Module 6

Channel Coding

Lesson 33

Introduction to Error Control Coding

After reading this lesson, you will learn about

- ***Basic concept of Error Control Coding;***
- ***Categorizes of error control processes;***
- ***Factors describing FEC codes;***
- ***Block Codes – Encoding and Decoding;***

The primary function of an error control encoder-decoder pair (also known as a codec) is to enhance the reliability of message during transmission of information carrying symbols through a communication channel. An error control code can also ease the design process of a digital transmission system in multiple ways such as the following:

a) The transmission power requirement of a digital transmission scheme can be reduced by the use of an error control codec. This aspect is exploited in the design of most of the modern wireless digital communication systems such as a cellular mobile communication system.

b) Even the size of a transmitting or receiving antenna can be reduced by the use of an error control codec while maintaining the same level of end-to-end performance [example: VSAT (Very Small Aperture Terminal) network terminals].

c) Access of more users to same radio frequency in a multi-access communication system can be ensured by the use of error control technique [example: cellular CDMA].

d) Jamming margin in a spread spectrum communication system can be effectively increased by using suitable error control technique. Increased jamming margin allows signal transmission to a desired receiver in battlefield and elsewhere even if the enemy tries to drown the signal by transmitting high power in-band noise.

In this section we present a short overview of various error control codes.

Since C. E. Shannon's pioneering work on mathematical theory for digital communications in 1948-49, the subject of error control coding has emerged as a powerful and practical means of achieving efficient and reliable communication of information in a cost effective manner. Suitability of numerous error control schemes in digital transmission systems, wire line and wireless, has been studied and reported in detail in the literature.

The major categories of activities on error control coding can broadly be identified as the following:

a) to find codes with good structural properties and good asymptotic error performance,

b) to devise efficient encoding and decoding strategies for the codes and

c) to explore the applicability of good coding schemes in various digital transmission and storage systems and to evaluate their performance.

The encoding operation for a (n, k) error control code is a kind of mapping of sequences, chosen from a k-dimensional subspace to a larger, n-dimensional vector space of n-tuples defined over a finite field and with n > k. Decoding refers to a reverse mapping operation for estimating the probable information sequence from the knowledge of the received coded sequence. If the elements (bit, dibit or a symbol made of group of bits) of the message sequence at the input to the encoder are defined over a finite field of q_i elements and the sequence elements at the output of the encoder are defined over (same or a different) finite field with q_o elements, the code rate or 'coding efficiency' R of the code is defined as:

$R = (L_{in} \log_2 q_i) / (L_{out} \log_2 q_o)$, where L_{in} and L_{out} denote the lengths of input and output sequences respectively. The code rate is a dimensionless proper fraction.

For a binary code, $q_i = q_o = 2$ and hence, $R = L_{in} / L_{out}$. A (7,4) Hamming code is an example of a binary block code whose rate R = 4/7. This code will be addressed later in greater detail. For an error correction code, R < 1.0 and this implies that some additional information (in the form of 'parity symbol' or 'redundant symbol') is added during the process of encoding. This redundant information, infused in a controlled way, help us in decoding a received sequence to extract a reliable estimate of the information bearing sequence.

Now, it is interesting to note that the purpose of error control can be achieved in some situations even without accomplishing the complete process of decoding. Accordingly, the process of error control can be categorized into the following:

a) Forward Error Correction (FEC)

Complete process of decoding is applied on the received sequence to detect error positions in the sequence and correct the erroneous symbols. However, the process of error correction is not fool-proof and occasionally the decoder may either fail to detect presence of errors in a received sequence or, may detect errors at wrong locations, resulting in a few more erroneous symbols. This happens if, for example, too much noise gets added to the signal during transmission through a wireless channel.

b) Auto Repeat reQuest (ARQ)

In some applications (such as in data communications) it is important to receive only error-free information, even if it means more than usual delay in transmission and reception. A conceptually simple method of error detection and retransmission is useful in such situations. The error control decoder, at the receiver, only checks the presence of any error in a received sequence (this is a relatively easy task compared to full error correction). In case any error is detected, a request is sent back to the transmitter (via return channel, which must be available for this purpose) for re-transmitting the sequence (or packet) once again. The process ideally continues till an error-free sequence is received and, this may involve considerable delay in receipt and may result in delay for subsequent sequences.

Another aspect of this scheme is that the transmitter should have enough provision for storing new sequences while a packet is repeated several times. One may think of several interesting variations of the basic scheme of ARQ. Three important and popular variations are: i) Stop and Wait ARQ, ii) Continuous ARQ and iii) Selective Repeat ARQ.

c) Hybrid ARQ

After a brief recollection of the above two error control schemes, viz. FEC and ARQ, one may suggest combining the better features of both the schemes and this is, indeed, feasible and meaningful. Significant reduction in retransmission request is possible by using a moderately powerful FEC in an ARQ scheme. This saves considerable wastage in resources such as time and bandwidth and increases the throughput of the transmission system at an acceptably small packet error rate compared to any ARQ scheme with only error detection feature. This scheme is popular especially in digital satellite communication systems.

