# Week 10: Error detection and error correction codes (Textbook Chapter 11)

Prof. Michael Gastpar

Slides by Prof. M. Gastpar and Prof. em. B. Rimoldi

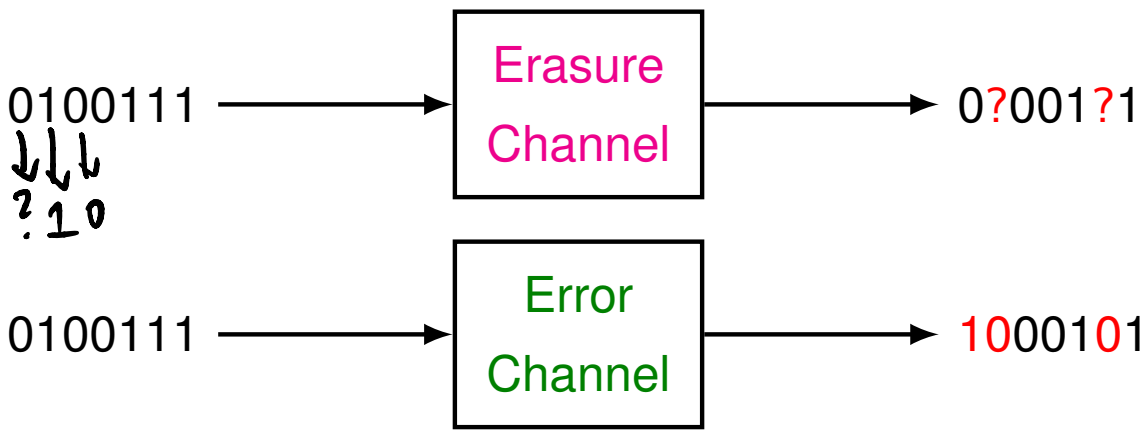
EPFL

Spring Semester 2025

# Outline

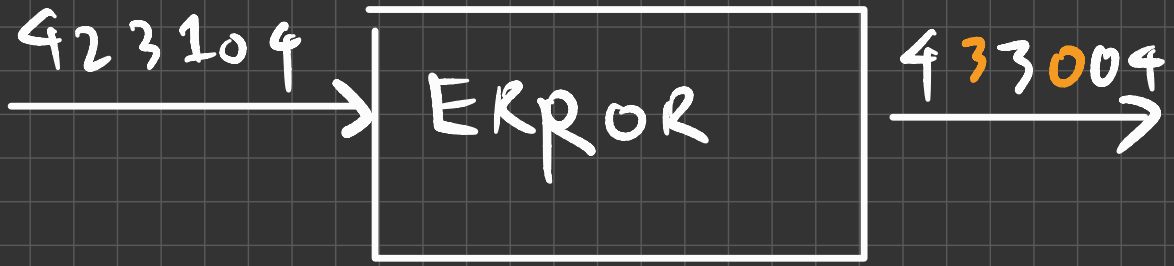
- The Internet often drops packets due to congestion.
- Not all the bits on a storage device can be retrieved.
- Wireless signals are very noisy.

We consider two types of channel models:

