ECC, Probability and LDPCC

CoCoNut, 2016
Emmanuela Orsini

Previously

$[8, 3, 5]$ Linear Code over $GF(3)$

```
  [1 0 0 1 1 0 2 2]
G=[0 1 0 2 1 2 2 0]
  [0 0 1 0 2 1 2 2]
```

```
  [1 0 0 0 0 1 2 2]
  [0 1 0 0 0 0 1 2]
H=[0 0 1 0 0 2 1 2]
  [0 0 0 1 0 1 1 0]
  [0 0 0 0 1 2 2 2]
```

```
(0 0 0 0 0 2 0 0)
(2 0 1 2 1)
(0 0 0 0 0 0 2 0)
(1 2 2 2 1)
(0 0 0 0 0 0 0 2)
(1 1 1 0 1)
(0 0 0 0 1 0 0 0)
(0 0 0 0 1)
(0 0 0 0 0 0 1 0)
(2 1 1 1 2)
(0 0 0 0 0 1 0 0)
(1 0 2 1 2)
(0 0 0 0 0 0 0 1)
(2 2 2 0 2)
(0 0 0 0 2 0 0 0)
(0 0 0 0 2)
(0 0 0 1 0 0 0 0)
(0 0 0 1 0)
(0 0 0 2 0 0 0 0)
(0 0 0 2 0)
(0 0 1 0 0 0 0 0)
(0 0 1 0 0)
(0 0 2 0 0 0 0 0)
(0 0 2 0 0)
(0 1 0 0 0 0 0 0)
(0 1 0 0 0)
(0 2 0 0 0 0 0 0)
(0 2 0 0 0)
(1 0 0 0 0 0 0 0)
(1 0 0 0 0)
(2 0 0 0 0 0 0 0)
(2 0 0 0 0)
(0 0 0 1 0 0 0 2)
(1 1 1 1 1)
```

```
(0 0 0 0 2 0 1 0)
(2 1 1 1 1)
(0 0 0 0 0 1 0 1)
(0 2 1 1 1)
(0 0 0 2 0 2 0 0)
(2 0 1 1 1)
(0 0 0 2 0 0 2 0)
(1 2 2 1 1)
(0 0 0 0 2 1 0 0)
(1 0 2 1 1)
(0 0 0 0 0 0 1 1)
(1 0 0 1 1)
(0 0 0 1 1 0 0 0)
(0 0 0 1 1)
(0 0 0 2 0 0 0 2)
(1 1 1 2 1)
(0 1 0 0 0 2 0 0)
(2 1 1 2 1)
(0 0 2 0 0 0 2 0)
(1 2 1 2 1)
(0 2 0 0 0 2 0 0)
(2 2 1 2 1)
(2 0 0 0 0 2 0 0)
(1 0 1 2 1)
(1 0 0 0 0 2 0 0)
(0 0 1 2 1)
(0 2 0 0 0 0 2 0)
(1 1 2 2 1)
(1 0 0 0 0 0 2 0)
(2 2 2 2 1)
(2 0 0 0 0 0 2 0)
(0 2 2 2 1)
(0 1 0 0 0 0 2 0)
(1 0 2 2 1)
```

```
(0 0 1 0 0 2 0 0)
(2 0 2 2 1)
(0 0 0 0 0 1 1 0)
(0 1 0 2 1)
(0 0 1 0 0 0 2 0)
(1 2 0 2 1)
(0 0 2 0 0 2 0 0)
(2 0 0 2 1)
(0 0 0 2 1 0 0 0)
(0 0 0 2 1)
(1 0 0 0 0 0 0 2)
(2 1 1 0 1)
(2 0 0 0 0 0 0 2)
(0 1 1 0 1)
(0 1 0 0 0 0 0 2)
(1 2 1 0 1)
(0 2 0 0 0 0 0 2)
(1 0 1 0 1)
(0 0 0 1 0 2 0 0)
(2 0 1 0 1)
(0 0 1 0 1 0 0 0)
(0 0 1 0 1)
(0 0 1 0 0 0 0 2)
(1 1 2 0 1)
(0 0 0 1 0 0 2 0)
(1 2 2 0 1)
(0 0 0 0 2 0 0 1)
(2 2 2 0 1)
(0 0 2 0 1 0 0 0)
(0 0 2 0 1)
(0 0 2 0 0 0 0 2)
(1 1 0 0 1)
(0 1 0 0 1 0 0 0)
(0 1 0 0 1)
```

```
(0 2 0 0 1 0 0 0)
(0 2 0 0 1)
(1 0 0 0 1 0 0 0)
(1 0 0 0 1)
(2 0 0 0 1 0 0 0)
(2 0 0 0 1)
(2 0 0 0 0 0 1 0)
(1 1 1 1 2)
(1 0 0 0 0 0 1 0)
(0 1 1 1 2)
(0 1 0 0 0 0 1 0)
(2 2 1 1 2)
(0 0 2 0 0 1 0 0)
(1 0 1 1 2)
(0 2 0 0 0 0 1 0)
(2 0 1 1 2)
(0 1 0 0 0 1 0 0)
(1 1 2 1 2)
(0 0 1 0 0 0 1 0)
(2 1 2 1 2)
(0 2 0 0 0 1 0 0)
(1 2 2 1 2)
(0 0 0 1 0 0 0 1)
(2 2 2 1 2)
(1 0 0 0 0 1 0 0)
(2 0 2 1 2)
(2 0 0 0 0 1 0 0)
(0 0 2 1 2)
(0 0 2 0 0 0 1 0)
(2 1 0 1 2)
(0 0 0 0 0 2 2 0)
(0 2 0 1 2)
(0 0 1 0 0 1 0 0)
(1 0 0 1 2)
```

```
(0 0 0 1 2 0 0 0)
(0 0 0 1 2)
(0 0 0 1 0 0 1 0)
(2 1 1 2 2)
(0 0 0 0 1 2 0 0)
(2 0 1 2 2)
(0 0 0 0 0 2 0 2)
(0 1 2 2 2)
(0 0 0 0 1 0 2 0)
(1 2 2 2 2)
(0 0 0 2 0 0 0 1)
(2 2 2 2 2)
(0 0 0 1 0 1 0 0)
(1 0 2 2 2)
(0 0 0 0 0 0 2 2)
(2 0 0 2 2)
(0 0 0 2 2 0 0 0)
(0 0 0 2 2)
(0 0 0 0 1 0 0 2)
(1 1 1 0 2)
(0 0 0 2 0 0 1 0)
(2 1 1 0 2)
(0 0 2 0 0 0 0 1)
(2 2 1 0 2)
(0 0 1 0 2 0 0 0)
(0 0 1 0 2)
(0 2 0 0 0 0 0 1)
(2 1 2 0 2)
(2 0 0 0 0 0 0 1)
(1 2 2 0 2)
(1 0 0 0 0 0 0 1)
(0 2 2 0 2)
(0 0 0 2 0 1 0 0)
(1 0 2 0 2)
```

