Channel coding

Introduction & linear codes

Manuel A. Vázquez
Jose Miguel Leiva
Joaquín Míguez

February 27, 2024

Index

Index

(Channel) Coding

Goal

Add redundancy to the transmitted information so that it can be recovered if errors happen during transmission.

(Channel) Coding

Goal

Add redundancy to the transmitted information so that it can be recovered if errors happen during transmission.

Example: repetition code

- $0 \to 000$
- $1 \to 111$

so that, e.g.,

$$010 \to 000\,111\,000$$

(Channel) Coding

Goal

Add redundancy to the transmitted information so that it can be recovered if errors happen during transmission.

Example: repetition code

- $0 \rightarrow 000$
- $1 \rightarrow 111$

so that, e.g.,

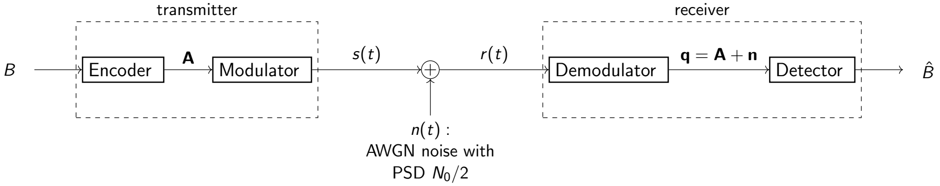
$$010 \rightarrow 000\,111\,000$$

What should we *decide* it was transmitted if we receive

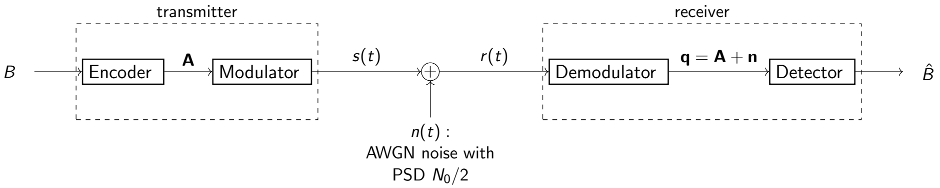
$$010\,100\,000\,?$$

000 (instead of 010)!

Digital communications system

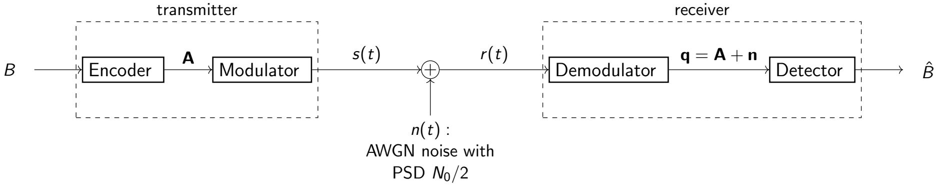


Digital communications system



This model can be analyzed at different levels...

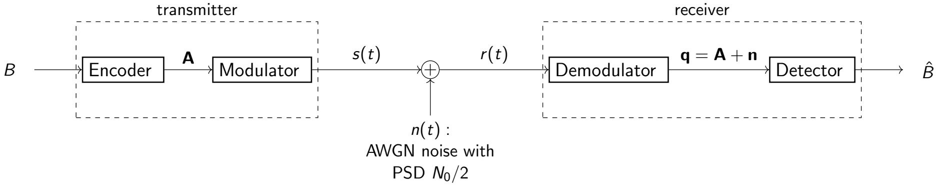
Digital communications system



This model can be analyzed at different levels...

- Digital channel

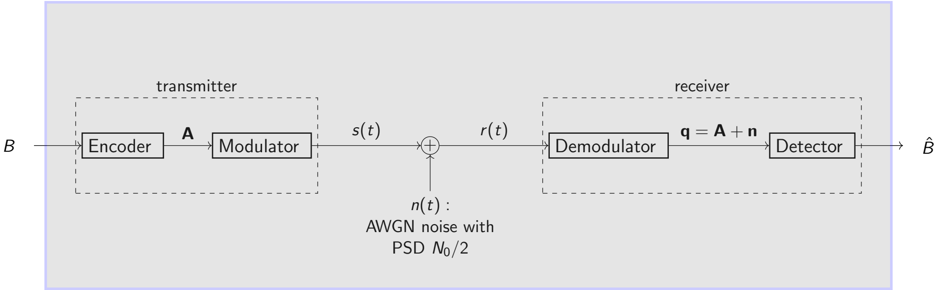
Digital communications system



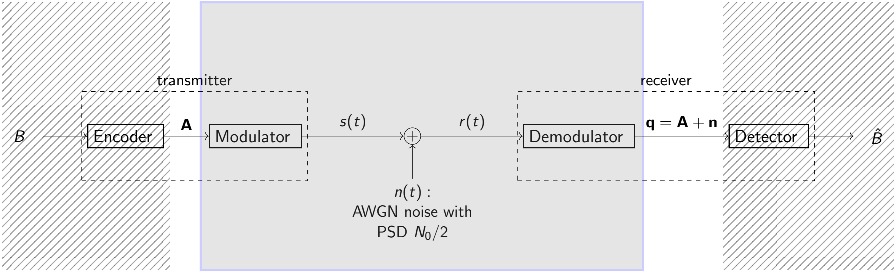
This model can be analyzed at different levels...

- Digital channel
- Gaussian channel

Digital channel

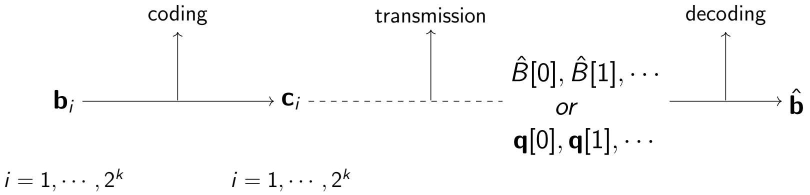


Gaussian channel (with digital input)



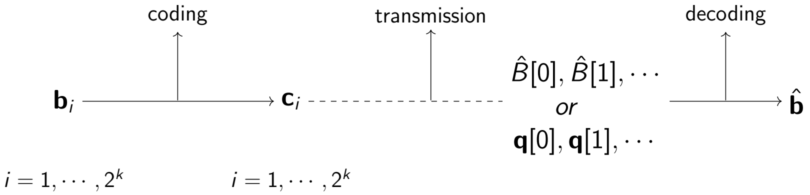
Some basic concepts

- **Code**
  Mapping from a sequence of k bits, $\mathbf{b} \in \{\mathbf{b}_1, \mathbf{b}_2, \cdots\}$, onto another one of $n > k$ bits, $\mathbf{c} \in \{\mathbf{c}_1, \mathbf{c}_2, \cdots\}$.



Some basic concepts

- **Code**
  Mapping from a sequence of k bits, $\mathbf{b} \in \{\mathbf{b}_1, \mathbf{b}_2, \cdots\}$, onto another one of $n > k$ bits, $\mathbf{c} \in \{\mathbf{c}_1, \mathbf{c}_2, \cdots\}$.



