An introduction to error correcting codes

Martin Leslie

Department of Mathematics
University of Arizona

April 10, 2009

First example: ISBNs

- A 10 digit ISBN (International Standard Book Number) is of the form

$$x_1 x_2 \cdots x_9 x_{10}$$

where

$$x_{10} \equiv 1x_1 + 2x_2 + 3x_3 + \ldots + 9x_9 \pmod{11}.$$

- This 'parity digit', x_{10}, is used to detect input errors. In particular, if a valid ISBN has exactly one digit altered or two adjacent digits transposed it will no longer be a valid ISBN.

Some algebra: a field of order 2

- A field is a 'set of numbers' that has a 0, a 1 and is such that you can add, subtract, multiply and divide.
- For example: $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$. These are all infinite fields.
- But there are finite fields as well. For example $\mathbb{F}_2 = \{0, 1\}$ with addition and multiplication modulo 2.

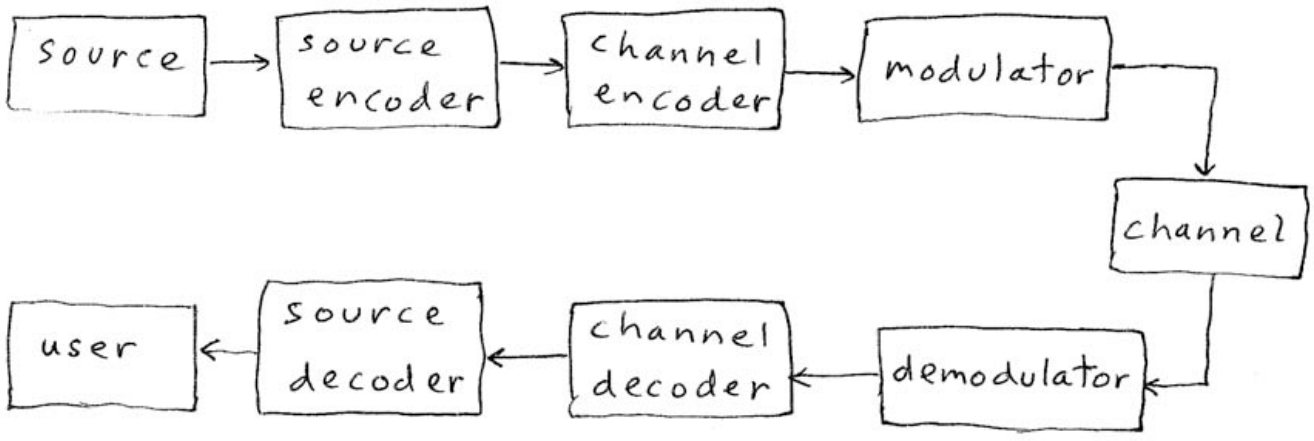
$+$	0	1
0	0	1
1	1	0

$\times$	0	1
0	0	0
1	0	1

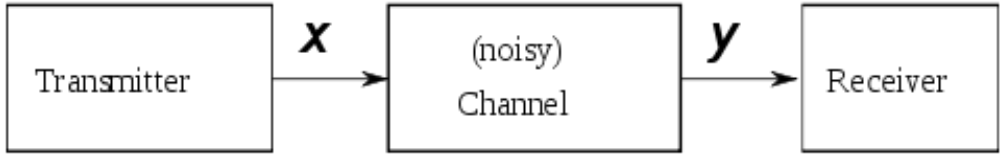
- We can work with polynomials and also do linear algebra over any field.

Communication

- Communication systems look something like

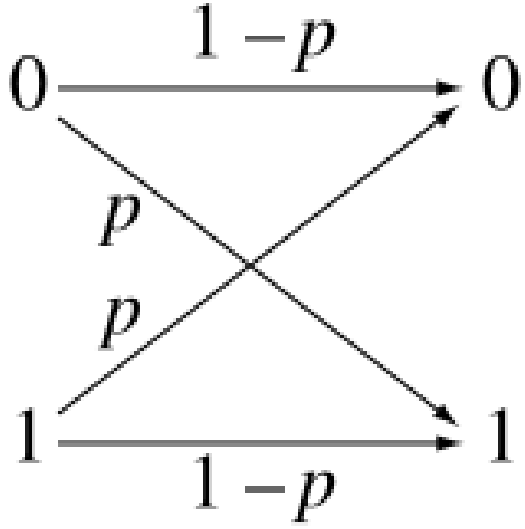


- Today we're just talking about channel coding so our picture is



A noisy channel

- The channel model we will use is the binary symmetric channel (BSC) which takes a binary input and with probability $p < 1/2$ switches it.