Henceforth, in this section we focus on some additional features of FEC schemes only. Application of an FEC code and a judicious choice of the code parameters are guided by several conflicting factors. Some of these factors are described in brief:

a) Nature of communication channel

Effects of many physical communication channel manifest in random and isolated errors while some channels cause bursty errors. The modulation technique employed for transmission of information, sensitivity level of a receiver (in dBm), rate of information transmission are some other issues.

b) Available channel bandwidth

As mentioned, use of an error-control scheme involves addition of controlled redundancy to original message. This redundancy in transmitted massage calls for larger bandwidth than what would be required for an encoded system. This undesirable fact is tolerable because of the obtainable gains or advantages of coded communication system over an encoded one for a specified overall system performance in terms of BER or cost.

c) Hardware complexity cost and delay

Some FEC codes of larger block length asymptotically satisfy the requirements of high rate as well as good error correcting capability but the hardware complexity, volume, cost and decoding delay of such decoders may be enormous. For a system designer, the choice of block length is somewhat limited.

d) The coding gain

FEC codes of different code rates and block sizes offer different coding gains in E_b/N_o over an uncoded system. At the first level, the coding gain is defined as:

$[(\frac{E_b}{N_0}$ in dB needed by a uncoded system to achieve a specified BER of 10^{-x}) –

$(\frac{E_b}{N_0}$ in dB needed by an FEC coded system to achieve the same BER of 10^{-x}).]

Fig. 6.33.1 shows a tree classifying some FEC codes based on their structures.

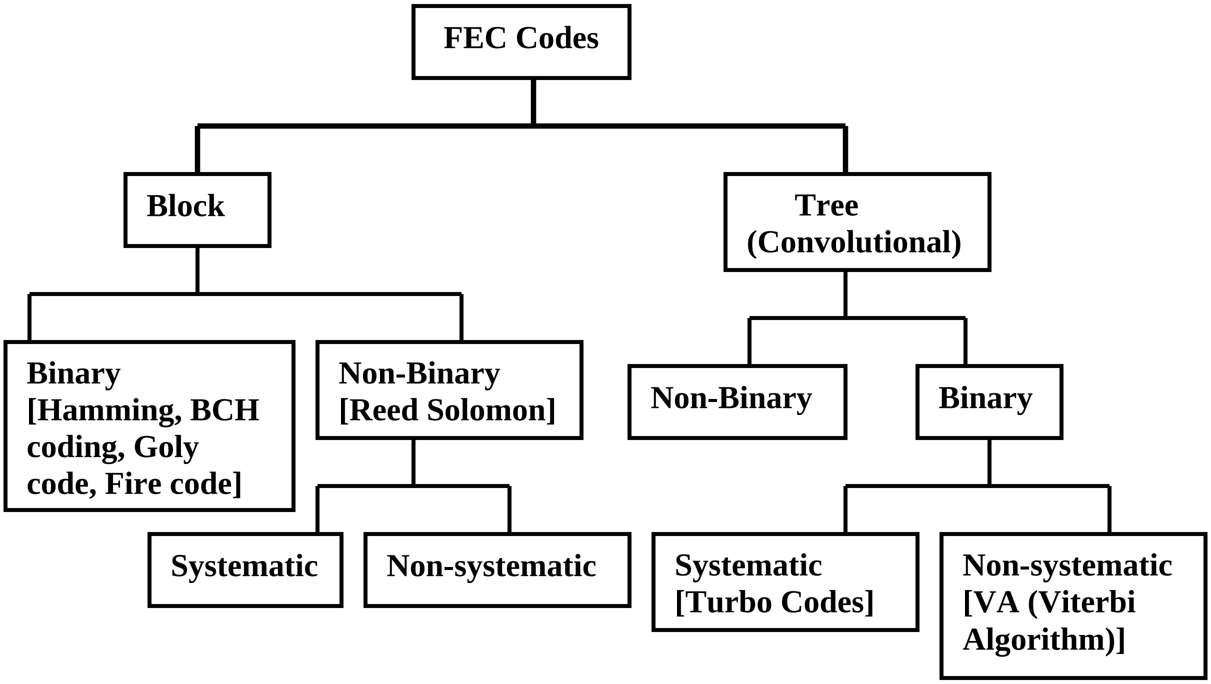
FEC Codes
Block
Tree (Convolutional)
Binary [Hamming, BCH coding, Goly code, Fire code]
Non-Binary [Reed Solomon]
Non-Binary
Binary
Systematic
Non-systematic
Systematic [Turbo Codes]
Non-systematic [VA (Viterbi Algorithm)]

Fig. 6.33.1 *Classification of FEC codes*

Block Codes

The encoder of a block code operates on a group of bits at a time. A group or ‘block’ of ‘k’ information bits (or symbols) are coded using some procedure to generate a larger block of ‘n’ bits (or symbols). Such a block code is referred as an *(n ,k)* code.