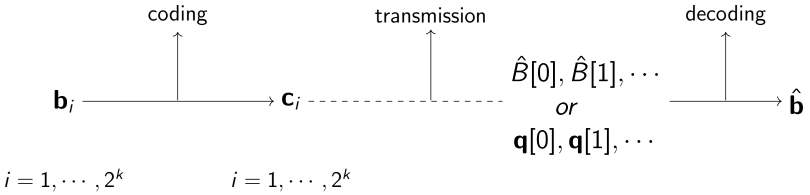
- **Probability of error for $\mathbf{b}_i$**

$$P_e^i = Pr\{\hat{\mathbf{b}} \neq \mathbf{b}_i | \mathbf{b} = \mathbf{b}_i\}, \; i = 1, \ldots, 2^k$$

- **Maximum probability of error**: $P_e^{\max} = \max_i P_e^i$

Some basic concepts

- **Code**
  Mapping from a sequence of k bits, $\mathbf{b} \in \{\mathbf{b}_1, \mathbf{b}_2, \cdots\}$, onto another one of $n > k$ bits, $\mathbf{c} \in \{\mathbf{c}_1, \mathbf{c}_2, \cdots\}$.



- **Probability of error for $\mathbf{b}_i$**

$$P_e^i = Pr\{\hat{\mathbf{b}} \neq \mathbf{b}_i | \mathbf{b} = \mathbf{b}_i\}, \; i = 1, \ldots, 2^k$$

- **Maximum probability of error**: $P_e^{\max} = \max_i P_e^i$
- **Rate**: The rate of a code is the number of information bits, k, carried by a codeword of length n.

$$R = k/n$$

Codeword vs bit error probability

- P_e: codeword error probability

$$P_e = \frac{\#\text{ codewords received incorrectly}}{\text{overall }\#\text{ codewords}} = \frac{v}{w}$$

Codeword vs bit error probability

- P_e: codeword error probability

$$P_e = \frac{\#\ \text{codewords received incorrectly}}{\text{overall}\ \#\ \text{codewords}} = \frac{v}{w}$$

- BER (**B**it **E**rror **R**ate): bit error probability

$$BER = \frac{\#\ \text{incorrect bits}}{\#\ \text{transmitted bits}}$$

(they match if every codeword carries a single information bit)

Codeword vs bit error probability

- P_e: codeword error probability

$$P_e = \frac{\#\text{ codewords received incorrectly}}{\text{overall }\#\text{ codewords}} = \frac{v}{w}$$

- BER (**B**it **E**rror **R**ate): bit error probability

$$BER = \frac{\#\text{ incorrect bits}}{\#\text{ transmitted bits}}$$

(they match if every codeword carries a single information bit)

worst-case scenario $\rightarrow$ $BER = \frac{v \times k}{w \times k} = P_e$

Codeword vs bit error probability

- P_e: codeword error probability

$$P_e = \frac{\#\text{ codewords received incorrectly}}{\text{overall }\#\text{ codewords}} = \frac{v}{w}$$

- BER (**B**it **E**rror **R**ate): bit error probability

$$BER = \frac{\#\text{ incorrect bits}}{\#\text{ transmitted bits}}$$

(they match if every codeword carries a single information bit)

$$\left.\begin{array}{ll} \text{worst-case scenario} \rightarrow & BER = \frac{v \times k}{w \times k} = P_e \\ \text{best-case scenario} \rightarrow & BER = \frac{v \times 1}{w \times k} = \frac{P_e}{k} \end{array}\right\}$$

Codeword vs bit error probability

- P_e: codeword error probability

$$P_e = \frac{\#\text{ codewords received incorrectly}}{\text{overall }\#\text{ codewords}} = \frac{v}{w}$$

- BER (**B**it **E**rror **R**ate): bit error probability

$$BER = \frac{\#\text{ incorrect bits}}{\#\text{ transmitted bits}}$$

(they match if every codeword carries a single information bit)

$$\left.\begin{array}{lll} \text{worst-case scenario} \rightarrow & BER = \frac{v \times k}{w \times k} = P_e \\ \text{best-case scenario} \rightarrow & BER = \frac{v \times 1}{w \times k} = \frac{P_e}{k} \end{array}\right\} \Rightarrow \frac{P_e}{k} \leq BER \leq P_e$$

Channel coding theorem

Theorem: Channel coding (Shannon, 1948)

If C is the capacity of a channel, then it is possible to *reliably* transmit with rate $R < C$.

Channel coding theorem

Theorem: Channel coding (Shannon, 1948)

If C is the capacity of a channel, then it is possible to *reliably* transmit with rate $R < C$.

Capacity

It is the maximum of the mutual information between the input and output of the channel.

Channel coding theorem

Theorem: Channel coding (Shannon, 1948)

If C is the capacity of a channel, then it is possible to *reliably* transmit with rate $R < C$.

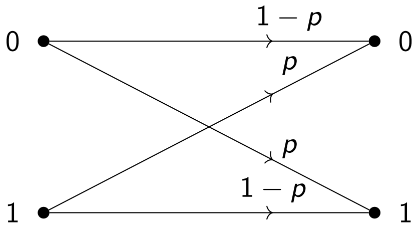
Capacity

It is the maximum of the mutual information between the input and output of the channel.

Reliable transmission

There is a sequence of codes $(n, k) = (n, nR)$ *such that, when* $n \to \infty$, $P_e^{\max} \to 0$.

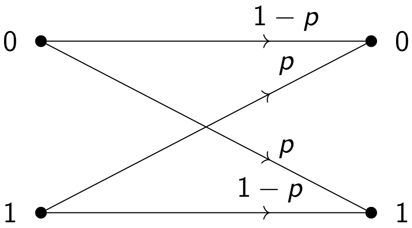
Channel coding theorem: example



$$C = 1 - H_b(p),$$

being p the channel BER and H_b the binary entropy.

Channel coding theorem: example



$$C = 1 - H_b(p),$$

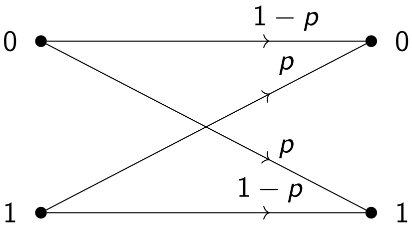
being p the channel BER and H_b the binary entropy.

Let us consider 4 binary channels with

$$p = 0.15 \Rightarrow C_1 = 0.39 \qquad p = 0.13 \Rightarrow C_2 = 0.44$$
$$p = 0.17 \Rightarrow C_3 = 0.34 \qquad p = 0.19 \Rightarrow C_4 = 0.29$$

and a code with rate $R = 1/3 = 0.33$.

Channel coding theorem: example



$$C = 1 - H_b(p),$$

being p the channel BER and H_b the binary entropy.

Let us consider 4 binary channels with

$$p = 0.15 \Rightarrow C_1 = 0.39 \qquad p = 0.13 \Rightarrow C_2 = 0.44$$
$$p = 0.17 \Rightarrow C_3 = 0.34 \qquad p = 0.19 \Rightarrow C_4 = 0.29$$

and a code with rate $R = 1/3 = 0.33$.

Channel coding theorem

A code with rate $R = 1/3$ only respects the Shannon limit in the first three scenarios.

