BURST ERROR CORRECTING CONVOLUTIONAL CODES

R.M.F. Goodman

Summary

This paper reviews several burst-error correction techniques which use convolutional codes, and introduces some new developments being studied at Hull.

1. Introduction

Most practical data transmission channels exhibit error statistics which are non-Gaussian and bursty. For example, telephone data channels are subject to impulsive noise which causes bursts of errors. Such a channel may be considered 'classically bursty' in that bursts are separated by very long error-free gaps. Another bursty channel is the H.F. data transmission channel. Here bursts are diffuse, that is, there is a higher background level of errors due to the effects of interference, fading, and multipath: and bursts can really only be classified as periods in which the overall density of errors rises past some aribtrary threshold. In such systems there is never any really lengthy error-free period.

Errors also occur in a bursty fashion when computer data is stored on magnetic media such as tape or floppy disk. In the case of tape, data is stored in parallel tracks at densities of the order of a thousand bits per inch, and there are several common causes of errors. These include oxide particles on the head or the tape, voids or drop-outs in the oxide coating of the tape, dust or other foreign particles on the tape and head, and tape damage due to handling. These defects usually only affect one track at a given time, but cause bursts of errors on that track. Similarly, floppy disks have a high density of information. For example a single density 5¼" floppy disk can hold a million bits of information. Errors again occur in bursts due to the magnetic nature of the medium and the high density of data storage in terms of bits per inch.

In order to deal with such channels it is necessary to use error control codes that are capable of dealing with bursts of errors. In addition, it is usually not possible to classify a channel as 'purely bursty' so that some random-error-correction power is required. In this paper we look at several burst correction techniques using convolutional codes. An advantage of using convolutional codes is that it is easy to design good burst correcting codes, particularly for the correction of long bursts, with a wide variety of code rates.

2. Convolutional Codes and Burst Correction

A convolutional code is one in which the encoder accepts a block of k_o message bits as input and outputs a segment of $n_o > k_o$ coded bits. The n_o bits are produced by mod -2 additions on message bits over K blocks, where K is the constraint length of the code in segments. If the

Dr. Goodman is with the Department of Electronic Engineering, University of Hull, HULL. HU6 7RX.

k_o message bits are reproduced exactly in the n_o bit segment the code is systematic. Alternatively, the code can be expressed as a (n,k) code where $n = Kn_o$, $k = Kk_o$ and n is the constraint length in bits. The code rate R is given by $R = k_o/n_o$.

As far as burst correction is concerned we are interested in the burst correcting ability of the code b, where b is the length of a burst in bits. A burst of length b is usually defined relative to a guard space g, which is a sequence of all-zero error digits. A burst is defined as a sequence of error bits starting and ending with a '1' and separated from other bursts by at least g '0' digits. For a good scheme, therefore, the ratio g/b should be as small as possible.

An upper bound[1] on b is given by

$$b \leqslant \frac{K(1-R)n_o}{(1+R)} + n_o - 1 = \frac{2K}{3} + 1$$

for a ½ rate code, and computer generated or constructional codes can approach this bound very closely.

A well known bound[2] on burst-correcting capability relative to guard space is given by

$$\frac{g}{b} \geqslant \frac{1+R}{1-R} = 3 \text{ for a ½ rate code.}$$

and again many practical codes approach this.

There is, therefore, no real problem in finding codes which approach the optimum burst correcting power. What is a problem is choosing the right decoding scheme for a particular burst noise situation.

3. Burst-Correcting Decoding Schemes

In this section we review some of the popular burst-error-correcting convolutional coding schemes.

3.1 Detection and Retransmission

In this scheme the convolutional code is used purely as an error detecting code, and uses a feedback channel to request re-transmission of a coded 'block' that has been detected to be in error. The decoder operates as shown in figure 1, and basically consists of a replica encoder. The 'syndrome' is formed by modulo-2 addition of the received parity and calculated parity. If a syndrome bit equal to '1' appears at any time during the reception of an incoming block, that block is flagged as being in error and a retransmission requested.

The constraint length K of code is usually chosen to be much less than the total 'block' length of the transmitted packet. This is because the encoder must be 'flushed out' with K-1 zeros after the last message bit, and before the 1st message bit in the next 'block'. Thus there is an overhead of K-1 bits per block. The probability of undetected error, however, can be made very low with such a scheme. If the code rate is given by $\frac{k_o}{n_o} = R$, then the probability of undetected error is the probability of an all-zero syndrome which is $2^{-(n_o-k_o)K}$. For example, a ½-rate code with decoding constraint length K = 20 segments (40 bits) would give a probability of undetected error of 2^{-20} i.e. 10^{-6}. In general, in order to detect any burst of length b or less, $(n_o - k_o)K \geqslant b$.