Encoding of block code

The encoder takes k information bits and computes $(n-k)$ parity bits from these bits using a specific code generator matrix. A codeword of ‘n’ bits (or symbols) is generated in the process. This operation may be ‘systematic’ or ‘nonsystematic’. In a systematic encode the first (or last) k bits in a codeword are the information bits and the rest $(n-k)$ bits are the parity check bits. For a nonsystematic code, the information symbols do not occupy such fixed positions in a codeword. In fact, they may not be identified by a simple mean.

Following matrix notation, the encoding operation can be described as a matrix multiplication:

$$\mathbf{C} = \mathbf{d}.\mathbf{G} \qquad 6.33.1$$

where, **d** : information matrix and **G** : generator matrix.

For a systematic block code, the generator matrix can be expressed as

$$G = [I_k / P],$$

where ‘I’ is a (k x k) identity matrix and ‘P’ is a (k x [n-k]) parity check matrix.

Following an equivalent polynomial notation, the encoder starts with a ‘message polynomial’ defined as below.

$$m(x) = m_0 + m_1x + m_2x_2 + ... + m_{k-1}x_{k-1} = \sum_{i=0}^{k=1} m_i x_i \qquad 6.33.2$$

Here m_i’s are the information bits (or symbols) and ‘x’ is an indeterminate representing unit delay. The exponents of ‘x’ indicate number of unit delays. For example, the above polynomial indicates that the first bit of the information sequence is ‘m_0’, the second bit is ‘m_1’ and the k-th bit is ‘m_k’.

For a binary block code, m_i’s are ‘0’ or ‘1’, i.e. they are elements of Galois Field [GF(2)]. GF(2) is a finite field consisting of two elements. The ‘+’ in the above expression indicates the ‘addition’ operation as defined in the finite field. Though addition of m_0 and m_1x does not mean anything, the ‘addition’ operator is useful in adding two or more polynomials.

For example, if,

$p(x) = p_0 + p_1x + p_2x^2 + ... + p_{k-1}x^{k-1}$ and $q(x) = q_0 + q_1x + q_2x^2 + ... + q_{k-1}x^{k-1}$

then $p(x) + q(x) = (p_0 + q_0) + (p_1 + q_1)x + (p2 + q_2)x^2 + ...$

Interestingly, over GF(2), the ‘addition’ and ‘subtraction’ operations are the same and following Boolean logic, it is equivalent to Exclusive-OR operation.

The codeword polynomial c(x) is defined as:

$$c(x) = \sum_{i=1}^{n-1} c_i x^i = c_0 + c_1x + ...c_{n-1}x^{n-1} \qquad 6.33.3$$

For a binary code $c_i \in$ GF(2). Note that the codeword polynomial is of degree ‘n-1’, implying that there are ‘n’ coefficients in the polynomial. The encoding strategy is described using a ‘generator polynomial’ g(x):

$$g(x)=\sum_{i=0}^{n-k} g_i x^i = g_0 + g_1 x + \ldots\ldots\ldots + g_{n-k} x^{n-k} \qquad 6.33.4$$

The generator polynomial g(x) is of degree (n-k), implying that it has (n-k)+1 coefficients. The coefficient of x^0 is '1' for all binary codes.

The nonsystematic encoding procedure for a block code is described as a 'multiplication' of the message polynomial and the generator polynomial:

$$c(x)= m(x).g(x) \qquad 6.33.5$$

The codeword for systematic encoding is described as:

$$c'(x)=x^{n\text{-}k}.m(x)-R_{g(x)}\left[x^{n\text{-}k}.m(x)\right] \qquad 6.33.6$$

Here, $R_{g(x)}[y(x)]$ denotes the remainder polynomial when y(x) is divided by g(x). So, the degree of the remainder polynomial is (n-k-1) or less. The polynomial $x^{n\text{-}k}$.m(x) denotes a shifted version of the message polynomial m(x), delayed by (n-k) units.

Note that irrespective of whether a code is systematic or nonsystematic, a codeword polynomial c(x) or $c'(x)$ is fully divisible by its generator polynomial. This is an important property of block codes which is exploited in the receiver during decoding operation. Another interesting point is that the generator polynomial of a binary block code happens to be a valid codeword having minimum number of '1'-s as its constant coefficients g_i-s.