- This is a good model for deep space communications but not so good for hard drives or for terrestrial communications where errors often come in bursts.

Repetition codes

- We're sending binary messages over a channel which introduces some errors. How can we try to make sure we can correct these errors?
- The first idea is to use a repetition code: repeat each bit n times for some odd n. Then decode the message using majority rule.
- For example if $n = 3$ we would encode the message 10011 as $111\ 000\ 000\ 111\ 111$. If an error, say in the 5th position, is added by the channel we will receive $111\ 010\ 000\ 111\ 111$ and successfully decode the message. If we're unlucky and have two errors in the same three bit codeword then we will decode the message incorrectly.

How good are repetition codes?

- The *information rate* of this repetition code is $R = 1/n$ information bits/code bit.
- The chance of incorrectly decoding a given block is the chance that there is an error in more than half of the n bits sent over the channel. This is

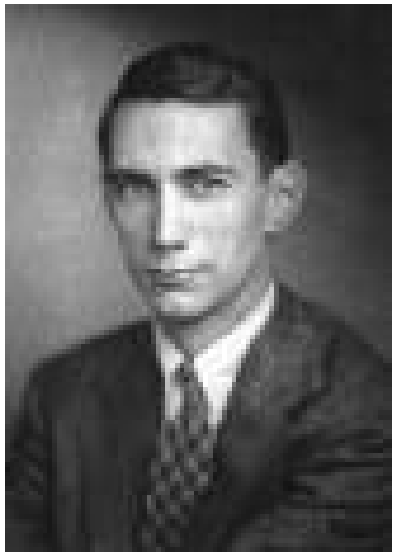
$$p_{cw} = \sum_{n/2 < i \leq n} \binom{n}{i} p^i (1-p)^{n-i}.$$

- As $n \to \infty$ this probability goes to zero. But also the information rate goes to zero. Can we do better?

Noisy–channel coding theorem

Theorem (Shannon, 1948)

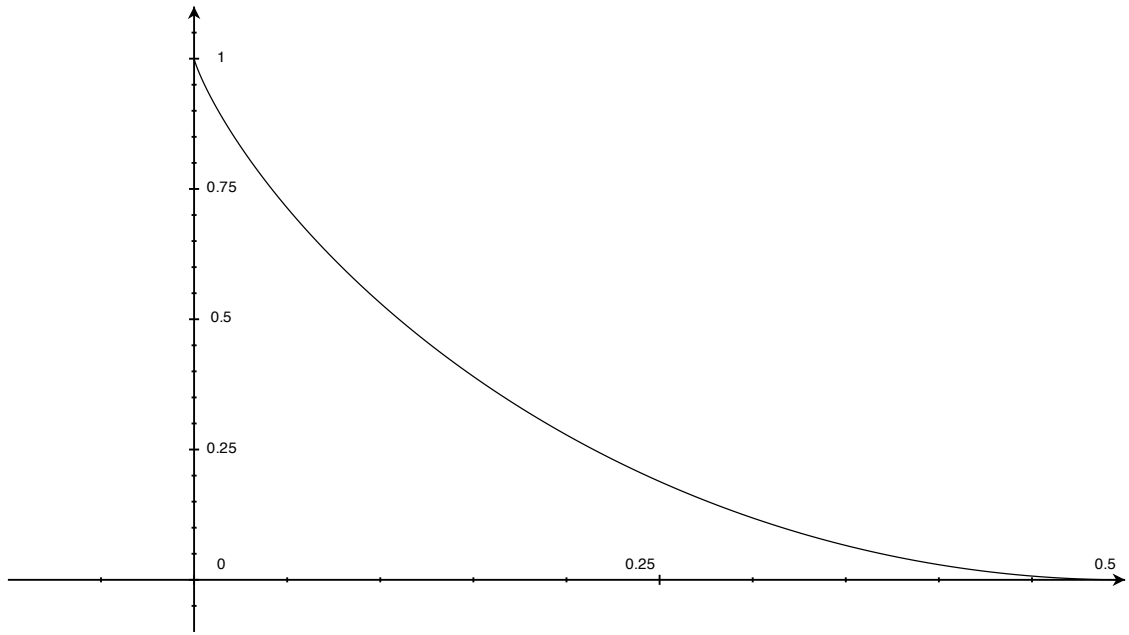
A given channel has a capacity C. If $R < C$ then for all $\epsilon > 0$ there exists a code with information rate R and probability of decoding a block incorrectly less than ϵ. This is not true for $R > C$.



