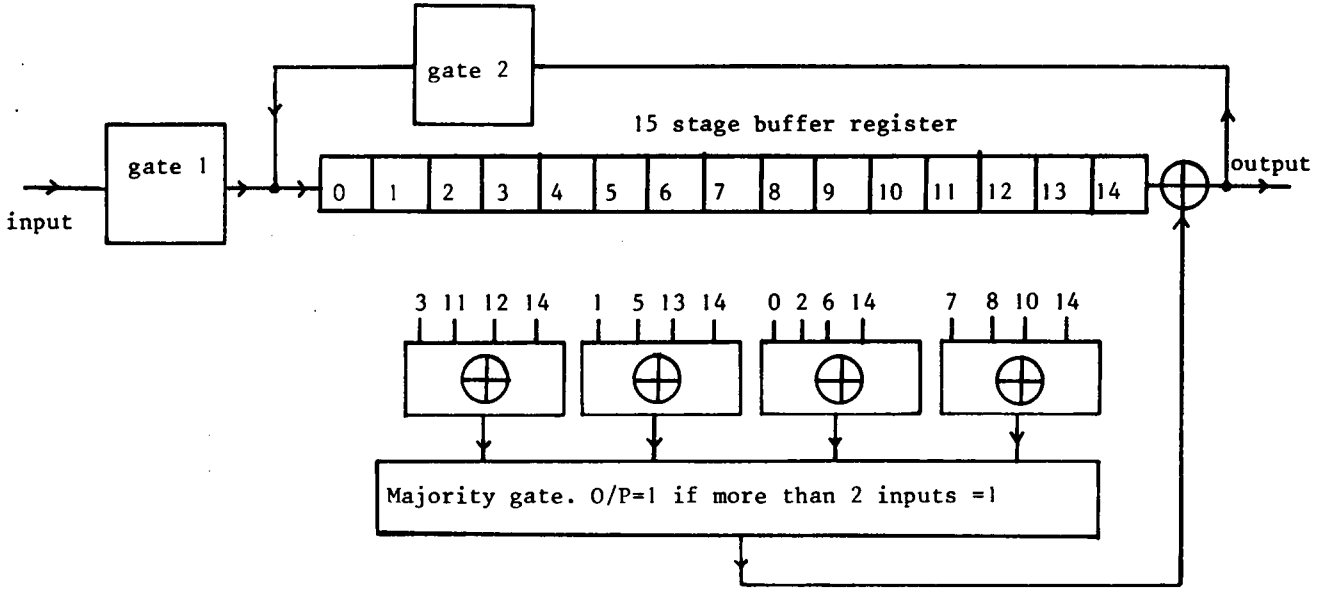

Figure 5. Type II one-step majority decoder for the (15,7) cyclic code

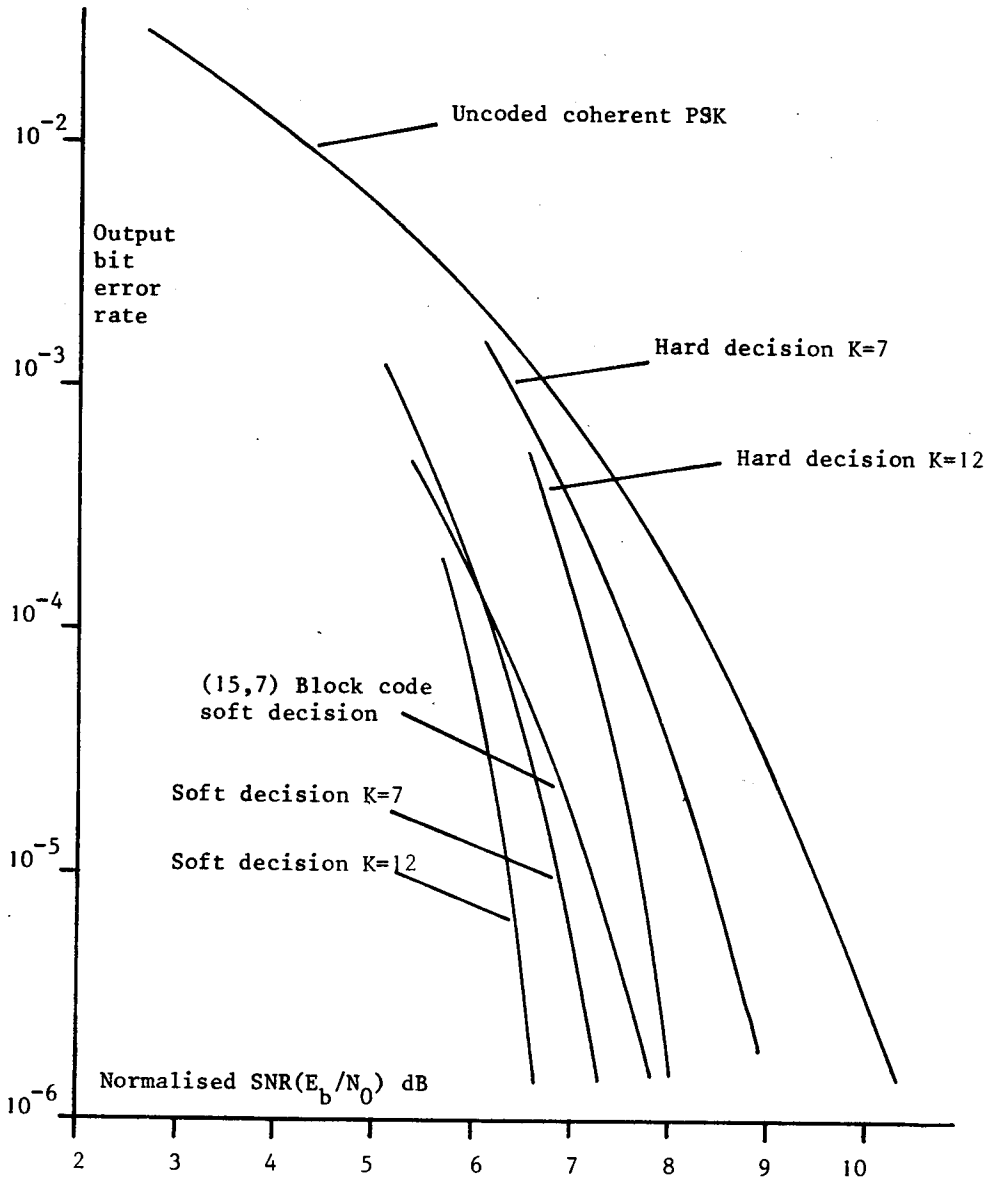

Figure 6. Performance curves