In many cases (such as magnetic tape storage) a 'repeat request' of the original data is not possible and error detection schemes can not be used.

3.2 Interleaving

Interleavingis a powerful technique for randomising bursts of errors, and thus enabling a powerful random-error-correcting convolutional decoding scheme such as Viterbi decoding or sequential decoding to be used under bursty conditions.

There are two ways of interleaving a convolutional code. Firstly symbol interleaving. This is performed by simply reading the coded sequence into the rows of an array with λ columns and M rows, where λM is the total packet length in bits. The bits are then read out to the channel sequentially by columns, and the inverse operation performed at the decoder. Thus bits that were adjacent on the channel are separated by λ bits at the decoder. By this means a t-error random correcting code with burst correcting ability $b \leq t$ can be interleaved to correct any combination of t or fewer bursts of length λ or less over λK segments, or single bursts of length λb.

Secondly, a convolutional code can be segment-interleaved. That is, n_0-bit segments are separated by λ segments = $n_0\lambda$ bits. In this case we can think of the data stream as being sent to λ parallel encoders whose outputs are segment interleaved sequentially. This method cannot be used if $b < n_0$ in the original code. It is, however, easier to implement than symbol interleaving.

3.3 Burst Correcting Decoders

There are two main classes of 'pure' constructional burst correcting codes. These are BPM (Berlekamp-Preparata-Massey) codes, and Iwadare codes.[1] These codes can be used to construct reasonable power burst-correcting codes by interleaving the basic codes of each class. In general, however, it has been shown that computer generated codes perform better. Also, it is always possible to produce good long burst-correcting codes by interleaving a good short code.

In general form of a decoder for a computer generated burst-correcting ½-rate code can take the form shown in figure 2. The decoder consists of a received data register and a syndrome register. The read only memory is 'addressed' by the syndrome and outputs a '1' only if the syndrome corresponds to a correctable burst with an error in the right-hand end of the data register. The '1' corrects the data bit about to be output, and is also fed-back to remove the effect of the error from the symdrome. The constraint length of the code is limited by the size of the ROM, but again interleaving can be used to produce a longer effective constraint length.

3.4 Burst and Random Error Correction

In many cases it is not possible to define a channel as purely 'burst' or purely 'random'. A channel may exhibit a predominantly bursty nature against a background of random errors, or may have high error density bursts separated by medium density 'gaps'. In any of these cases it is necessary to have some random-error-correcting power in the coding scheme. The pure burst correcting schemes described so far have very poor random correcting power, and are thus unsuitable for channels such as these.

The first scheme which exhibits both random and burst correcting power is diffuse threshold decoding.[3]

A typical decoder is shown in figure 3, and contains a replica encoder plus a syndrome register. The taps on the syndrome register are parity checks which are orthogonal on the error digit at the right hand end of the message bits register. Thus an error in this position will cause all four check sums to go to '1'. An additional error will cause one of the sums to revert to a '0' but a clear majority of the sums are still '1'. The decoder is therefore intrinsically capable of correcting double errors. By separating the encoder taps by $\approx\beta$ bits the decoder achieves its burst correcting power. This is because no burst of length $b = 2\beta$ can cause more than two check sums to fail unless one of the errors is at the right hand end of the data register. The diffuse code therefore has a random correction power of 2, and a burst power of 2β relative to a guard space of $g = 6\beta+2$. Notice that $g/b \doteq 3$ for β large, indicating that the code is very nearly optimum. By extending this principle to threshold decodable codes with greater power it is possible to produce a wide range of good burst and random correcting codes.

The second technique is due to Gallager and is usually referred to as the Gallager burst decoder.[3] The decoder can operate in an adaptive random/burst mode, and is again based on a random error correcting threshold decodable code of constraint length $K = m+1$. Figure 4 shows such a decoder. The decoder uses the weight of the orthogonal checks to decide whether to move ahead without change, to perform a correction, or to shift to the other mode. The decoder operates as follows. In the normal or 'Random' mode the decoder simply corrects errors before they get into the M stage register, and through to the output. If a long dense burst of errors occurs the correction power of the code will be exceeded, but the code is designed so that this situation canbe detected with a high probability. Thus it is highly probable that code overload will be detected before erroneous data has left the decoder. At this point the decoder assumes that it has detected the start of a burst of length 2N or less, half of which is confined to the N stages of the replica encoder. If no errors are in the first m stages of the replica encoder then the syndrome bits S are a good estimate of the error bits in the burst, and these are simply allowed through to the output, to correct the burst. Again feedback is used to remove the effects of errors from the syndrome, and when successive syndrome bits revert to all zeros, the decoder returns to the random mode. The Gallager decoder can correct most bursts of length 2N relative to a guard space of 2N+m. Note that $g/b = 2N/(2N+m) \doteq 1$ for practical values of N. Thus the burst correcting power exceeds the bound previously mentioned, but only at the expense of not being able to correct <u>all</u> bursts of $\leqslant 2N$.