$[8,3,5]$ Linear Code over $GF(3)$

```
   [1 0 0 1 1 0 2 2]
G= [0 1 0 2 1 2 2 0]
   [0 0 1 0 2 1 2 2]
```

```
   [1 0 0 0 0 1 2 2]
   [0 1 0 0 0 0 1 2]
H= [0 0 1 0 0 2 1 2]
   [0 0 0 1 0 1 1 0]
   [0 0 0 0 1 2 2 2]
```

```
r:=Vector(GF(3),[1,1,0,1,0,2,0,2]);
r*Transpose(H);
(1 2 2 0 2)
```

Consider the usual $[7,4]_2$ Hamming code with parity-check matrix H, and suppose $\mathbf{r} = 1011000$ is the received word. Then the associated syndrome is

$$\mathbf{s} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}.$$

The decoding procedure for binary Hamming code is as follows: Suppose a single error occurred in the ith component, then $\mathbf{s} = H\mathbf{e}_i = \mathbf{h}_i$, where $\mathbf{h}_i$ is the ith column of H.

- Compute the syndrome $\mathbf{s} = H\mathbf{r}$;
- Find the column $\mathbf{h}_i$ of H that matches the syndrome;
- Complement the ith bit of the received word.

The simplest way to add redundancy to a transmission is to repeat each symbol of the message a fixed number of times.

- $\mathcal{A} = \{a_1, \ldots, a_q\}$
- A codeword is given by a symbol of $\mathcal{A}$ repeated n times

$$C_q^{rep}(n) = \{\underbrace{a_1 \ldots a_1}_{n}, \underbrace{a_2 \ldots a_2}_{n}, \ldots, \underbrace{a_q \ldots a_q}_{n}\},$$

- Minimum distance? Length? Dimension? Rate?
- Decoding with a majority vote strategy (it can be proved equivalent to the MLD)