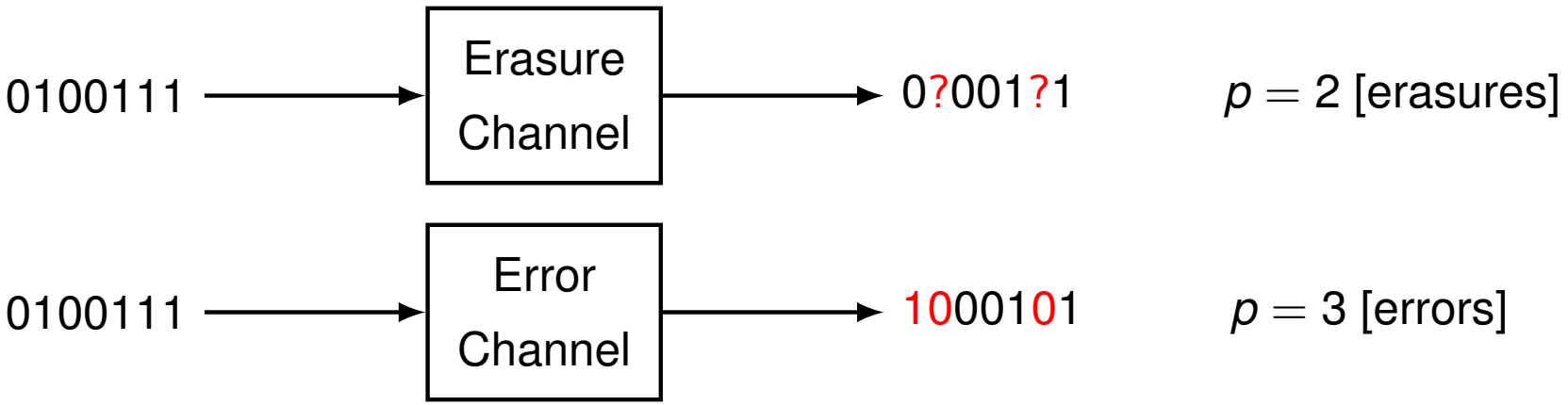



(The channel input alphabet is not necessarily binary.)


$A = \{0, 1, 2, 3, 4\}$



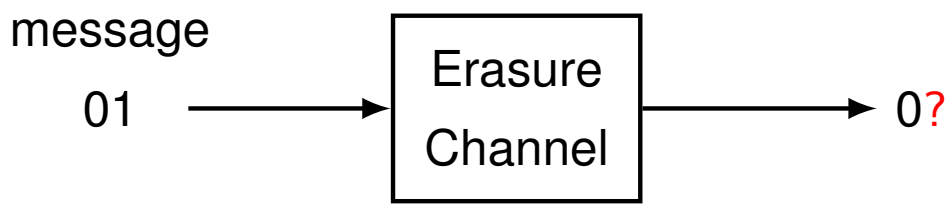
We define the **erasure weight** $p$ (resp. **error weight** $p$) as the total number of erasures (resp. errors).




Erasures are easier to deal with: they are essentially channel errors of known location.

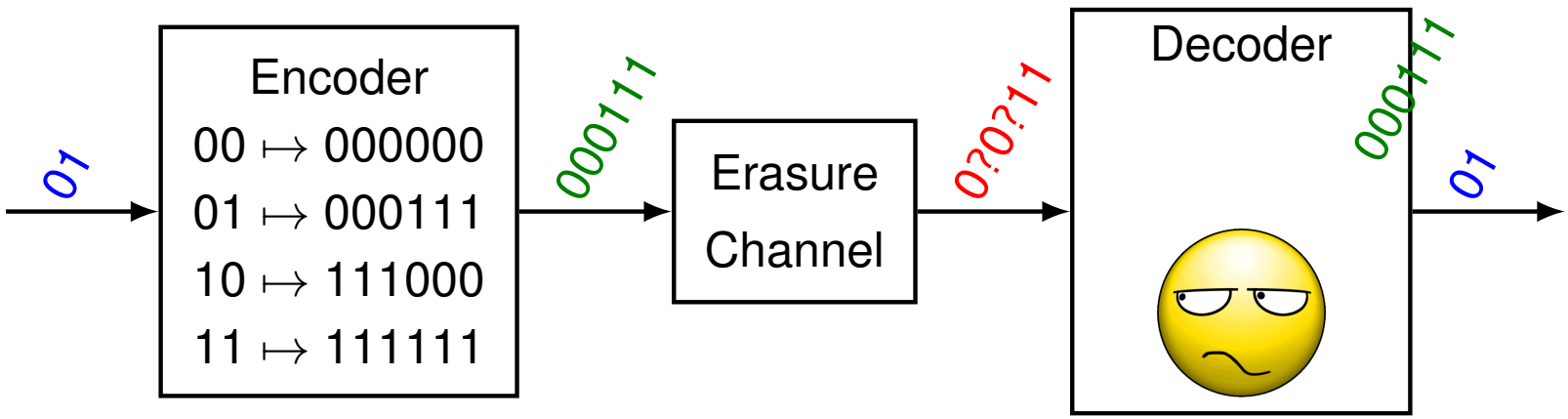
# CHANNEL CODING TO DEAL WITH ERASURES

Suppose that the source outputs 2 bits, and we store them as is (no channel coding):



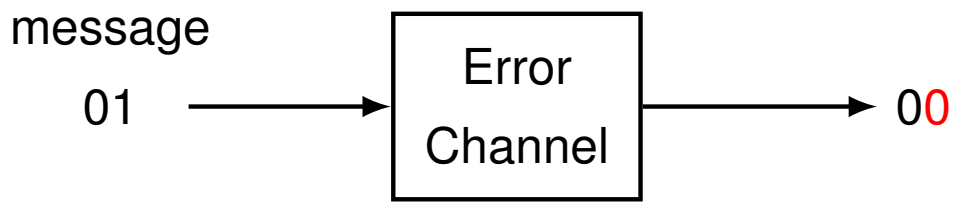

If any bit is erased, there is no way to determine the original message. (All hypotheses are equally valid.)