Channel coding theorem: example

The figure shows the evolution of the codeword error probability as a function of n: it approaches 0 when $R < C$.

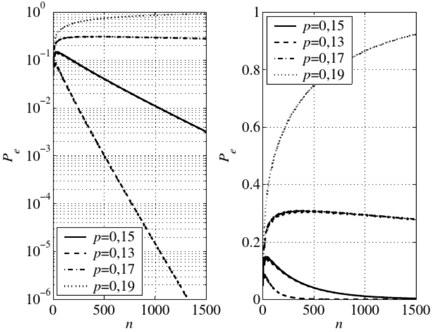


Figure: Left: logarithmic scale; right: linear scale

Definitions

Definition: Redundancy

The number of bits, $r = n - k$, added by the encoder.

It allows rewriting the rate of the code as $R = \frac{k}{n} = \frac{n-r}{n} = 1 - \frac{r}{n}$

Definitions

Definition: Redundancy

The number of bits, $r = n - k$, added by the encoder.

It allows rewriting the rate of the code as $R = \frac{k}{n} = \frac{n-r}{n} = 1 - \frac{r}{n}$

Definition: Hamming distance...

...between two binary sequences is the number of different bits.

It is a measure of how different two sequences of bits are. For instance, $d_H(1010, 1001) = 2$.

Definitions

Definition: Redundancy

The number of bits, $r = n - k$, added by the encoder.

It allows rewriting the rate of the code as $R = \frac{k}{n} = \frac{n-r}{n} = 1 - \frac{r}{n}$

Definition: Hamming distance...

...between two binary sequences is the number of different bits.

It is a measure of how different two sequences of bits are. For instance, $d_H(1010, 1001) = 2$.

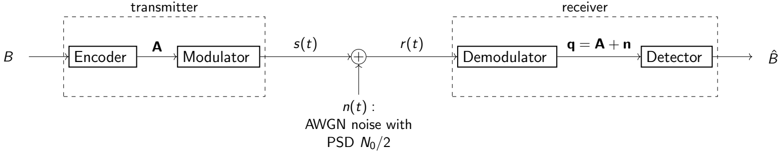
Definition: Minimum distance of a code

$$d_{min} = \min_{i \neq j} d_H(\mathbf{c_i}, \mathbf{c_j})$$

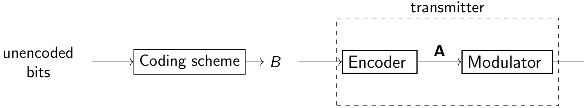
Index

Coding

In the usual model for a digital communications system,



the coding scheme is always placed *before* the system



and we have

$$\left.\begin{aligned} B[0] &= C[0] \\ B[1] &= C[1] \\ \vdots & \quad \vdots \end{aligned}\right\} \text{codeword}$$

Index

Hard decoding

- Decoding at the *bit level*

Hard decoding

- Decoding at the *bit level*
- It relies on the digital channel

Hard decoding

- Decoding at the *bit level*
- It relies on the digital channel

- The input to the decoder are bits coming from the Detector, the $\hat{B}$'s.

Hard decoding

- Decoding at the *bit level*
- It relies on the digital channel

- The input to the decoder are bits coming from the Detector, the $\hat{B}$'s.
- Metric is the **Hamming distance**.

Hard decoding

- Decoding at the *bit level*
- It relies on the digital channel

B — Digital channel — $\hat{B}$

- The input to the decoder are bits coming from the Detector, the $\hat{B}$'s.
- Metric is the **Hamming distance**.

Notation

$$\mathbf{c}_i = \left[C^i[0], C^i[1], \cdots C^i[n-1]\right] \equiv i\text{-th codeword}$$

$$\mathbf{r} = \left[\hat{B}[0], \hat{B}[1], \cdots \hat{B}[n-1]\right] \equiv \text{received word}$$

Hard decoding: decision rule

- Maximum a Posteriori (MAP) rule: we decide $\mathbf{c}_i$ if

$$p(\mathbf{c}_i|\mathbf{r}) > p(\mathbf{c}_j|\mathbf{r}) \qquad \forall j \neq i$$

Hard decoding: decision rule

- Maximum a Posteriori (MAP) rule: we decide $\mathbf{c}_i$ if

$$p(\mathbf{c}_i|\mathbf{r}) > p(\mathbf{c}_j|\mathbf{r}) \qquad \forall j \neq i$$

- If all the codewords are equally likely, it is equivalent to Maximum Likelihood (ML),

$$p(\mathbf{r}|\mathbf{c}_i) > p(\mathbf{r}|\mathbf{c}_j) \qquad \forall j \neq i$$

Hard decoding: decision rule

- Maximum a Posteriori (MAP) rule: we decide $\mathbf{c}_i$ if

$$p(\mathbf{c}_i|\mathbf{r}) > p(\mathbf{c}_j|\mathbf{r}) \qquad \forall j \neq i$$

- If all the codewords are equally likely, it is equivalent to Maximum Likelihood (ML),

$$p(\mathbf{r}|\mathbf{c}_i) > p(\mathbf{r}|\mathbf{c}_j) \qquad \forall j \neq i$$

  - Likelihoods can be expressed in terms of d_H

$$p(\mathbf{r}|\mathbf{c}_i) = \epsilon^{d_H(\mathbf{r},\mathbf{c}_i)}(1-\epsilon)^{n-d_H(\mathbf{r},\mathbf{c}_i)}$$

$\epsilon \equiv$ *channel* bit error probability

Hard decoding: decision rule

- Maximum a Posteriori (MAP) rule: we decide $\mathbf{c}_i$ if

$$p(\mathbf{c}_i|\mathbf{r}) > p(\mathbf{c}_j|\mathbf{r}) \qquad \forall j \neq i$$

- If all the codewords are equally likely, it is equivalent to Maximum Likelihood (ML),

$$p(\mathbf{r}|\mathbf{c}_i) > p(\mathbf{r}|\mathbf{c}_j) \qquad \forall j \neq i$$

  - Likelihoods can be expressed in terms of d_H

$$p(\mathbf{r}|\mathbf{c}_i) = \epsilon^{d_H(\mathbf{r},\mathbf{c}_i)}(1-\epsilon)^{n-d_H(\mathbf{r},\mathbf{c}_i)}$$

  $\epsilon \equiv$ *channel* bit error probability

- If $\epsilon < 0.5$ ML rule is tantamount to deciding $\mathbf{c}_i$ if

$$d_H(\mathbf{r},\mathbf{c}_i) < d_H(\mathbf{r},\mathbf{c}_j) \qquad \forall j \neq i.$$

Hard decoding: error detection vs. correction

Assuming errors happened during transmission, there are two possible scenarios:

Hard decoding: error detection vs. correction

Assuming errors happened during transmission, there are two possible scenarios:

- We do **not** detect them

Hard decoding: error detection vs. correction

Assuming errors happened during transmission, there are two possible scenarios:

- We do **not** detect them
  (we only detect errors if $\mathbf{r} \neq \mathbf{c}_i \quad i = 1, \ldots, 2^k$)

Hard decoding: error detection vs. correction

Assuming errors happened during transmission, there are two possible scenarios:

- We do **not** detect them
  (we only detect errors if $\mathbf{r} \neq \mathbf{c}_i \quad i = 1, \ldots, 2^k$)
- We do detect them, in which case we must make a decision:

Hard decoding: error detection vs. correction

Assuming errors happened during transmission, there are two possible scenarios:

- We do **not** detect them
  (we only detect errors if $\mathbf{r} \neq \mathbf{c}_i \quad i = 1, \ldots, 2^k$)
- We do detect them, in which case we must make a decision:
  - We don't risk correct them and request a *retransmission*
    (we **cannot** correct *with confidence*)

Hard decoding: error detection vs. correction

Assuming errors happened during transmission, there are two possible scenarios:

- We do **not** detect them
  (we only detect errors if $\mathbf{r} \neq \mathbf{c}_i \quad i = 1, \ldots, 2^k$)
- We do detect them, in which case we must make a decision:
  - We don't risk correct them and request a *retransmission*
    (we **cannot** correct *with confidence*)
  - we *try* and correct them
    (a risk is is involved!!)