This lecture

A communication scheme

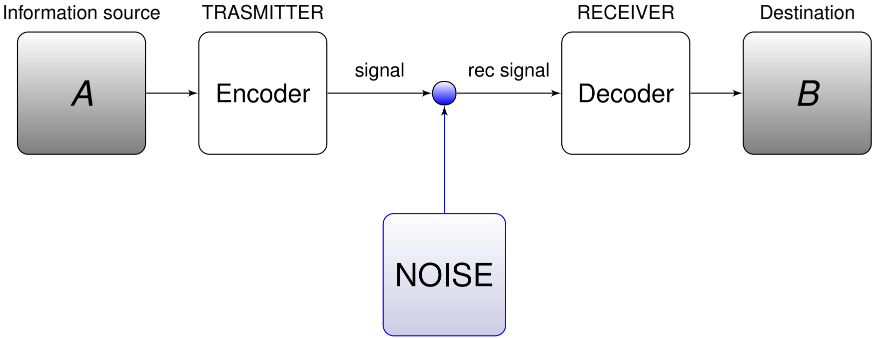


Figure: Communication channel model with noise

A communication scheme

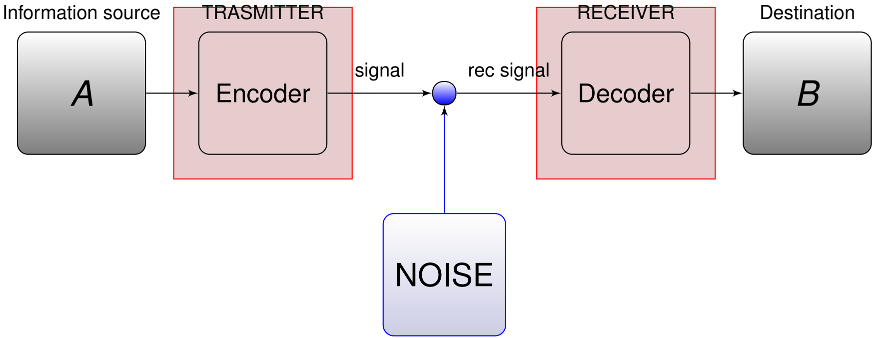
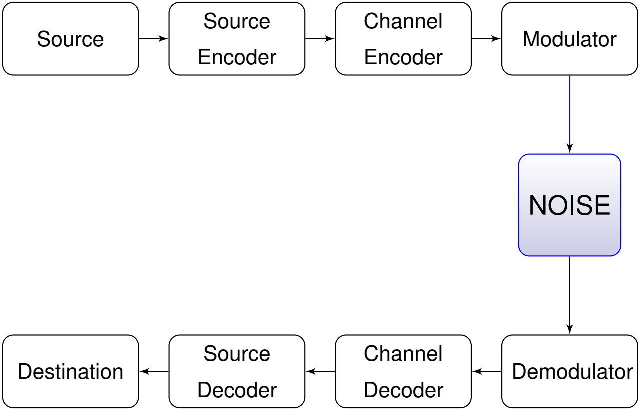
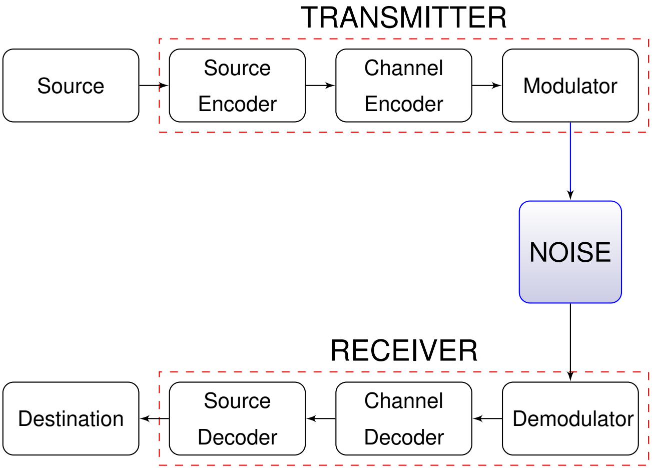


Figure: Communication channel model with noise

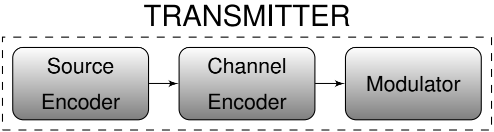
A more detailed communication scheme



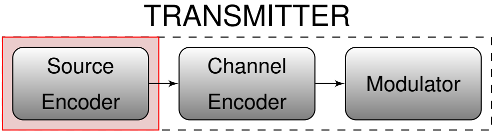
A more detailed communication scheme



A more detailed communication scheme

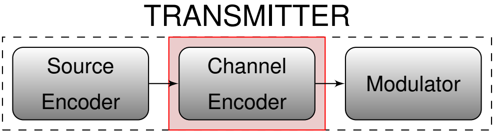


A more detailed communication scheme



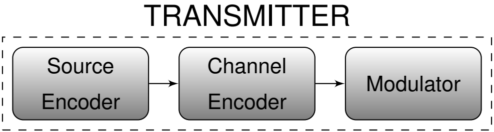
- *Source encoder*: it represents the source message in a compact way by removing unnecessary content (data compression)

A more detailed communication scheme



- *Source encoder*: it represents the source message in a compact way by removing unnecessary content (data compression)
- *Channel encoder*: it outputs a "channel codeword", introducing a systematic redundancy to tolerate errors that might be introduced by the channel (*error correcting encoding*).