Now suppose that we do some channel coding:



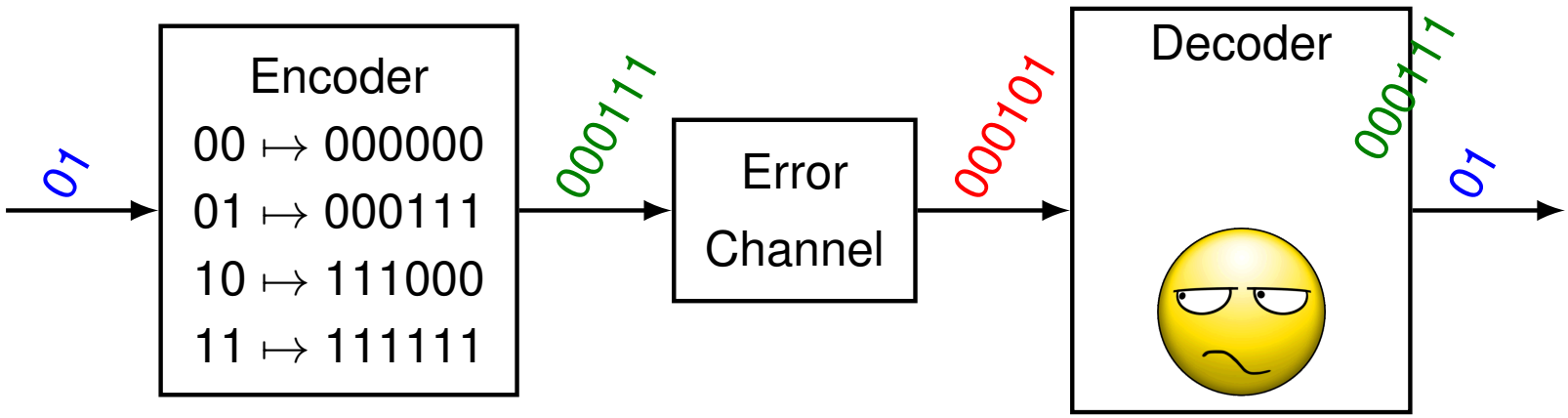

The decoder is able to fill the erased positions, because only one codeword is consistent with the observed channel output.

Suppose that the source outputs 2 bits, and we store them as is (no channel coding):




There is no way to tell that the channel flipped a bit.

Now suppose that we do channel coding:




The channel output is not a valid codeword. The decoder recognizes it, and assumes that the transmitted codeword is the one that agrees in most positions with the observed channel output.

# WE STUDY ONLY BLOCK CODES

The above is an $(n, k)$ **block code** with $n = 6$ and $k = 2$: each $k$ source symbols are substituted by $n$ channel symbols over the same alphabet.

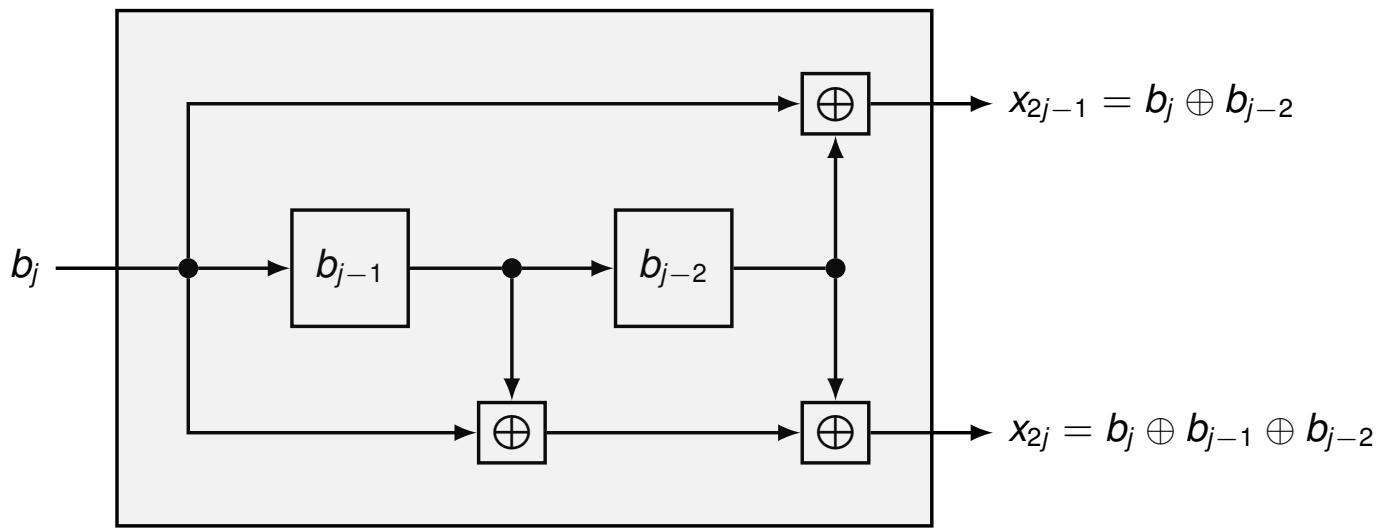
Since the alphabet is $\{0, 1\}$, we call it a **binary** $(n, k)$ block code.