Hard decoding: error detection vs. correction

Assuming errors happened during transmission, there are two possible scenarios:

- We do **not** detect them
  (we only detect errors if $\mathbf{r} \neq \mathbf{c}_i \quad i = 1, \ldots, 2^k$)
- We do detect them, in which case we must make a decision:
  - We don't risk correct them and request a *retransmission*
    (we **cannot** correct *with confidence*)
  - we *try* and correct them
    (a risk is is involved!!)

We need a *policy* for the latter scenario: in this course we **always** try and fix the errors.

Hard decoding: detection

- We detect a word error when **less than** d_{min} bit errors happen.

Hard decoding: detection

- We detect a word error when **less than** d_{min} bit errors happen.
- Probability of an erroneous codeword going **undetected** (at least d_{min} bit errors)

$$P_{nd} \leq \sum_{m=d_{min}}^{n} \binom{n}{m} \epsilon^m (1-\epsilon)^{n-m}$$

where ϵ is the bit error probability in the system, and d_{min} is the minimum distance between codewords.

Hard decoding: detection

- We detect a word error when **less than** d_{min} bit errors happen.
- Probability of an erroneous codeword going **undetected** (at least d_{min} bit errors)

$$P_{nd} \leq \sum_{m=d_{min}}^{n} \binom{n}{m} \epsilon^{m} (1-\epsilon)^{n-m}$$

where ϵ is the bit error probability in the system, and d_{min} is the minimum distance between codewords.

A bound on the probability of error...

...since it might happen that d_{min} bit errors do not turn a codeword into another one $\Rightarrow \leq$ rather than $=$

Hard decoding: correction ("always correct" policy)

- Decoding is correct if there are less than $d_{min}/2$ erroneous bits
  $\Rightarrow$ the code can correct **up to**

$$t = \lfloor (d_{min} - 1)/2 \rfloor \text{ errors.}$$

Hard decoding: correction ("always correct" policy)

- Decoding is correct if there are less than $d_{min}/2$ erroneous bits $\Rightarrow$ the code can correct **up to**

$$t = \lfloor (d_{min} - 1)/2 \rfloor \text{ errors.}$$

- Error correction probability:

$$P_e \leq \sum_{m=t+1}^{n} \binom{n}{m} \epsilon^m (1-\epsilon)^{n-m}$$

Hard decoding: correction ("always correct" policy)

- Decoding is correct if there are less than $d_{min}/2$ erroneous bits $\Rightarrow$ the code can correct **up to**

$$t = \lfloor (d_{min} - 1)/2 \rfloor \text{ errors.}$$

- Error correction probability:

$$P_e \leq \sum_{m=t+1}^{n} \binom{n}{m} \epsilon^m (1-\epsilon)^{n-m}$$

A bound on the probability of error...

...since it is possible to correct more than t errors (there is no guarantee, though) $\Rightarrow \leq$ rather than $=$

Hard decoding: correction ("always correct" policy)

- Decoding is correct if there are less than $d_{min}/2$ erroneous bits $\Rightarrow$ the code can correct **up to**

$$t = \lfloor (d_{min} - 1)/2 \rfloor \text{ errors.}$$

- Error correction probability:

$$P_e \leq \sum_{m=t+1}^{n} \binom{n}{m} \epsilon^m (1-\epsilon)^{n-m}$$

A bound on the probability of error...

...since it is possible to correct more than t errors (there is no guarantee, though) $\Rightarrow \leq$ rather than $=$

Approximate bound

The first element in the summation is a good approximation if ϵ is small and d_{min} large.

Soft decoding

- Decoding at the *element from the constellation level*

Soft decoding

- Decoding at the *element from the constellation level*
- It relies on the Gaussian channel

with

$$\mathbf{q} = \mathbf{A} + \mathbf{n}$$

where $\mathbf{n}$ is a Gaussian noise vector.

Soft decoding

- Decoding at the *element from the constellation level*
- It relies on the Gaussian channel

with

$$\mathbf{q} = \mathbf{A} + \mathbf{n}$$

where $\mathbf{n}$ is a Gaussian noise vector.

- The input to the decoder are the observations coming from the Demodulator, the $\mathbf{q}$'s.

Soft decoding

- Decoding at the *element from the constellation level*
- It relies on the Gaussian channel

  with

$$\mathbf{q} = \mathbf{A} + \mathbf{n}$$

  where $\mathbf{n}$ is a Gaussian noise vector.
- The input to the decoder are the observations coming from the Demodulator, the $\mathbf{q}$'s.
- Metric is **Euclidean distance**

Soft decoding

- Decoding at the *element from the constellation level*
- It relies on the Gaussian channel

  with

$$\mathbf{q} = \mathbf{A} + \mathbf{n}$$

  where **n** is a Gaussian noise vector.
- The input to the decoder are the observations coming from the Demodulator, the **q**'s.
- Metric is **Euclidean distance**

Notation

$$m \equiv \# \text{ bits carried by every } \mathbf{A}$$

$$\tilde{\mathbf{c}}_i = \left[\mathbf{A}^{(i)}[0], \mathbf{A}^{(i)}[1], \cdots \mathbf{A}^{(i)}[n/m-1]\right] \equiv i\text{-th codeword}$$

$$\tilde{\mathbf{r}} = [\mathbf{q}[0], \mathbf{q}[1], \cdots \mathbf{q}[n/m-1]] \equiv \text{received word}$$

Soft decoding: correction

- The *codeword* error probability can be approximated as

$$P_e \approx \kappa \mathrm{Q}\left(\frac{d_{min}/2}{\sqrt{N_0/2}}\right) \quad (1)$$

where κ is the *kiss number*.

Definition: kiss number

It is the maximum number of codewords that are at distance d_{min} from any given.

Coding gain

- If we set equal the *BER* with and without coding, the **coding gain** is obtained as

$$G = \frac{(E_b/N_0)_{nc}}{(E_b/N_0)_c}$$

Coding gain

- If we set equal the *BER* with and without coding, the **coding gain** is obtained as

$$G = \frac{(E_b/N_0)_{nc}}{(E_b/N_0)_c}$$

- Different for soft and hard decoding

Coding gain

- If we set equal the *BER* with and without coding, the **coding gain** is obtained as

$$G = \frac{(E_b/N_0)_{nc}}{(E_b/N_0)_c}$$

- Different for soft and hard decoding

To compute the individual E_b/N_0's, it is often useful...