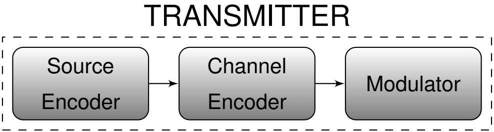
A more detailed communication scheme



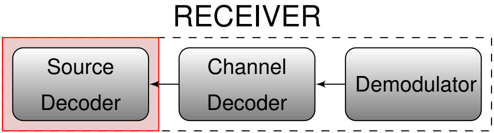
- *Source encoder*: it represents the source message in a compact way by removing unnecessary content (data compression)
- *Channel encoder*: it outputs a “channel codeword”, introducing a systematic redundancy to tolerate errors that might be introduced by the channel (*error correcting encoding*).

- *Channel decoder*: reverse operation of channel encoder

A more detailed communication scheme



- *Source encoder*: it represents the source message in a compact way by removing unnecessary content (data compression)
- *Channel encoder*: it outputs a “channel codeword”, introducing a systematic redundancy to tolerate errors that might be introduced by the channel (*error correcting encoding*).



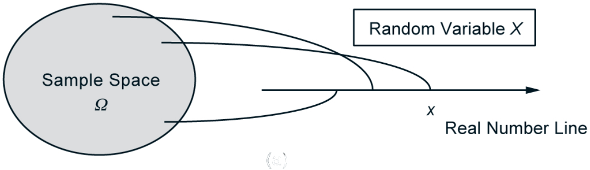
- *Channel decoder*: reverse operation of channel encoder
- *Source decoder* : reverse operation of source encoder

- Procedures for data compression
- How much can we compress? Is there a limit?
- How much redundancy is necessary to minimize the error probability in decoding?
- Is it true that to minimize the error probability we need to add a lot of redundancy, or we can achieve the same result more efficiently?

A random variable is a real-valued function of the experimental outcome:

$$X : \Omega \to E \subseteq \mathbb{R}.$$

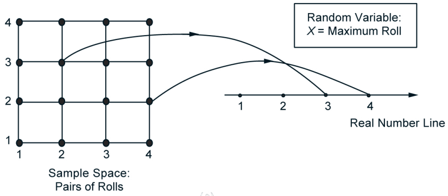
Visualization of a random variable: it is a function that assigns a numerical value to each possible outcome of the experiment



Discrete random variable: if the set of possible values is finite or at most countably finite.

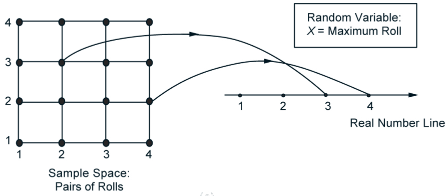
Random variables II

Example Consider two rolls of a 4-sided dice. Let X be the maximum of the two rolls. If the outcome is $(4, 2)$ then the value of X is 4, if the outcome is $(2, 3)$, the value of X is 3.



Random variables II

Example Consider two rolls of a 4-sided dice. Let X be the maximum of the two rolls. If the outcome is $(4,2)$ then the value of X is 4, if the outcome is $(2,3)$, the value of X is 3.



1. Let X be a discrete random variable, the *probability mass function* (probability for short) of X, denoted by p_X is the probability of the event $\{X = x\}$ consisting of all outcomes that give rise to a value of X equal to x:

$$p_X(x) = \Pr(X = x).$$

In many experiments, more than a single random variable is involved. For example, to calculate probabilities involving two random variables X and Y we need the joint probability of X and Y.

Definition

Let X and Y be two discrete random variables associated with the same experiment. The **joint probability** of X and Y is defined as

$$p_{X,Y}(x,y) = \Pr(X = x, Y = y),$$

for all pairs of numerical values (x, y) that X and Y can take.

- Let E be the set consisting of all (x, y) values, then

$$\Pr((X, Y) \in E) = \sum_{(x,y)\in E} p_{X,Y}(x,y).$$

- The marginal probability mass function of X and Y, denoted $p_X(x)$ and $p_Y(y)$ are given by

$$p_X(x) = \sum_y p_{X,Y}(x,y) \qquad p_Y(y) = \sum_x p_{X,Y}(x,y)$$

- The marginal probability mass function of X and Y, denoted $p_X(x)$ and $p_Y(y)$ are given by

$$p_X(x) = \sum_y p_{X,Y}(x,y) \qquad p_Y(y) = \sum_x p_{X,Y}(x,y)$$

Example. The input source to a noisy channel is a r. v. X over a, b, c, d. The output for this channel is a r. v. Y over the same alphabet. The joint probability of these two random variables is:

X \ Y	a	b	c	d	$\Pr_X(x)$
a	$1/8$	$1/16$	$1/16$	$1/4$	$1/2$
b	$1/16$	$1/8$	$1/16$	0	$1/4$
c	$1/32$	$1/32$	$1/16$	0	$1/8$
d	$1/32$	$1/32$	$1/16$	0	$1/8$
$\Pr_Y(y)$	$1/4$	$1/4$	$1/4$	$1/4$	1

- The marginal probability mass function of X and Y, denoted $p_X(x)$ and $p_Y(y)$ are given by

$$p_X(x) = \sum_y p_{X,Y}(x,y) \qquad p_Y(y) = \sum_x p_{X,Y}(x,y)$$

Example. The input source to a noisy channel is a r. v. X over a, b, c, d. The output for this channel is a r. v. Y over the same alphabet. The joint probability of these two random variables is:

X \ Y	a	b	c	d	$\Pr_X(x)$
a	$1/8$	$1/16$	$1/16$	$1/4$	$1/2$
b	$1/16$	$1/8$	$1/16$	0	$1/4$
c	$1/32$	$1/32$	$1/16$	0	$1/8$
d	$1/32$	$1/32$	$1/16$	0	$1/8$
$\Pr_Y(y)$	$1/4$	$1/4$	$1/4$	$1/4$	1

X \ Y	y_1	$\dots$	y_l	$\Pr_X(x)$
x_1	$p_{X,Y}(x_1, y_1)$	$\dots$	$p_{X,Y}(x_1, y_l)$	$p_X(x_1)$
x_2	$p_{X,Y}(x_2, y_1)$	$\dots$	$p_{X,Y}(x_2, y_l)$	$p_X(x_2)$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_h	$p_{X,Y}(x_h, y_1)$	$\dots$	$p_{X,Y}(x_h, y_l)$	$p_X(x_h)$
$\Pr_y(y)$	$p_Y(y_1)$	$\dots$	$p_Y(y_n)$	1

- The marginal probability mass function of X and Y, denoted $p_X(x)$ and $p_Y(y)$ are given by

$$p_X(x) = \sum_y p_{X,Y}(x,y) \qquad p_Y(y) = \sum_x p_{X,Y}(x,y)$$

Example. The input source to a noisy channel is a r. v. X over a, b, c, d. The output for this channel is a r. v. Y over the same alphabet. The joint probability of these two random variables is:

X \ Y	a	b	c	d	$\Pr_X(x)$
a	$1/8$	$1/16$	$1/16$	$1/4$	$1/2$
b	$1/16$	$1/8$	$1/16$	0	$1/4$
c	$1/32$	$1/32$	$1/16$	0	$1/8$
d	$1/32$	$1/32$	$1/16$	0	$1/8$
$\Pr_Y(y)$	$1/4$	$1/4$	$1/4$	$1/4$	1

X \ Y	y_1	$\dots$	y_l	$\Pr_X(x)$
x_1	$p_{X,Y}(x_1, y_1)$	$\dots$	$p_{X,Y}(x_1, y_l)$	$p_X(x_1)$
x_2	$p_{X,Y}(x_2, y_1)$	$\dots$	$p_{X,Y}(x_2, y_l)$	$p_X(x_2)$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$
x_h	$p_{X,Y}(x_h, y_1)$	$\dots$	$p_{X,Y}(x_h, y_l)$	$p_X(x_h)$
$\Pr_y(y)$	$p_Y(y_1)$	$\dots$	$p_Y(y_n)$	1

$$\begin{aligned}\Pr_Y(Y=a) = &\Pr_{X,Y}(X=a, Y=a)+\\ &\Pr_{X,Y}(X=b, Y=a)+\\ &\Pr_{X,Y}(X=c, Y=a)+\\ &\Pr_{X,Y}(X=d, Y=a) = 1/4\end{aligned}$$

- Conditional probability:

$$p_{X|Y}(x,y) = \Pr(X = x|Y = y).$$

$$p_{X|Y}(x,y) = \frac{\Pr(X = x, Y = y)}{\Pr(Y = y)} = \frac{p_{X,Y}(x,y)}{p_Y(y)}.$$

- Multiplication rule:

$$p_{X,Y}(x,y) = p_{X|Y}(x,y) \cdot p_Y(y) \quad p_X(x) > 0$$

and

$$p_{X,Y}(x,y) = p_{Y|X}(x,y) \cdot p_X(x) \quad \text{if} \quad p_Y > 0.$$

- Two random variables X and Y are independent if

$$p_{X|Y}(x,y) = p_X(x).$$

In this case we also have that

$$p_{Y|X}(x,y) = p_Y(y).$$

Problem 1 We choose one box at random and extract from it a ball. We want to compute the probability that the extracted ball is green.