We consider only block codes.

# EXAMPLE OF A NON-BLOCK CODE

The following is a convolutional encoder: every encoder input symbol produces two encoder output symbols.

The output pair produced at any given time is a linear function of the corresponding encoder input and encoder state (the previous two inputs).




(The name comes from linear system theory, done in your second year.)

# TERMINOLOGY

- The code $\mathcal{C}$ is the set of codewords.
- A codeword $c$ is an element of $\mathcal{A}^n$. (The alphabet $\mathcal{A}$ is $\{0, 1\}$ in our example).
- The block-length is $n$.
- Each codeword carries $k = \log_2 |\mathcal{C}|$ information bits. ($k = \log_2 8 = 3$ bits in our example.)
- The rate is $\frac{k}{n}$ bits per symbol.

*Encoder*

| $k = 3$ | | $n = 7$ |
|---|---|---|
| 000 | $\mapsto$ | 0000000 |
| 001 | $\mapsto$ | 0011100 |
| 010 | $\mapsto$ | 0111011 |
| 100 | $\mapsto$ | 1110100 |
| 011 | $\mapsto$ | 0100111 |
| 101 | $\mapsto$ | 1101000 |
| 110 | $\mapsto$ | 1001111 |
| 111 | $\mapsto$ | 1010011 |

The Hamming distance $d(x, y)$ between two $n$-tuples $x$ and $y$ is the number of positions in which they differ.

## Example (Hamming distance)

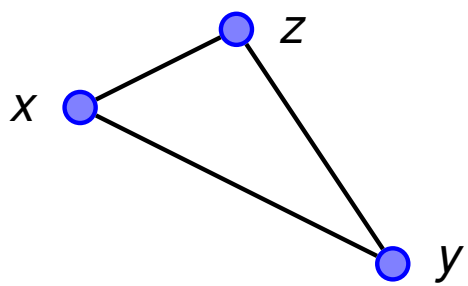
- $x = (101110)$, $y = (100110)$, $d(x, y) = 1$
- $x = (0427222)$, $y = (1227986)$, $d(x, y) = 5$
- $x = (0427222)$, $y = (0427222)$, $d(x, y) = 0$
- $x = (00)$, $y = (22)$, $d(x, y) = 2$

# The Hamming distance is indeed a distance

In math, a function of two variables is a **distance if it satisfies the following axioms:**

## Definition (distance axioms)

1. **non-negativity:** $d(x, y) \geq 0$ **with equality iff** $x = y$**.**
2. **symmetry:** $d(x, y) = d(y, x)$**.**
3. **triangle inequality:** $d(x, z) \leq d(x, y) + d(y, z)$**.**



$$d(x,y) = \sum_{i=1}^{n} d(x_i, y_i)$$

**CLAIM:** $d(x_i, z_i) \leq d(x_i, y_i) + d(y_i, z_i)$ (*)

**PROOF:** **CASE 1:** $x_i = z_i \Leftrightarrow d(x_i, z_i) = 0$

$\Rightarrow$ (*) HOLDS

**CASE 2:** $x_i \neq z_i \Leftrightarrow d(x_i, z_i) = 1$

AT LEAST ONE OF

$d(x_i, y_i) = 1$ OR

$d(y_i, z_i) = 1$

(OR BOTH)

$\Rightarrow$ (*) HOLDS.

$$d(x,y) = \sum_{i=1}^{n} d(x_i, y_i)$$

CLAIM: $d(x_i, z_i) \leq d(x_i, y_i) + d(y_i, z_i)$

EITHER: $d(x_i, z_i) = 0$ ✓

OR: $d(x_i, z_i) = 1$

$\Updownarrow$

$x_i \neq z_i \Rightarrow$ either $d(x_i, y_i) = 1$ ✓ or $d(y_i, z_i) = 1$ (or both)

**Proof:** We need to check that the triangle inequality holds.

Let $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_n)$, $z = (z_1, \ldots, z_n)$.