BSC capacity

- The capacity of the binary symmetric channel with crossover probability p is

$$C = 1 + p\log_2(p) + (1-p)\log_2(1-p).$$



Linear codes

- An $[n, k]_2$ code C is a k-dimensional linear subspace of $\mathbb{F}_2^n$.
- The basis vectors of C are the rows of the *generating matrix* G.
- With this matrix we can carry out encoding by the function from $\mathbb{F}_2^k \to \mathbb{F}_2^n$ that sends $u \mapsto uG$.
- Then the information rate is $R = k/n$.

The Hamming [7,4] code

- Let $G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$
- Then the codewords of C are

0000 000	1000 110	0010 111
1111 111	0100 011	1001 011
	1010 001	1100 101
	1101 000	1110 010
	0110 100	0111 001
	0011 010	1011 100
	0001 101	0101 110

The Parity Check Matrix

- If a code C has generating matrix $G = (I \ P)$ then define its *parity check matrix* to be $H^T = \begin{pmatrix} P \\ I \end{pmatrix}$.
- Notice that $GH^T = P + P = 0$.
- So if $c \in C$ we know $c = uG$ and thus

$$cH^T = uGH^T = 0.$$

- For the Hamming [7,4] code we have $H^T = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

Hamming distance

- The Hamming distance $d_H(u, v)$ for $u, v \in \mathbb{F}_2^n$ is the number of places in which u and v differ.
- This satisfies all the axioms of a metric on $\mathbb{F}_2^n$.
- The Hamming weight is the number of 1's in a vector,

$$\mathrm{wt}(u) = d_H(u, 0).$$

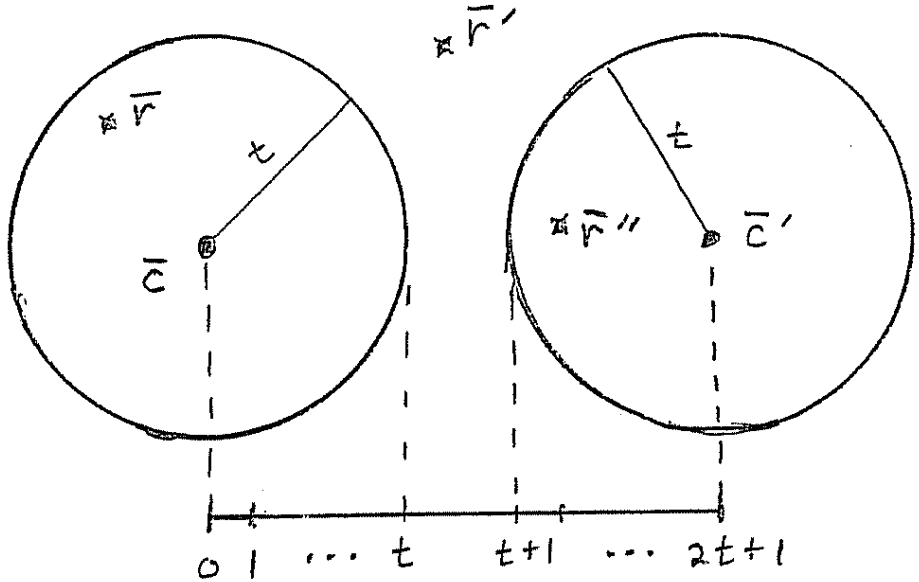
- Define d to be the minimum Hamming distance between any two vectors in C. Then

$$d = \min_{u \neq v \in C} d_H(u, v) = \min_{u \in C \setminus \{0\}} \mathrm{wt}(u).$$

- Then we talk about an $[n, k, d]_2$ code.

