8.3 Concepts and General Properties

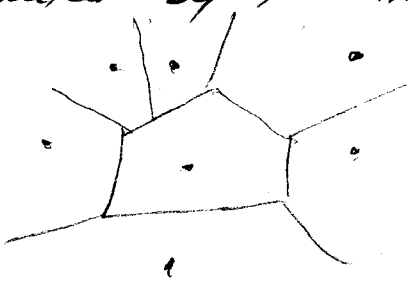
- consider (n, k) block codes. take k info bits as a sequence, release codeword of length n into channel. use $p = n-k$ check bits

- Hamming weight of a sequence is number of non zero positions. If binary, it's the number of 1's.

- Hamming distance between two codewords is number of positions in which they differ. If binary, it's the weight of $\underline{c}_1 + \underline{c}_2$.

- a code is a subset of the 2^n possible sequences (assume binary), each of them being a codeword. 2^k codewords.

- we try to pick the codewords so they are as far apart as possible in n-space. The performance is limited by the minimum distance d^*



each possible $\underline{r}$ is mapped onto closest codeword

- if we want to correct t errors, then $d^* \geq 2t+1$

  if $d^* = 2t$ there is at least one $\underline{r}$ equidistant from two codewords, so can't correct it.

- if we want to detect d errors, then $d^* \geq d+1$

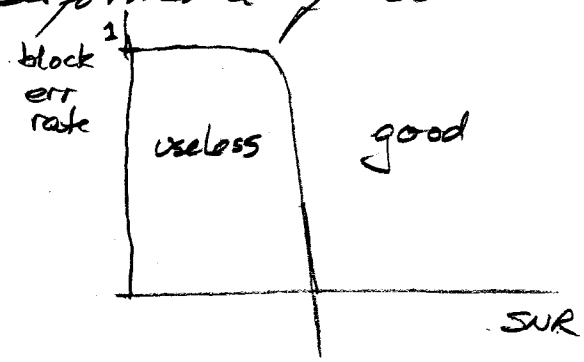
- detection easier than correction (Hellman "Error Detection Made Simple" ICC 74) and requires fewer check bits.

- lots of strange integer packing problems in code construction. if divides nicely then union of all sequences in Hamming spheres of radius t about all code words is size 2^n. No sequences left over. A <u>perfect code</u>.

- with t-error correction,

$$P_e' = \sum_{k=t+1}^{n} \binom{n}{k} P_e^k (1-P_e)^{n-k} = P_e^{t+1} \times (\text{poly in } P_e)$$

since high powers of P_e rise sharply with P_e, and because P_e itself is an exponential function of SNR, there is an approx threshold on performance of coded systems:



read Wolf "Determination of Prob of Undet Err for Lin Bin Block Codes" ICC '81

- value of length: compare two rate ½ codes: one is (2000, 1000) & corrects 100 errors; the other (200, 100) & corrects 10 errors.

Transmit 1000 bits as one word or 10 short ones. No decoding errors with shorter words only if they are distributed favourably. If 10 bit errors per word, then can handle 100 total, like the long code. But a single burst of 11 errors can cause it to fail.

- value of structure: we could search all possible patterns of length n to find good codes, but $\binom{2^n}{1^n}$ problem. Even if we found codes this way, how do we decode — examine all 2^k words to see which is closest to $\underline{r}$?

  Have to use special algebraic structure to define code families just to keep it manageable.

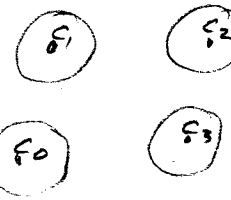
- read Wolf "State of the Art of Error Control Techniques" for a good survey of common families

- refs R Blahut "Theory and Practice of Error Control Codes" Addison Wesley 83

  Lin & Costello "Error Control Coding" Prentice Hall 1983

  Michelson & Levesque "Error Control Techniques for Digital Communication" Wiley Interscience 1985

- complete decoder puts out some codeword for every possible $\underline{r}$. incomplete decoder puts out codeword if $\underline{r}$ close enough, but indicates failure if $\underline{r}$ too far from any $\underline{c}_i$



2 decoding spheres

- hybrid FEC/ARQ. use incomplete decoder, ask for retransmission if decoding failure (too many errors)

- <u>undetected error</u> results if decoder releases a code word other than the one sent. radius of decoding spheres lets us trade off probs of:

  correct detection, decoder failure, undetected error.