The Hamming distance is additive in the sense $d(x, z) = \sum_{i=1}^{n} \underbrace{d(x_i, z_i)}_{0 \text{ or } 1}$.

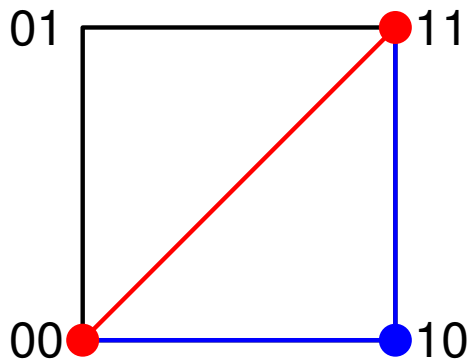
- if $d(x_i, z_i) = 0$, then $d(x_i, z_i) \leq d(x_i, y_i) + d(y_i, z_i)$.
- if $d(x_i, z_i) = 1$, then either $d(x_i, y_i) = 1$ or $d(y_i, z_i) = 1$ or both.
- Hence $d(x_i, z_i) \leq d(x_i, y_i) + d(y_i, z_i)$.
- By adding over all $i$,

$$d(x, z) = \sum_{i=1}^{n} d(x_i, z_i) \leq \sum_{i=1}^{n} \big(d(x_i, y_i) + d(y_i, z_i)\big) = d(x, y) + d(y, z).$$

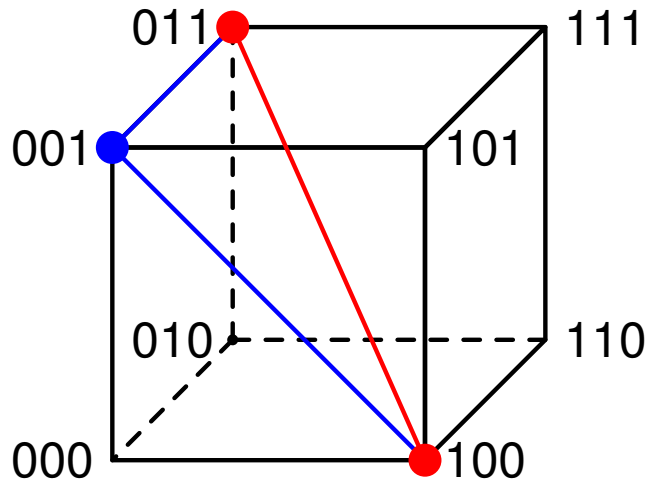
□

# Geometrical Interpretation

An $n$-length sequence of integers may be seen as an element of $\mathbb{R}^n$.