The Gallager decoder operates best on a channel which has dense long bursts separated by long error free gaps. The diffuse decoder operates best on a diffuse burst channel with a high background density of random errors.

4. <u>More Powerful Schemes</u>

At Hull we are investigating various methods of improving the performance of burst correcting convolutional coding schemes.

Firstly, we have devised schemes for using soft-decision information in diffuse threshold decoders and in Gallager decoders.[4] In a hard-decision error-control-coded binary data transmission system the receiver/demodulator makes a hard 0/1 decision on each incoming data signal before feeding the demodulated bit to the error-correction decoder. For example, in a multiphase modulation system a 'hard' decision is made at each phase boundary. This procedure results in a degradation of the channel decoder's performance. A soft-decision demodulator, on the other hand, assigns a 'confidence' value to each output bit, in addition to the 'hard' binary 0 or 1 decision. In essence this means that each demodulated bit is quantised into $Q > 2$ levels, rather than $Q = 2$ levels as in the hard-decision case. This confidence information can then be used to improve the error-correction decoder's performance (in terms of lower output bit error rate) without incurring any further redundancy penalty.

We have shown[5] that significant improvements in performance are achievable if soft-decision decoding is implemented on bursty channels such as the H.F. channel. In particular, the multiple burst performance of these decoders is very significantly improved. For example, when soft-decision was applied to the diffuse threshold decoder described earlier, in a bit error rate environment of approximately 1 in 16 the output error rate was improved by a factor of approximately two.

Secondly, we are investigating the use of convolutional burst correction schems for multi-track magnetic tape storage. We have devised a scheme which may be called convolutional product coding. This is shown in figure 5. Each data track of the tape (for example 8) is encoded with a burst correcting convolutional code of a suitable rate. The ninth track is a transverse parity check on the 8 data tracks, and is also encoded with the same convolutional burst correcting code. The decoder essentially consists of nine separate decoders. Decoding proceeds as follows. Under normal conditions each decoder decodes the bits of its track in a segment by segment manner. When all the data bits in a particular transverse line have emerged from the nine decoders a check on the transverse parity is made. If the parity is correct then this line of transverse data is output as ocrrect. If a burst which is confined to a single track occurs then the code can normally correct this and proceed. The code is, however, designed to detect overload with a high probability, and if a bad burst occurs the decoder will refuse to decode rather than make a decoding error. In this case decoding proceeds on the other eight tracks in such a way that the other decoders 'overtake' the 'stuck' decoder by one segment. The transverse parity can then decide what the data bits about to be output by the 'stuck' decoder should be. The bits are corrected and the 'stuck' decoder can usually proceed. If not, a similar procedure occurs. In this way, the decoders can 'help' each other, and by this means long bursts can be effectively corrected.

5. References

1. Peterson, W.W., and Weldon, E.J.: 'Error-Correcting Codes' (MIT Press, 1972).

2. Gallager, R.G.: 'Information theory and reliable communication' (Wiley, New York, 1968).

3. Kohlenberg, A. and Forney, G.D., Jr: 'Convolutional coding for channels with memory'. IEEE Trans. on Inf. Th., vol. IT-14, no. 5, Sept. 1968, pp. 618-626.

4. Goodman, R.M.F.: 'Soft-decision threshold decoders', Algebraic Coding Theory and Applications, Ed. G. Longo, C.I.S.M., No. 258, Springer-Verlag, 1979, pp. 423-447.

5. Farrell, P.G. and Goodman, R.M.F.: 'Soft-decision error control coding for H.F. data transmission', Proc. IEE, Vol. 127, Pt. F, No. 5, Oct 1980.