- if error detection only, then undetected error only if error pattern makes received word $\underline{r}$ mimic another codeword. Simple bound:

  - if (n, k) code over rotten channel ($P_e = \frac{1}{2}$), then received $\underline{r}$ are completely random vectors. There are 2^n of them, of which only 2^k are codewords.

    So $P[\text{undet error}] = 2^k/2^n = 2^{-(n-k)} = 2^{-p}$

  - would expect true prob to be <u>much</u> lower than this on reasonable channels. It is, but for $P_e \sim 0.1 - 0.4$ and for some pathological codes or bursty channels, it can be a little higher.

- we have seen <u>block codes</u> — take block of k bits, encode into block of n bits. The other major class of codes is <u>tree codes</u>, of which the most common is <u>convolutional codes</u>

- unlike block codes, convolutional codes have no fixed length, though they are still linear.

  • generated by shift register of K stages, v output equations. In operation, shift in an info bit, read out and transmit v bits.

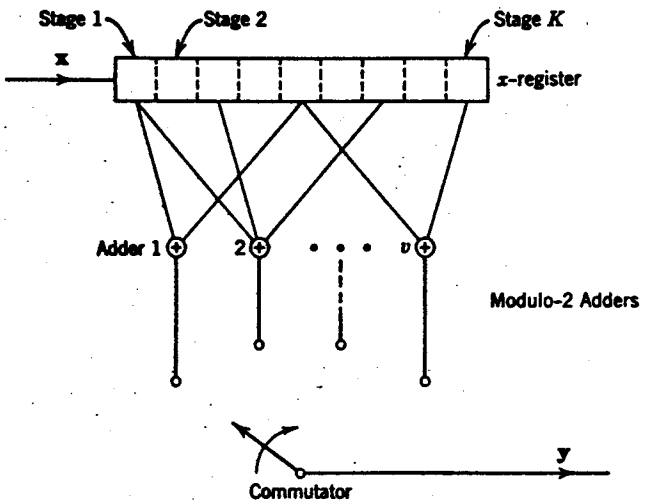


Figure 6.32 Binary convolutional encoder. The x-register is a K-stage shift register.

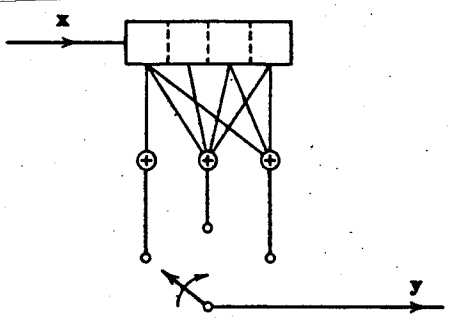


Figure 6.33 A particular $K = 4$, $v = 3$ convolutional encoder.

K = constraint length
1/v = rate

more generally, clock in b bits, read out v bits, rate b/v.

- no limit on transmitted info length
- decoding is continuous, like sliding window.

Coding Theorem

We have seen that block error rate P_B has a threshold in BER P_e and SNR, and that long codes are better that short ones. What is the behaviour in the limit?

define rate $R = k/n$

capacity $C = 1 + P_e \log_2 P_e + (1-P_e)\log_2(1-P_e)$

then
$$\lim_{n\to\infty} P_B = \begin{cases} 0 & R<C \\ 1 & R>C \end{cases}$$

(proof Lathi sections 8.3, 8.4)

Doesn't give a design procedure, just says that as $n\to\infty$, all codes with $R>C$ are bad, and that some codes (even randomly selected ones) with $R<C$ are good.

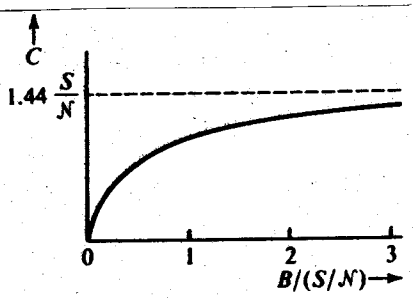
For continuous channels: sequences of pulses with Gaussian amplitudes having mean square value S, additive Gaussian noise variance N, pulse rate $2B$ [Lathi 8.5]:

$$C_p = \tfrac{1}{2}\log_2(1+S/N) \quad \text{bits/pulse}$$

$$C = B\log_2(1+S/N) \quad \text{bits/sec}$$

$$= B\log_2(1+2E_p/N)$$

$$= B\log_2\left(1+\frac{P}{NB}\right) \quad \text{bps}$$



Monotonic increase with B, limit $1.44\,P/N$ bps
Not much change for $BN/P > 1$ ($E_p/N < \frac{1}{2}$)