$d(00, 11) \leq d(00, 10) + d(10, 11)$

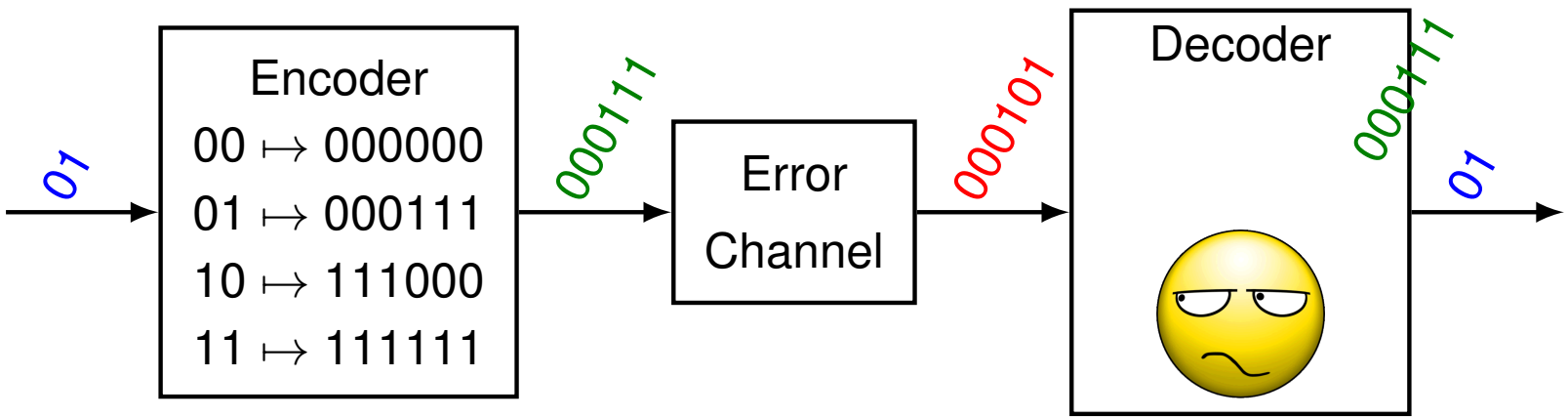



$d(011, 100) \leq d(011, 001) + d(001, 100)$

# MINIMUM-DISTANCE DECODER

$d_H(000101, 111111) = 4$

- The decoder guesses the encoder input based on the channel output.
- Here we consider only **minimum-distance decoders**.




$d_H(000101, 000000) = 2$

$d_H(000101, 000111) = 1$

$d_H(000101, 111000) = 5$

Let $y$ be the channel output observed by the decoder. A **minimum-distance decoder** decides that the channel input is (one of) the $\hat{c} \in \mathcal{C}$ for which $d(y, \hat{c})$ is minimized:

$$\hat{c} = \arg\min_{x \in \mathcal{C}} d(y, x)$$

$$\hat{c} \in \arg\min_{x \in \mathcal{C}} d(y, x)$$

The justification is that a small error weight is more likely than a large one.

## Example (minimum-distance decoder)

Let $y = (0110111)$ be the channel output.

The encoder decides that the channel input was $\hat{c} = (0100111)$.

| | *Encoder* | |
|---|---|---|
| $k = 3$ | | $n = 7$ |
| 000 | $\mapsto$ | 0000000 |
| 001 | $\mapsto$ | 0011100 |
| 010 | $\mapsto$ | 0111011 |
| 100 | $\mapsto$ | 1110100 |
| 011 | $\mapsto$ | 0100111 |
| 101 | $\mapsto$ | 1101000 |
| 110 | $\mapsto$ | 1001111 |
| 111 | $\mapsto$ | 1010011 |

## Definition (minimum distance)

The minimum distance of a code $\mathcal{C}$ is

$$d_{min}(\mathcal{C}) = \min_{x,y \in \mathcal{C}; x \neq y} d(x, y)$$

$c_0$ $c_1$ $c_2$ $c_3$

## Example

$\mathcal{C} = \{000000, 100110, 011001, 111111\} \Longrightarrow d_{min}(\mathcal{C}) = 3.$

$d(c_0, c_1) = 3$ $d(c_1, c_2) = 6$ $d(c_2, c_3) = 3$

$d(c_0, c_2) = 3$ $d(c_1, c_3) = 3$

$d(c_0, c_3) = 6$

# Code $\mathcal{C}$

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 |

$$n = 4$$

$$k = \log_2 |\mathcal{C}| = \log_2 16 = 4$$

$$\text{Rate} = \frac{k}{n} = \frac{4}{4} = 1.$$

$$d_{min} = 1.$$

# Code $\mathcal{C}$

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 |

$$n = 4$$

$$k = \log_2 |\mathcal{C}| = \log_2 8 = 3$$

$$\text{Rate} = \frac{k}{n} = \frac{3}{4}$$

$$d_{min} = 1.$$

# Code $\mathcal{C}$

0000
0011
1100
1111

$n = 4$

$k = \log_2 |\mathcal{C}| = \log_2 4 = 2$

$\text{Rate} = \frac{2}{4} = \frac{1}{2}$.

$d_{min} = 2$

Code $C$

0000

1111

$n = 4$

$k = \log_2 |C| = \log_2 2 = 1$

$\text{Rate} = \frac{k}{n} = \frac{1}{4}$

$d_{min} = 4$

# CODE $\mathcal{C}$

000
001
002
010
011
012
020
021
022
100
101
102
110
111
112
120
⋮
222

$n = 3$

$k = \log_2 |\mathcal{C}| = \log_2 27$

$\text{Rate} = \dfrac{\log_2 27}{3}$ $\dfrac{\text{bits}}{\text{channel use}}$

$k = \log_3 |\mathcal{C}| = \log_3 27 = 3$

ALPHABET SIZE

$\text{Rate} = \dfrac{k}{n} = \dfrac{3}{3} = 1$

$d_{min} = 1.$

# CODE $C$

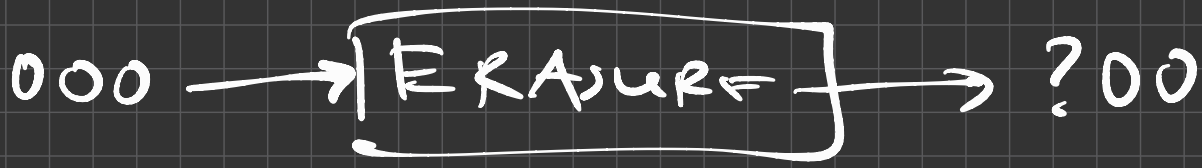
000
011
022
110
220
121
102
201
212

$n = 3$

$k = \log_3 |C| = \log_3 9 = 2$

$\text{Rate} = \frac{k}{n} = \frac{2}{3}$.