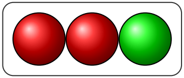
Solution:

- X the random value that takes values in $\{1, 2\}$
- Y the random value that takes values $\{g, r\}$

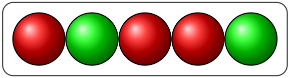
We need to compute

$$\Pr(Y = g) = \Pr(X = 1) \cdot \Pr(Y = g | X = 1) + \Pr(X = 2) \cdot \Pr(Y = g | X = 2)$$

$\mathcal{B}_1$



$\mathcal{B}_2$



- **Sum rule**: We can rewrite the marginal probabilities: Given X and Y two random variables, and E_X, E_Y the set of all values x and y respectively

$$Pr_X(x) = \sum_{y \in E_y} \Pr(X = x, Y = y) =$$

$$\sum_{y \in E_y} \Pr(Y = y) \cdot \Pr(X = x | Y = y)$$

- **Bayes rule**: From the multiplication rule

$$p_{X,Y}(x, y) = p_{X|Y}(x, y) \cdot p_Y(y) \quad p_{X,Y}(x, y) = p_{Y|X}(x, y) \cdot p_X(x)$$

we obtain

$$p_{X|Y}(x, y) = \frac{p_{Y|X}(x, y) \cdot p_X(x)}{p_Y(y)}$$

Problem 2 We now want to compute the probability that $\mathcal{B}_1$ is the chosen box, knowing that the extracted ball is green.

Solution:

- X takes values $\{0, 1\}$
- Y takes values $\{g, r\}$

We need to compute $\Pr(X = 1 | Y = g)$. Using the multiplication rule,

$$\Pr(X = 1 | Y = g) \cdot \Pr(Y = g) = \Pr(Y = g | X = 1) \cdot \Pr(X = 1)$$

and from this

$$\Pr(X = 1 | Y = g) = \frac{\Pr(Y = g | X = 1) \cdot \Pr(X = 1)}{\Pr(Y = g)} = 5/11$$

Suppose a BSC with crossover probability p ($p < 1/2$).

$$\mathbf{c} \in \mathbb{F}_2^n \text{ is sent and } \mathbf{r} \in \mathbb{F}_2^n \text{ is received}$$

The probabilities

$$\Pr(c|r) \quad \Pr(c) \quad \Pr(r|c) \quad \Pr(r)$$

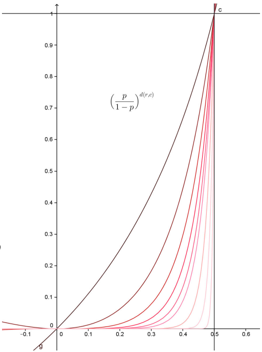
are related by Bayes'rule

$$\Pr(c|r) = \frac{\Pr(r|c)\Pr(c)}{\Pr(r)}$$

- $\hat{\mathbf{c}} = max_{\mathbf{c} \in C} \Pr(\mathbf{c}|\mathbf{r})$ Maximum a posteriori probability
- $\hat{\mathbf{c}} = max_{\mathbf{c} \in C} \Pr(\mathbf{r}|\mathbf{c})$ Maximum likelihood decoder

- $\mathbf{r} = r_1 \ldots r_n$ and $\mathbf{c} = c_1 \ldots c_n$
- $\Pr(\mathbf{r}|\mathbf{c}) = \prod_{i=1}^{n} \Pr(r_i|c_i)$ since we assumed that bit errors are independent.
- $$\begin{cases} \Pr(r_i|c_i) = p & \text{if } r_i \neq c_i \\ \Pr(r_i|c_i) = 1-p & \text{if } r_i = c_i \end{cases}$$
- $$\Pr(\mathbf{r}|\mathbf{c}) = p^{d(\mathbf{r},\mathbf{c})}(1-p)^{n-d(\mathbf{r},\mathbf{c})} = (1-p)^n \left(\frac{p}{1-p}\right)^{d(\mathbf{r},\mathbf{c})},$$

  Since $0 < p < 1/2$ and $0 < p/(1-p) < 1$, we can deduce that maximizing $\Pr(\mathbf{r}|\mathbf{c})$ is equivalent to minimizing $d(\mathbf{r},\mathbf{c})$.



This means that on a BSC, maximum likelihood decoding and nearest decoding are the same.

- This result can be generalized to the q-ary symmetric channel.

- Bit error probability p_b: suppose that a codeword is represented by a binary vector **c** of length N; this is the *average* probability that a bit of **r** is not equal to the corresponding bit of **c**.