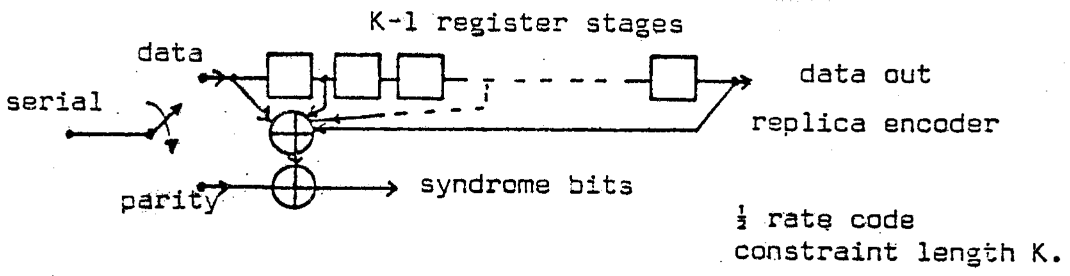

Fig.1. Burst Detection

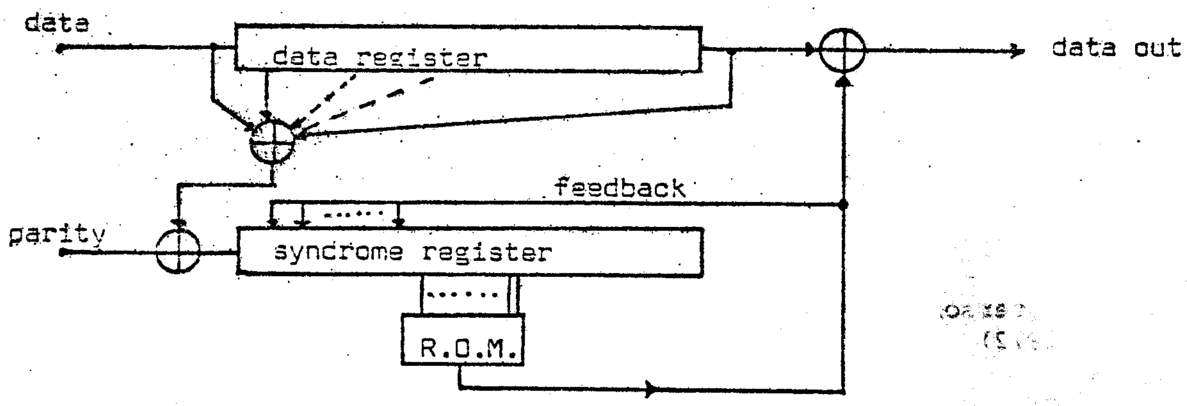

Fig.2. General Burst Decoder

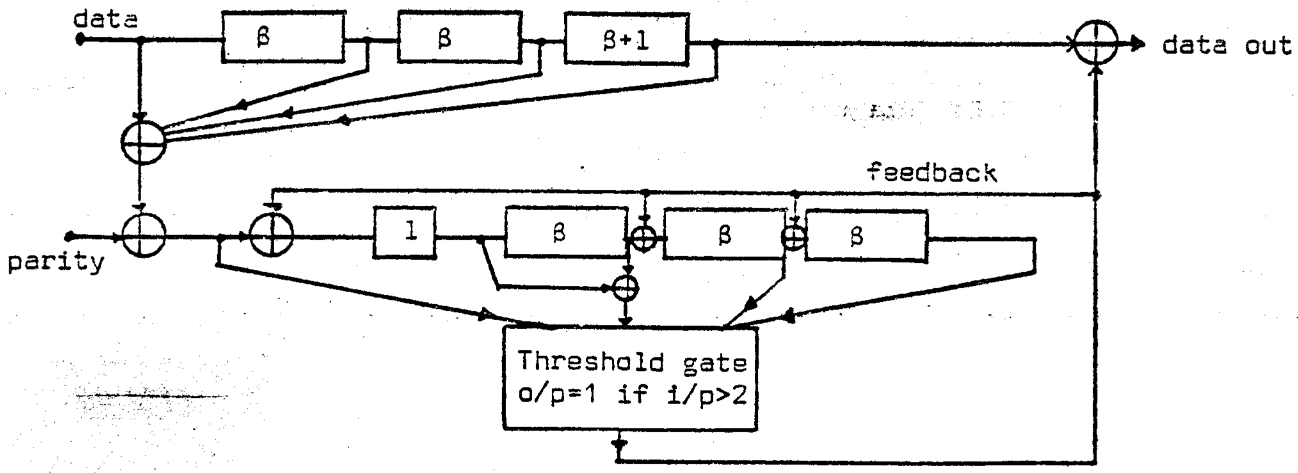