$d_{min} = 2$



# HAMMING DISTANCE:

1) $d(x,y) \geq 0$ w. eq. iff. $x = y$.

2) $d(x,y) = d(y,x)$

3) $d(x,z) \leq d(x,y) + d(y,z)$

# WHAT TO EXPECT FROM A DECODER

For an error channel:

(1) channel-error correction: the best is if the decoder recognizes and corrects the channel errors. In this case, the encoder input is recovered error-free.

(2) channel-error detection: in some circumstances, the encoder is able to detect the presence of channel errors but it is unable to correct them. The receiver may or may not ask for retransmission.

(3) decoding error: the worse is if the decoder tries to do as in (1) and makes the wrong decision.

For an erasure channel:

(1) erasure-correction: the best is if the decoder is capable of filling in the erased positions. In this case, the encoder input is recovered error-free.

(2) (erasure detection: unlike errors, erasures are always detected.)

(3) decoding error: the worse is if the decoder fills-in one or more erased positions with incorrect symbols.

# Error detection: how it relates to $d_{min}(\mathcal{C})$?

**Theorem (error detection: textbook theorem 11.2)**

1. Channel errors of weight $p < d_{min}(\mathcal{C})$ do not lead to a codeword. Hence they are detected.
2. Some channel errors of weight $p \geq d_{min}(\mathcal{C})$ do lead to another codeword. Hence they cannot be detected by a minimum-distance decoder.

Note: Erasures are always detected (by definition).

**Proof:**

1. Let $c \in \mathcal{C}$ be transmitted and $y$ be received.
   - If $p = d(c, y) < d_{min}(\mathcal{C})$, $y$ cannot be a codeword, therefore the error is detected.
2. We construct an example in which a channel error of weight $p = d_{min}(\mathcal{C})$ cannot be detected.
   - Let $c$ and $c'$ be codewords at distance $d_{min}(\mathcal{C})$.
   - Suppose that $c$ is the channel input and the channel output is $y = c'$.
   - $y$ is a codeword. A minimum-distance decoder will decide that no channel error has occurred.

□

## Example (Error Detection)

Let the encoding map be the MOD 97-10 procedure:

$$u \mapsto v = (100 \cdot u) + \left(98 - [100 \cdot u]_{97}\right)$$

Recall that $v$ is considered as valid if $[v]_{97} = 1$.

For example, $u = 0216936631 \mapsto v = 021693663165$.

Suppose $v$ is transmitted and $v'$ is received, $d(v, v') = 1$.

We can always write $v' = v + a10^k$ with $a \in \{-9, \ldots, -1, 1, \ldots, 9\}$.

The only way for $v'$ to be a valid codeword is if $\left[a10^k\right]_{97} = 0$.

Since $[10]_{97}$ is invertible, so is $[10^k]_{97}$, hence $a = 0$.

Therefore all weight 1 errors are detected, implying that the minimum distance is at least 2.

# ERASURE CORRECTION: HOW IT RELATES TO $d_{min}(\mathcal{C})$?

**THEOREM (ERASURE CORRECTION: TEXTBOOK THEOREM 11.3)**

A minimum-distance decoder for a code $\mathcal{C}$ **corrects** (fills in) **all the erasures** of weight $p$ iff $p < d_{min}(\mathcal{C})$.

## Proof

$\Leftarrow$: Suppose that $p < d_{min}(\mathcal{C})$.

Let $c$ and $y$ be the input and the output of an erasure channel, respectively, with $d(c, y) = p$.

We show that there is only one way to fill in the erased positions.

Let $c \in \mathcal{C}$ and $\tilde{c} \in \mathcal{C}$ be two codewords that agree with $y$ in the non-erased positions.

Clearly $d(c, \tilde{c}) \leq p < d_{min}(\mathcal{C})$. This is possible only if $c = \tilde{c}$.

□

## Proof

$\Rightarrow$: We use contraposition.

Suppose that $p = d_{min}(\mathcal{C})$.

We construct an example where the decoder will not always decode correctly.