Stirling's approximation

$$Q(x) \approx \frac{1}{2} e^{-\frac{x^2}{2}}$$

Coding gain: example

Let us consider a binary antipodal constellation 2-PAM ($\pm\sqrt{E_s}$), with the code

$\mathbf{b}_i$	$\mathbf{c}_i$
00	000
01	011
10	110
11	101

Coding gain: example - hard decoding

- This code cannot correct any error since $t = \lfloor (d_{min} - 1)/2 \rfloor = 0$, and the codeword error probability is

$$P_e \leq \sum_{m=1}^{3} \binom{3}{m} \epsilon^m (1-\epsilon)^{n-m} \approx 3\epsilon$$

where $\epsilon = Q(\sqrt{2E_s/N_0})$.

Coding gain: example - hard decoding

- This code cannot correct any error since $t = \lfloor (d_{min} - 1)/2 \rfloor = 0$, and the codeword error probability is

$$P_e \leq \sum_{m=1}^{3} \binom{3}{m} \epsilon^m (1-\epsilon)^{n-m} \approx 3\epsilon$$

where $\epsilon = Q(\sqrt{2E_s/N_0})$.

- Bit error probability

$$BER \approx \frac{2}{3} 3Q\left(\sqrt{\frac{2E_s}{N_0}}\right)$$

Coding gain: example - hard decoding

- This code cannot correct any error since $t = \lfloor (d_{min} - 1)/2 \rfloor = 0$, and the codeword error probability is

$$P_e \leq \sum_{m=1}^{3} \binom{3}{m} \epsilon^m (1-\epsilon)^{n-m} \approx 3\epsilon$$

where $\epsilon = Q(\sqrt{2E_s/N_0})$.

- Bit error probability

$$BER \approx \frac{2}{3} 3Q\left(\sqrt{\frac{2E_s}{N_0}}\right)$$

- In order to express it in terms of E_b, we use that $2E_b = 3E_s$, and hence

$$BER \approx 2Q\left(\sqrt{\frac{4E_b}{3N_0}}\right)$$

Coding gain: example - soft decoding

- We decide **b** from the output of the Gaussian channel,

$$\mathbf{q} = (\mathbf{q}[0], \mathbf{q}[1], \mathbf{q}[2]) = (\mathbf{A}[0] + \mathbf{n}[0], \mathbf{A}[1] + \mathbf{n}[1], \mathbf{A}[2] + \mathbf{n}[2])$$

Coding gain: example - soft decoding

- We decide **b** from the output of the Gaussian channel,

$$\mathbf{q} = (\mathbf{q}[0], \mathbf{q}[1], \mathbf{q}[2]) = (\mathbf{A}[0] + \mathbf{n}[0], \mathbf{A}[1] + \mathbf{n}[1], \mathbf{A}[2] + \mathbf{n}[2])$$

- Tantamount to the detector for the constellation

$$\begin{pmatrix} -\sqrt{E_s} \\ -\sqrt{E_s} \\ -\sqrt{E_s} \end{pmatrix}, \quad \begin{pmatrix} -\sqrt{E_s} \\ \sqrt{E_s} \\ \sqrt{E_s} \end{pmatrix}, \quad \begin{pmatrix} \sqrt{E_s} \\ \sqrt{E_s} \\ -\sqrt{E_s} \end{pmatrix}, \quad \begin{pmatrix} \sqrt{E_s} \\ -\sqrt{E_s} \\ \sqrt{E_s} \end{pmatrix}$$

which has minimum (Euclidean) distance $d_{min} = 2\sqrt{2E_s}$

Coding gain: example - soft decoding

- We decide $\mathbf{b}$ from the output of the Gaussian channel,

$$\mathbf{q} = (\mathbf{q}[0], \mathbf{q}[1], \mathbf{q}[2]) = (\mathbf{A}[0] + \mathbf{n}[0], \mathbf{A}[1] + \mathbf{n}[1], \mathbf{A}[2] + \mathbf{n}[2])$$

- Tantamount to the detector for the constellation

$$\begin{pmatrix} -\sqrt{E_s} \\ -\sqrt{E_s} \\ -\sqrt{E_s} \end{pmatrix}, \quad \begin{pmatrix} -\sqrt{E_s} \\ \sqrt{E_s} \\ \sqrt{E_s} \end{pmatrix}, \quad \begin{pmatrix} \sqrt{E_s} \\ \sqrt{E_s} \\ -\sqrt{E_s} \end{pmatrix}, \quad \begin{pmatrix} \sqrt{E_s} \\ -\sqrt{E_s} \\ \sqrt{E_s} \end{pmatrix}$$

  which has minimum (Euclidean) distance $d_{min} = 2\sqrt{2E_s}$
- From (1) the codeword error probability is

$$P_e \approx 3Q\left(\sqrt{\frac{4E_s}{N_0}}\right)$$

Coding gain: example - soft decoding

- We decide $\mathbf{b}$ from the output of the Gaussian channel,

$$\mathbf{q} = (\mathbf{q}[0], \mathbf{q}[1], \mathbf{q}[2]) = (\mathbf{A}[0] + \mathbf{n}[0], \mathbf{A}[1] + \mathbf{n}[1], \mathbf{A}[2] + \mathbf{n}[2])$$

- Tantamount to the detector for the constellation

$$\begin{pmatrix} -\sqrt{E_s} \\ -\sqrt{E_s} \\ -\sqrt{E_s} \end{pmatrix}, \quad \begin{pmatrix} -\sqrt{E_s} \\ \sqrt{E_s} \\ \sqrt{E_s} \end{pmatrix}, \quad \begin{pmatrix} \sqrt{E_s} \\ \sqrt{E_s} \\ -\sqrt{E_s} \end{pmatrix}, \quad \begin{pmatrix} \sqrt{E_s} \\ -\sqrt{E_s} \\ \sqrt{E_s} \end{pmatrix}$$

  which has minimum (Euclidean) distance $d_{min} = 2\sqrt{2E_s}$
- From (1) the codeword error probability is

$$P_e \approx 3Q\left(\sqrt{\frac{4E_s}{N_0}}\right)$$

- BER as a function of E_b:

$$BER \approx 2Q\left(\sqrt{\frac{8E_b}{3N_0}}\right)$$

Coding gain: example - hard vs soft decoding

- Without coding, we have $E_b = E_s$, and

$$\mathrm{BER}_{nc} = \epsilon = Q\left(\sqrt{2E_b/N_0}\right)$$

Coding gain: example - hard vs soft decoding

- Without coding, we have $E_b = E_s$, and

$$\mathrm{BER}_{nc} = \epsilon = Q\left(\sqrt{2E_b/N_0}\right)$$

- Gain with hard decoding
  - We set equal BER_c and BER_{nc}
  - Approximation: $Q(\cdot)$

$$G = \frac{(E_b/N_0)_{nc}}{(E_b/N_0)_c} = 2/3 \approx -1.76dB$$

  - We are actually losing performance!! (expected, since the code is not able correct any error)