Fig.3. Diffuse Threshold Decoder

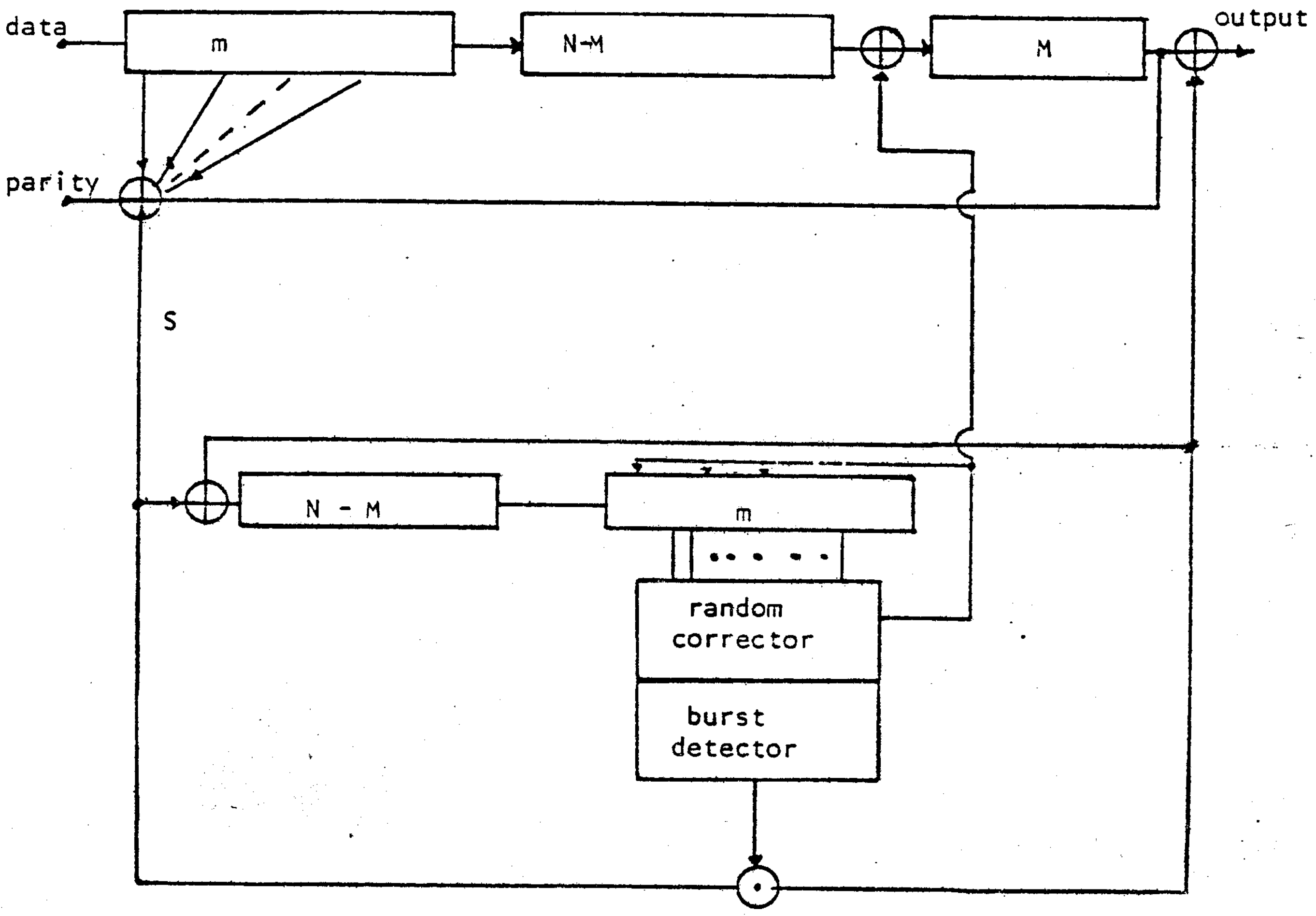

Fig.4. Gallager Decoder

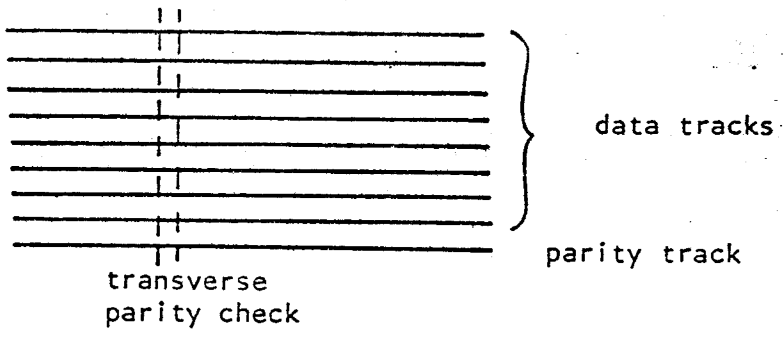

Fig.5. Convolutional Product Decoder