- Block error probability p_B: of a code and decoder, for a given channel, and for a given input probability $\Pr(c_i)$, is the probability that at least one symbol of a block of symbols is decoded incorrectly.

- The optimal decoder: is the one that minimizes the probability of block error. If a uniform input distribution on **c** is assumed the optimal decoder is the MLD decoder.

PROBLEM 1

Compute the error probability of the code R_3 (the Repetition code of length 3) for a binary symmetric channel with noise level p.

PROBLEM 1

Compute the error probability of the code R_3 (the Repetition code of length 3) for a binary symmetric channel with noise level p.

Solution: An error is made by R_3 if

1. if 3 bits are flipped, this happens with probability p^3
2. if exactly 2 bits are flipped, this happens with probability $3p^2(1-p)$

$$p_b = p_B = 3p^2(1-p)+p^3 = 3p^2-2p^3 \approx 3p^2$$

PROBLEM 1

Compute the error probability of the code R_3 (the Repetition code of length 3) for a binary symmetric channel with noise level p.

Solution: An error is made by R_3 if

1. if 3 bits are flipped, this happens with probability p^3
2. if exactly 2 bits are flipped, this happens with probability $3p^2(1-p)$

$$p_b = p_B = 3p^2(1-p)+p^3 = 3p^2-2p^3 \approx 3p^2$$

PROBLEM 2

Assuming $p = 0.1$, find how many repetitions are required to get the probability of error down to 10^{-15}.

PROBLEM 1

Compute the error probability of the code R_3 (the Repetition code of length 3) for a binary symmetric channel with noise level p.

Solution: An error is made by R_3 if

1. if 3 bits are flipped, this happens with probability p^3
2. if exactly 2 bits are flipped, this happens with probability $3p^2(1-p)$

$$p_b = p_B = 3p^2(1-p)+p^3 = 3p^2-2p^3 \approx 3p^2$$

PROBLEM 2

Assuming $p = 0.1$, find how many repetitions are required to get the probability of error down to 10^{-15}.
Solution:

$$n \approx 61$$

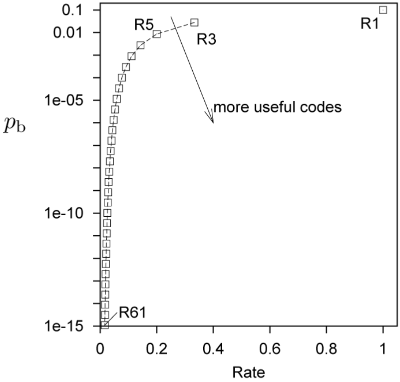
PROBLEM 1

Compute the error probability of the code R_3 (the Repetition code of length 3) for a binary symmetric channel with noise level p.

Solution: An error is made by R_3 if

1. if 3 bits are flipped, this happens with probability p^3
2. if exactly 2 bits are flipped, this happens with probability $3p^2(1-p)$

$$p_b = p_B = 3p^2(1-p)+p^3 = 3p^2-2p^3 \approx 3p^2$$



PROBLEM 2

Assuming $p = 0.1$, find how many repetitions are required to get the probability of error down to 10^{-15}.

Solution:

$$n \approx 61$$

PROBLEM 1

Calculate the probability of block error p_B of the $[7,4,3]_2$ Hamming code as a function of the noise level p.

Solution:

$$p_B \approx 21p^2$$

PROBLEM 1

Calculate the probability of block error p_B of the $[7, 4, 3]_2$ Hamming code as a function of the noise level p.

Solution:

$$p_B \approx 21p^2$$

PROBLEM 2

Calculate the probability of bit error p_b for the same code.

Solution:

$$p_b \approx 9p^2$$

PROBLEM 1

Calculate the probability of block error p_B of the $[7,4,3]_2$ Hamming code as a function of the noise level p.

Solution:

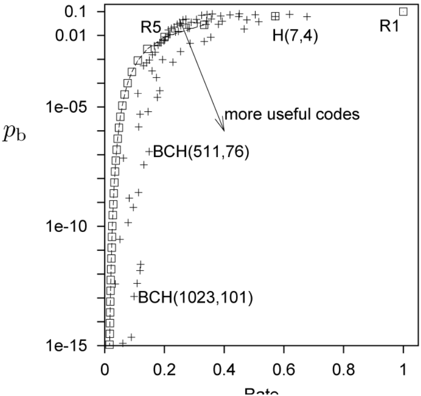
$$p_B \approx 21p^2$$

PROBLEM 2

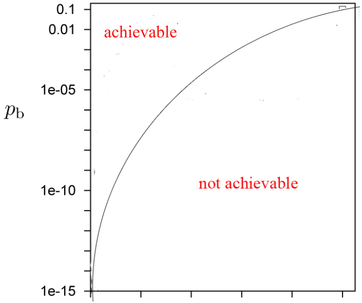
Calculate the probability of bit error p_b for the same code.