Coding gain: example - hard vs soft decoding

- Without coding, we have $E_b = E_s$, and

$$\mathrm{BER}_{nc} = \epsilon = Q\left(\sqrt{2E_b/N_0}\right)$$

- Gain with hard decoding
  - We set equal BER_c and BER_{nc}
  - Approximation: $Q(\cdot)$

$$G = \frac{(E_b/N_0)_{nc}}{(E_b/N_0)_c} = 2/3 \approx -1.76dB$$

  - We are actually losing performance!! (expected, since the code is not able correct any error)
- Soft decoding

$$G = 4/3 \approx 1.25dB$$

  - Now we are making good use of coding

Index

Linear block codes

Galois field modulo 2 ($GF(2)$)

$$a + b = (a + b)_2$$
$$a \cdot b = (a \cdot b)_2$$

Linear block codes

Galois field modulo 2 ($GF(2)$)

$$a + b = (a + b)_2$$
$$a \cdot b = (a \cdot b)_2$$

Definition: Linear Block Code

A linear block code is a code in which any linear combination of codewords is also a codeword.

Linear block codes

Galois field modulo 2 ($GF(2)$)

$$a + b = (a + b)_2$$
$$a \cdot b = (a \cdot b)_2$$

Definition: Linear Block Code

A linear block code is a code in which any linear combination of codewords is also a codeword.

Properties

- It is a subspace in $GF(2)^n$ with 2^k elements.

Linear block codes

Galois field modulo 2 ($GF(2)$)

$$a + b = (a + b)_2$$
$$a \cdot b = (a \cdot b)_2$$

Definition: Linear Block Code

A linear block code is a code in which any linear combination of codewords is also a codeword.

Properties

- It is a subspace in $GF(2)^n$ with 2^k elements.
- The all-zeros word is a codeword.

Linear block codes

Galois field modulo 2 ($GF(2)$)

$$a + b = (a + b)_2$$
$$a \cdot b = (a \cdot b)_2$$

Definition: Linear Block Code

A linear block code is a code in which any linear combination of codewords is also a codeword.

Properties

- It is a subspace in $GF(2)^n$ with 2^k elements.
- The all-zeros word is a codeword.
- Every codeword has at least another codeword that is at d_{min} from it.

Linear block codes

Galois field modulo 2 ($GF(2)$)

$$a + b = (a + b)_2$$
$$a \cdot b = (a \cdot b)_2$$

Definition: Linear Block Code

A linear block code is a code in which any linear combination of codewords is also a codeword.

Properties

- It is a subspace in $GF(2)^n$ with 2^k elements.
- The all-zeros word is a codeword.
- Every codeword has at least another codeword that is at d_{min} from it.
- d_{min} is the smallest weight (number of 1s) among the non-null codewords.

Linear block codes: structure

Elements in an (n, k) linear block code

Linear block codes: structure

Elements in an (n, k) linear block code

- $\mathbf{b}$ is the message, $[\qquad]_{1 \times k}$

Linear block codes: structure

Elements in an (n, k) linear block code

- **b** is the message, $[\ \ \ \ ]_{1 \times k}$
- **c** is the codeword, $[\ \ \ \ ]_{1 \times n}$

Linear block codes: structure

Elements in an (n, k) linear block code

- $\mathbf{b}$ is the message, $[\quad]_{1 \times k}$
- $\mathbf{c}$ is the codeword, $[\quad]_{1 \times n}$
- $\mathbf{r}$ is the received word, $[\quad]_{1 \times n}$ with

$$\mathbf{r} = \mathbf{c} + \mathbf{e}$$

  - $\mathbf{e}$ is the noise $[\quad]_{1 \times n}$

Linear block codes: structure

Elements in an (n, k) linear block code

- **b** is the message, $[\ \ ]_{1 \times k}$
- **c** is the codeword, $[\ \ ]_{1 \times n}$
- **r** is the received word, $[\ \ ]_{1 \times n}$ with

$$\mathbf{r} = \mathbf{c} + \mathbf{e}$$

  - **e** is the noise $[\ \ ]_{1 \times n}$
- **G** is the **generator** matrix,

(for encoding)

$[\ \ ]_{k \times n}$

Linear block codes: structure

Elements in an (n, k) linear block code

- $\mathbf{b}$ is the message, $[\ ]_{1 \times k}$
- $\mathbf{c}$ is the codeword, $[\ ]_{1 \times n}$
- $\mathbf{r}$ is the received word, $[\ ]_{1 \times n}$ with

$$\mathbf{r} = \mathbf{c} + \mathbf{e}$$

  - $\mathbf{e}$ is the noise $[\ ]_{1 \times n}$
- $\mathbf{G}$ is the **generator** matrix,

  (for encoding) $[\ ]_{k \times n}$

- $\mathbf{H}$ is the **parity-check** matrix,

  (for decoding) $[\ ]_{n - k \times n}$

Encoding

The mapping $\mathbf{b} \to \mathbf{c}$ is performed through matrix multiplication i.e.,

$$\mathbf{c} = \mathbf{b}\mathbf{G}.$$

Encoding

The mapping $\mathbf{b} \rightarrow \mathbf{c}$ is performed through matrix multiplication i.e.,

$$\mathbf{c} = \mathbf{b}\mathbf{G}.$$

Keep in mind:

- $\mathbf{b}$ is $1 \times k$
- $\mathbf{G}$ is $k \times n$
- $\mathbf{c}$ is $1 \times n$

Encoding

The mapping $\mathbf{b} \to \mathbf{c}$ is performed through matrix multiplication i.e.,

$$\mathbf{c} = \mathbf{b}\mathbf{G}.$$

Keep in mind:

- $\mathbf{b}$ is $1 \times k$
- $\mathbf{G}$ is $k \times n$
- $\mathbf{c}$ is $1 \times n$

Property

Every row of $\mathbf{G}$ is a codeword.

Parity-check matrix

Parity check matrix, $\mathbf{H}$, is the *orthogonal complement* of $\mathbf{G}$ so that

$$\mathbf{c}\mathbf{H}^\top = \mathbf{0} \Leftrightarrow \mathbf{c} \text{ is a codeword}$$

Parity-check matrix

Parity check matrix, $\mathbf{H}$, is the *orthogonal complement* of $\mathbf{G}$ so that

$$\mathbf{c}\mathbf{H}^\top = \mathbf{0} \Leftrightarrow \mathbf{c} \text{ is a codeword}$$

For the sake of convenience,

Definition: Syndrome

The syndrome of the received sequence $\mathbf{r}$ is

$$\mathbf{s} = \mathbf{r}\mathbf{H}^\top \quad \text{(with dimensions } 1 \times (n-k))$$

Then,

$$\mathbf{s} = \mathbf{0} \Leftrightarrow \mathbf{r} \text{ is a codeword.}$$

Parity-check matrix

Parity check matrix, $\mathbf{H}$, is the *orthogonal complement* of $\mathbf{G}$ so that

$$\mathbf{c}\mathbf{H}^\top = \mathbf{0} \Leftrightarrow \mathbf{c} \text{ is a codeword}$$