An intuitive approach to decoding for block codes

With 'k' information bits in a message block, the number of possible message patterns is 2^k. Now, imagine a 'k' dimensional signal space, where each block of 'k' information bits, representing a k-tuple, is a point. The 'k' dimensional signal space is completely filled with all possible message patterns. The operation of encoding adds (n-k) redundant bits. By this, a point from 'k' dimensional filled space is mapped to a bigger n-dimensional signal space which has 2^n positions. Mapping for each point is one to one. Let us call the bigger space as the code space. So, out of 2^n points, only 2^k points will be occupied by the encoded words and the other possible points in the n-dimensional code space remain vacant. So, we can talk about a 'distance' between two codeword.

A popular measure for distance between two code words is called Hamming distance (d_H), which is the number of places in which two binary tuples differ in the code space. An encoding scheme is essentially a strategy of mapping message tuples into a code space. Let, d_{Hmin} indicate the minimum Hamming distance among any two code words in the code space. A good encoding strategy tries to maximize d_{Hmin}. Higher the d_{Hmin}, more robust or powerful is the code in terms of error detection and error correction capability.

For a block code, if ‘e_d’ denotes the number of errors it can detect in a code word and‘t’ denotes the number of errors it can correct in a received word, then

$$e_d = d_{H_{min}} - 1 \text{ and } t = \frac{d_{H_{min}} - 1}{2} \qquad 6.33.7$$

Following the matrix description approach, a parity check matrix H is used for decoding several block codes. The H matrix is related to the generator matrix G and if ‘C’ is the matrix of encoded bits,

$$C.H^T = 0 \qquad 6.33.8$$

But during transmission or due to imperfect reception, the matrix Y of received bits may be different from C:

$$Y = C + e \qquad 6.33.9$$

where ‘e’ denotes an error vector.

Now, $Y.H^T = (C+e).H^T = C.H^T + e.H^T$ 6.33.10

The matrix $S = e.H^T$ is known as a ‘syndrome matrix’. It is interesting to note that the syndrome matrix S is independent of the coded matrix C. It is dependent only on the error vector and the parity check matrix. So, the decoder attempts to recover the correct error vector from the syndrome vector. A null syndrome matrix mostly means that the received matrix Y is error-free. In general, the relationship between S and the error vector ‘e’ is not one-to-one. For a given H, several error vectors may result in the same syndrome S. So, the decoder specifically attempts to make a best selection from a set of possible error vectors than could result in a specific syndrome S. The procedure may turn out to be very involved in terms of number of computations or time etc. As a compromise, some ‘incomplete’ decoding strategies are also adopted in practice. The family of ‘Algebraic Decoding’ is practically important in this regard.

Polynomial Description: Let us define two polynomials:

r(x) = Received word polynomial = $\sum_{i=0}^{n-1} r_i x^i = r_0 + r_1 x + ... + r_{n-1} x^{n-1}$

and e(x) = Error Polynomial $= \sum_{i=0}^{n-1} e_i x^i = e_0 + e_1 x + ... + e_{n-1} x^{n-1}$, 6.33.11

where for a binary code, $r_i \in GF(2)$, $e_i \in GF(2)$ and $r(x) = c(x) \oplus e(x)$. ‘c(x)’ is the transmitted code word polynomial. The notations ‘+’ and $\oplus$ are equivalent. Both the notations are used by authors.

Note that, the job of the decoder is to determine the most probable error vector e(x) after receiving the polynomial r(x). The decoder has complete knowledge of the g(x), i.e. the encoding strategy. The decoder divides whatever r(x) it receives by g(x) and looks at the remainder polynomial:

$$R_{g(x)}[r(x)] = R_{g(x)}[c(x) + e(x)]$$

$$= R_{g(x)}\left[c(x)\right] \oplus R_{g(x)}\left[e(x)\right]$$

$$= 0 + R_{g(x)}\left[e(x)\right] \qquad 6.33.12$$

We know that c(x) is divisible by g(x). So, if the remainder is non zero, then r(x) contains one or more errors. This is the error detection part of decoding. If there is error, the decoder proceeds further to identify the positions where errors have occurred and then prepares to correct them. The remainder polynomial, which is of degree (n-k-1) or less, is called the syndrome polynomial:

$$s(x) = R_{g(x)}\left[e(x)\right] \qquad 6.33.13$$

The syndrome polynomial plays an important role in algebraic decoding algorithms and similar to the S matrix, is dependent only on the errors, though multiple error sequences may lead to the same s(x).

Source:http://nptel.ac.in/courses/Webcourse-contents/IIT%20Kharagpur/Digi%20Comm/pdf-m-6/m6l33.pdf