Solution:

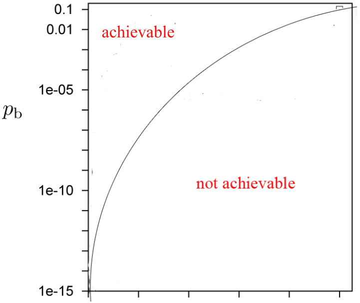
$$p_b \approx 9p^2$$

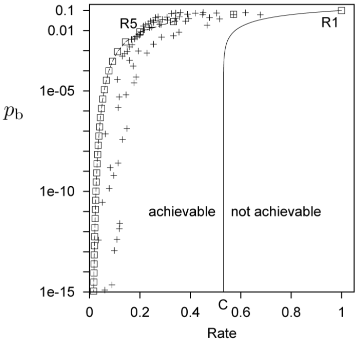


Good codes exist! but. . .



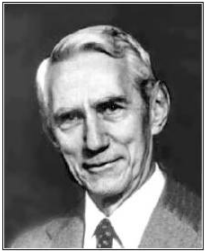
Good codes exist! but. . .





. . . but we do not know how to construct them

Shannon proved that for every DMC with a finite number of inputs and outputs points, one may define the notion of **channel capacity.**



Claude-Shannon 1916-2001

- If C is a code with rate $R > \mathcal{C}$, then the probability of error in decoding this code is bounded away from 0. (In other words, at any rate $R > \mathcal{C}$, reliable communication is not possible.)
- For any information rate $R < \mathcal{C}$ and any $\delta > 0$, there exists a code C of length n_δ and rate R, such that the probability of error in maximum likelihood decoding of this code is at most δ.

  Proof: **Non-constructive!**

How can we find good codes?? Ingredients of Shannon's proof:

- Random code
- Large block length
- Optimal decoding

How can we find good codes?? Ingredients of Shannon's proof:

- Random code
- Large block length
- Optimal decoding

Solution:

- Long, structured, "pseudorandom" codes
- Practical, near-optimal decoding algorithms

How can we find good codes?? Ingredients of Shannon's proof:

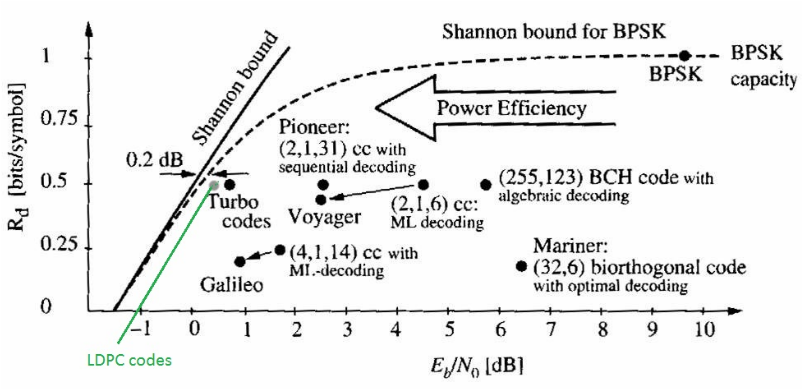
- Random code
- Large block length
- Optimal decoding

Solution:

- Long, structured, "pseudorandom" codes
- Practical, near-optimal decoding algorithms

State-of-art:

- Turbo codes and LDPC codes have brought Shannon limits to within reach on a wide range of channels


Shannon bound for BPSK
BPSK
capacity
BPSK
Shannon bound
Power Efficiency
Pioneer:
(2,1,31) cc with
sequential decoding
0.2 dB
(255,123) BCH code with
algebraic decoding
Turbo
codes
Voyager
(2,1,6) cc:
ML decoding
(4,1,14) cc with
ML-decoding
Galileo
Mariner:
(32,6) biorthogonal code
with optimal decoding
LDPC codes
R_d [bits/symbol]
E_b/N_0 [dB]
1
0.75
0.5
0.25
0
−1
0
1
2
3
4
5
6
7
8
9
10

- LDPC codes are capacity-approaching codes
  - ◇ G.hn/G.9960 (ITU-T Standard for networking over power lines, phone lines and coaxial cable)
  - ◇ 802.3an (10 Giga-bit/s Ethernet over Twisted pair)
  - ◇ DVB-S2 / DVB-T2 / DVB-C2 (Digital video broadcasting, 2nd Generation) and DMB-T/H (Digital video broadcasting)
  - ◇ WiMAX (IEEE 802.16e standard for microwave communications)
  - ◇ IEEE 802.11n-2009 (Wi-Fi standard)