For the sake of convenience,

Definition: Syndrome

The syndrome of the received sequence $\mathbf{r}$ is

$$\mathbf{s} = \mathbf{r}\mathbf{H}^\top \quad \text{(with dimensions } 1 \times (n-k))$$

Then,

$$\mathbf{s} = \mathbf{0} \Leftrightarrow \mathbf{r} \text{ is a codeword.}$$

Syndrome-error connection

$$\mathbf{s} = \mathbf{r}\mathbf{H}^T = (\mathbf{c}+\mathbf{e})\mathbf{H}^T = \cancelto{0}{\mathbf{c}\mathbf{H}^T} + \mathbf{e}\mathbf{H}^T = \mathbf{e}\mathbf{H}^T$$

Hard decoding: syndrome decoding

The minimum distance rule requires computing d_H between the received word, $\mathbf{r}$, and every codeword...but we can carry out **syndrome** decoding

Hard decoding: syndrome decoding

The minimum distance rule requires computing d_H between the received word, $\mathbf{r}$, and every codeword...but we can carry out **syndrome** decoding

Beforehand:

Fill up a table yielding the syndrome associated with every possible error,

error ($\mathbf{e}$)	syndrome($\mathbf{s}$)
$\vdots$	$\vdots$

(If several errors yield the same syndrome, choose the one that is most likely, i.e., the one with the smallest weight)

Hard decoding: syndrome decoding

The minimum distance rule requires computing d_H between the received word, **r**, and every codeword...but we can carry out **syndrome** decoding

Beforehand:

Fill up a table yielding the syndrome associated with every possible error,

error (**e**)	syndrome(**s**)
⋮	⋮

(If several errors yield the same syndrome, choose the one that is most likely, i.e., the one with the smallest weight)

In operation: given the received word, **r**:

Hard decoding: syndrome decoding

The minimum distance rule requires computing d_H between the received word, $\mathbf{r}$, and every codeword...but we can carry out **syndrome** decoding

Beforehand:

Fill up a table yielding the syndrome associated with every possible error,

error ($\mathbf{e}$)	syndrome($\mathbf{s}$)
⋮	⋮

(If several errors yield the same syndrome, choose the one that is most likely, i.e., the one with the smallest weight)

In operation: given the received word, $\mathbf{r}$:

1. Compute the syndrome $\mathbf{s} = \mathbf{r}\mathbf{H}^T$.

Hard decoding: syndrome decoding

The minimum distance rule requires computing d_H between the received word, **r**, and every codeword...but we can carry out **syndrome** decoding

Beforehand:

Fill up a table yielding the syndrome associated with every possible error,

error ($\mathbf{e}$)	syndrome($\mathbf{s}$)
$\vdots$	$\vdots$

(If several errors yield the same syndrome, choose the one that is most likely, i.e., the one with the smallest weight)

In operation: given the received word, **r**:

1. Compute the syndrome $\mathbf{s} = \mathbf{r}\mathbf{H}^T$.
2. Look up the table for the error pattern, **e**, with that syndrome

Hard decoding: syndrome decoding

The minimum distance rule requires computing d_H between the received word, **r**, and every codeword...but we can carry out **syndrome** decoding

Beforehand:

Fill up a table yielding the syndrome associated with every possible error,

error ($\mathbf{e}$)	syndrome($\mathbf{s}$)
$\vdots$	$\vdots$

(If several errors yield the same syndrome, choose the one that is most likely, i.e., the one with the smallest weight)

In operation: given the received word, **r**:

1. Compute the syndrome $\mathbf{s} = \mathbf{r}\mathbf{H}^T$.
2. Look up the table for the error pattern, **e**, with that syndrome
3. *Undo* the error

$$\hat{\mathbf{c}} = \mathbf{r} + \mathbf{e}$$

Systematic codes

Definition: Systematic code

A code in which the message is always embedded in the encoded sequence (in the same place).

Systematic codes

Definition: Systematic code

A code in which the message is always embedded in the encoded sequence (in the same place).

This can be easily imposed through the generator matrix,

$$\mathbf{G} = \begin{bmatrix} \mathbf{I}_k & \mathbf{P} \end{bmatrix} \quad \text{or} \quad \mathbf{G} = \begin{bmatrix} \mathbf{P} & \mathbf{I}_k \end{bmatrix}$$

Systematic codes

Definition: Systematic code

A code in which the message is always embedded in the encoded sequence (in the same place).

This can be easily imposed through the generator matrix,

$$\mathbf{G} = \begin{bmatrix} \mathbf{I}_k & \mathbf{P} \end{bmatrix} \quad \text{or} \quad \mathbf{G} = \begin{bmatrix} \mathbf{P} & \mathbf{I}_k \end{bmatrix}$$

- First/last k bits in $\mathbf{c}$ are equal to $\mathbf{b}$, and the remaining $n - k$ are redundancy.

Systematic codes

Definition: Systematic code

A code in which the message is always embedded in the encoded sequence (in the same place).

This can be easily imposed through the generator matrix,

$$\mathbf{G} = \begin{bmatrix} \mathbf{I}_k & \mathbf{P} \end{bmatrix} \quad \text{or} \quad \mathbf{G} = \begin{bmatrix} \mathbf{P} & \mathbf{I}_k \end{bmatrix}$$

- First/last k bits in $\mathbf{c}$ are equal to $\mathbf{b}$, and the remaining $n-k$ are redundancy.
- If $\mathbf{G} = \begin{bmatrix} \mathbf{I}_k & \mathbf{P} \end{bmatrix}$ it can be shown

$$\mathbf{H} = \begin{bmatrix} \mathbf{P}^T & \mathbf{I}_{n-k} \end{bmatrix}$$

Exercise

Prove it!

Systematic code example: Hamming $(7,4)$

generator matrix:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Parity-check matrix:

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Systematic code example: Hamming $(7,4)$

generator matrix:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Parity-check matrix:

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Every Hamming code:

Systematic code example: Hamming (7, 4)

generator matrix:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Parity-check matrix:

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Every Hamming code:

- It's *perfect*

Systematic code example: Hamming $(7,4)$

generator matrix:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Parity-check matrix:

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Every Hamming code:

- It's *perfect*
- $d_{min} = 3$

Systematic code example: Hamming $(7, 4)$

generator matrix:

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

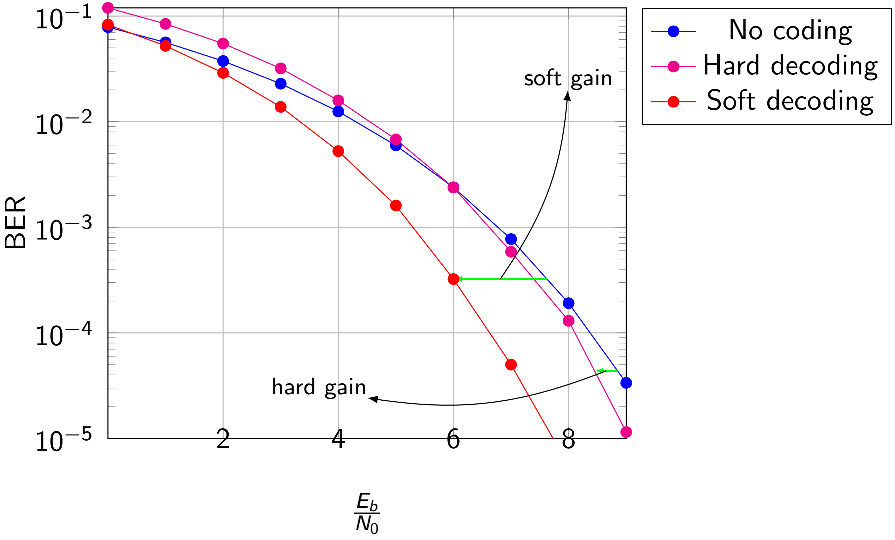
Parity-check matrix:

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Every Hamming code:

- It's *perfect*
- $d_{min} = 3$
- $k = 2^j - j - 1$ and $n = 2^j - 1 \; \forall j \in \mathbb{N} \geq 2$
  - $j = 2 \rightarrow (3, 1)$
  - $j = 3 \rightarrow (7, 4)$
  - $j = 4 \rightarrow (15, 11)$

Hamming $(7, 4)$: coding gain



Hamming $(7,4)$: decoding

Beforehand we apply

$$\mathbf{s} = \mathbf{e}\mathbf{H}^T$$

over every **e** that entails a single error (the code can only correct 1 erroneous bit):

error	syndrome
0000000	000
1000000	101
0100000	110
0010000	111
0001000	011
0000100	100
0000010	010
0000001	001

Hamming $(7,4)$: decoding

Beforehand we apply

$$\mathbf{s} = \mathbf{e}\mathbf{H}^T$$

over every $\mathbf{e}$ that entails a single error (the code can only correct 1 erroneous bit):

error	syndrome
0000000	000
1000000	101
0100000	110
0010000	111
0001000	011
0000100	100
0000010	010
0000001	001

Example: $\mathbf{r} = [1100101]$

$$\mathbf{s} = \mathbf{r}\mathbf{H}^\top = [1100101] \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [110]$$

and hence $\mathbf{e} = [0100000]$ so that

$$\hat{\mathbf{c}} = \mathbf{r} + \mathbf{e} = \mathbf{r} = [1000101]\,.$$

Equivalent codes

Computing H from G

If the code is systematic, we have an easy way of computing the parity-check matrix...

...but what if it's not?

Equivalent codes

Computing H from G

If the code is systematic, we have an easy way of computing the parity-check matrix...

...but what if it's not? If the code is **not** systematic, one can apply operations on the generator matrix, $\mathbf{G}$, to try and transform it into that of an *equivalent* systematic code, $\mathbf{G}' = \begin{bmatrix} \mathbf{I}_k & \mathbf{P} \end{bmatrix}$.

Allowed operations are:

- On rows: replace any row with a linear combination of itself and other rows **or** swapping rows.
- On columns: swapping columns.

Equivalent codes

Computing H from G

If the code is systematic, we have an easy way of computing the parity-check matrix...

...but what if it's not? If the code is **not** systematic, one can apply operations on the generator matrix, $\mathbf{G}$, to try and transform it into that of an *equivalent* systematic code, $\mathbf{G}' = \begin{bmatrix} \mathbf{I}_k & \mathbf{P} \end{bmatrix}$.

Allowed operations are:

On rows: replace any row with a linear combination of itself and other rows **or** swapping rows.

On columns: swapping columns.

Definition: Equivalent codes

Two codes are equivalent if they have the same codewords (after, maybe, reordering the bits).

Index

Cyclic codes

Large values of k and n

Working with matrices is not efficient!!

Cyclic codes

Large values of k and n

Working with matrices is not efficient!!

Definition: Cyclic code

It is a linear block code in which any *circular* shift of a codeword results in another codeword.

In a cyclic code,

- If $[c_0, c_1, \ldots, c_{n-1}]$ is a codeword, then so is $[c_{n-1}, c_0, c_1, \ldots, c_{n-2}]$
  - i.e., every codeword is a (circularly) shifted version of another codeword.

Polynomial representation of codewords

Codeword $[c_0, c_1, \cdots, c_{n-1}]$ is represented as the polynomial

$$c(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1} x^{n-1}$$

How is

$$[c_0, c_1, \cdots, c_{n-1}] \rightarrow [c_{n-1}, c_0, \cdots, c_{n-2}]$$

achieved mathematically?

Polynomial representation of codewords

Codeword $[c_0, c_1, \cdots, c_{n-1}]$ is represented as the polynomial

$$c(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1} x^{n-1}$$

How is

$$[c_0, c_1, \cdots, c_{n-1}] \to [c_{n-1}, c_0, \cdots, c_{n-2}]$$

achieved mathematically? By multiplying $c(x)$ times x modulo $(x^n - 1)$, i.e.,

$$\begin{aligned} xc(x) &= c_0 x + c_1 x^2 + \cdots + c_{n-1} x^n = c_0 x + \cdots + c_{n-1} x^n + c_{n-1} - c_{n-1} \\ &= c_{n-1}(x^n - 1) + c_{n-1} + c_0 x + c_1 x^2 + \cdots + c_{n-2} x^{n-1} \end{aligned}$$

Hence,

$$(xc(x))_{x^n-1} = \underbrace{c_{n-1} + c_0 x + c_1 x^2 + \cdots + c_{n-2} x^{n-1}}_{[c_{n-1}, c_0, \cdots, c_{n-2}]}$$

Encoding

$\mathbf{G}$ generator matrix	$\rightarrow$	$g(x)$ generator polynomial

Encoding

$\mathbf{G}$ generator matrix	$\rightarrow$	$g(x)$ generator polynomial

Coding is carried out by multiplying, modulo $x^n - 1$, the polynomial representing $\mathbf{b}_i$ by a **generator polynomial**, $g(x)$,

$$c(x) = (b(x)g(x))_{x^n - 1}$$

Encoding

$\mathbf{G}$ generator matrix	$\rightarrow$	$g(x)$ generator polynomial

Coding is carried out by multiplying, modulo $x^n - 1$, the polynomial representing $\mathbf{b}_i$ by a **generator polynomial**, $g(x)$,

$$c(x) = (b(x)g(x))_{x^n - 1}$$

The generator polynomial, $g(x)$,

- it is of degree $r = n - k$,
- it must be an irreducible polynomial

Decoding

$\mathbf{H}$	$\rightarrow$	$h(x)$
parity-check matrix		parity-check polynomial

Decoding

H parity-check matrix	$\rightarrow$	$h(x)$ parity-check polynomial

The parity-check polynomial, $h(x)$,

- it is of degree $r' = n - k - 1$,
- must satisfy

$$(g(x)h(x))_{x^n-1} = 0.$$

Decoding

$\mathbf{H}$ parity-check matrix	$\rightarrow$	$h(x)$ parity-check polynomial

The parity-check polynomial, $h(x)$,

- it is of degree $r' = n - k - 1$,
- must satisfy

$$(g(x)h(x))_{x^n-1} = 0.$$

Just like in regular linear block codes, we can perform **syndrome decoding**,

$$s(x) = (r(x)h(x))_{x^n-1}$$