Nearest neighbour decoding

- If we receive $u \in \mathbb{F}_2^n$ we decode it to an element v of C for which $d_H(u, v)$ is minimum.
- It's possible that this is incorrect decoding but it is certainly the best choice on average.
- If d is the minimum distance of C then we can correct at least $t = \left\lfloor \frac{d-1}{2} \right\rfloor$ errors.



Syndrome decoding

- If we send codeword c but the channel adds error e, we receive $r = c + e$ and then can find the *syndrome* $rH^T = (c + e)H^T = eH^T$.
- We can find syndromes for all the possible errors (elements of $\mathbb{F}_2^n$) added by the channel and for each syndrome find a most likely error that leads to it - one with minimum weight.
- This gives us a *syndrome table* which allows us to decode more easily. For example for the Hamming [7,4] code we have

syndrome	000	110	011	111	101	100	010	001
likely error	0	e_1	e_2	e_3	e_4	e_5	e_6	e_7

The Hamming bound

- We want $[n, k, d]$ codes with large $R = k/n$ and large d. These two requirements pull in opposite directions.
- In particular, by thinking about packing spheres of radius d inside $\mathbb{F}_2^n$ we get the Hamming bound

$$2^k \left(\sum_{i=0}^{t} \binom{n}{i} \right) \leq 2^n.$$

- If the Hamming bound is satisfied with equality we say a code is *perfect*. This means that there is never any ambiguity about where to decode a received message to: every element of $\mathbb{F}_2^n$ is inside exactly one radius d sphere centred at a codeword.

Some (families of) codes

- The repetition code for odd n is a $[n, 1, n]$ code.
- The parity check code for even n is an $[n, n-1, 1]$ code
- Hamming codes are $[2^m - 1, 2^m - m - 1, 3]$ codes
- Golay codes
- BCH codes, Reed-Muller codes, Reed-Solomon codes
- Algebraic geometry codes
- Convolutional codes
- Turbo codes
- Low Density Parity Check (LDPC) codes

More finite fields

- It is possible to construct finite fields of order 2^m for each positive integer m. We construct $\mathbb{F}_4 = \mathbb{F}_{2^2}$.
- To do this we take α to be a root of $x^2 + x + 1$ so $\alpha^2 = \alpha + 1$.
- Then $\mathbb{F}_4 = \mathbb{F}_2[\alpha] = \{0, 1, \alpha, \alpha + 1\} = \{0, 1, \alpha, \alpha^2\}$ with operations

$+$	0	1	α	$\alpha+1$
0	0	1	α	$\alpha+1$
1	1	0	$\alpha+1$	α
α	α	$\alpha+1$	0	1
$\alpha+1$	$\alpha+1$	α	1	0

$\times$	1	α	$\alpha+1$
1	1	α	$\alpha+1$
α	α	$\alpha+1$	1
$\alpha+1$	$\alpha+1$	1	α

A Reed-Solomon code

- Take $\mathbb{F}_{2^8} = \mathbb{F}_{256} = \{0, 1, \alpha, \ldots, \alpha^{254}\}$ for our finite field. Each element of this field can be described with 8 bits.
- Consider our messages to be polynomials over $\mathbb{F}_{256}$ of degree ≤ 222. Then encode such an f to a codeword

$$\left(f(1), f(\alpha), \ldots, f(\alpha^{254})\right).$$

- The resulting code is a $[255, 223]_{256}$ linear code.
- The way to think about this code is that the redundancy is coming from giving 255 different values of a degree 222 polynomial when in fact 223 values determine it.

How many errors can it correct?

- If d is minimum weight then some f has $255 - d$ zeros so $255 - d \leq \deg(f) \leq 222$ and thus $d \geq 33$.
- So we can correct at least $(33 - 1)/2 = 16$ errors.
- These errors are errors in $\mathbb{F}_{256}$ so if there are multiple errors within a byte it only counts as one error. This makes Reed-Solomon codes good at correcting burst errors.

CDs

- CDs use a "cross interleaved", "shortened" version of this Reed-Solomon code that can correct error bursts of up to 3500 bits (2.4mm radial distance).

The End

- Any questions?