Let $c$ and $c'$ be codewords at distance $d_{min}(\mathcal{C})$.

Let $c$ be the channel input, and suppose that the channel outputs the $y$ obtained by erasing the $p$ components of $c$ that differ from $c'$.

Notice that $d(c, y) = p = d(c', y)$.

If $c$ is a minimum-distance codeword, then so is $c'$.

□

# ERROR CORRECTION: HOW IT RELATES TO $d_{min}(\mathcal{C})$?

## THEOREM (ERROR CORRECTION: TEXTBOOK THEOREM 11.4)

A minimum-distance decoder for a code $\mathcal{C}$ **corrects all channel errors** of weight $p$ iff $p < \frac{d_{min}(\mathcal{C})}{2}$.

PROOF:

TRUE CODEWORD: $c$

NOISY CHANNEL OUTPUT: $y$

$$d(c,y) = p$$

LET $\hat{c}$ BE THE OUTPUT OF THE MINIMUM DISTANCE DECODER.

$$d(\hat{c},y) \leq p$$

THEN:

TRIANGLE!

$$d(c,\hat{c}) \leq \underbrace{d(c,y)}_{=p} + \underbrace{d(y,\hat{c})}_{\leq p}$$

$\leq 2p < d_{min}(\mathcal{C})$.

HENCE, $c = \hat{c}$.

HENCE, MIN. DIST. DECODER OUTPUTS THE CORRECT ANSWER.

□

**Proof of $\Leftarrow$:**

Let $c$ and $y$ be the input and the output of an error-channel, and suppose that $d(c, y) = p < \frac{d_{min}(\mathcal{C})}{2}$.

Let $\hat{c} \in \mathcal{C}$ be the guess made by a minimum-distance decoder that observes $y$.

We prove that $\hat{c} = c$.

$d(y, \hat{c}) \leq p$ because $d(y, c) = p$.

By the triangle inequality, $d(c, \hat{c}) \leq d(c, y) + d(y, \hat{c}) \leq 2p < d_{min}(\mathcal{C})$.

Hence $\hat{c} = c$. □

**Proof of $\Rightarrow$:** We use contraposition.

We have seen that if $p = d_{min}$, then the channel output can be a different codeword.

Hence it suffices to consider $d_{min} > p \geq \frac{d_{min}(\mathcal{C})}{2}$.

We construct an error pattern of weight $p$ that cannot be corrected.

Let $c$ and $c'$ be codewords at distance $d_{min}(\mathcal{C})$.

Let $y$ be obtained as follows: of the $d_{min}(\mathcal{C})$ positions where $c$ and $c'$ disagree, $p$ positions are chosen as in $c'$. All the remaining positions are chosen as in $c$. By construction,

$$d(c, y) = p$$
$$d(c', y) = d_{min}(\mathcal{C}) - p \leq 2p - p = p.$$

We see that $c'$ is at least as close to $y$ as $c$. □

# Detection/Correction Summary

| | detection guaranteed if | correction guaranteed if |
|---|---|---|
| erasure channel | (not applicable) | $p < d_{min}$ |
| error channel | $p < d_{min}$ | $p < \frac{d_{min}}{2}$ |

## Exercise

What is the minimum distance of code $\mathcal{C}$?

## Solution

$d_{min} = d(c_0, c_1) = 3$.

(Many other pairs $(c_i, c_j)$ satisfy $d(c_i, c_j) = 3$ as well.)

| code $\mathcal{C}$ |
|---|
| $c_0 = 0000000$ |
| $c_1 = 0011100$ |
| $c_2 = 0111011$ |
| $c_3 = 1110100$ |
| $c_4 = 0100111$ |
| $c_5 = 1101000$ |
| $c_6 = 1001111$ |
| $c_7 = 1010011$ |

Note: For notational convenience, we often write codewords without parenthesis or commas. For instance, 0000000 is a shorthand notation for $(0, 0, 0, 0, 0, 0, 0)$.

## Exercise (cont.)

How many erasures can $\mathcal{C}$ correct?

If $y_1 = ?01110?$, what was the transmitted codeword?

If $y_2 = 11???00$, what was the transmitted codeword?

If $y_3 = ???0011$, what was the transmitted codeword?

| code $\mathcal{C}$ |
|---|
| $c_0 = 0000000$ |
| $c_1 = 0011100$ |
| $c_2 = 0111011$ |
| $c_3 = 1110100$ |
| $c_4 = 0100111$ |
| $c_5 = 1101000$ |
| $c_6 = 1001111$ |
| $c_7 = 1010011$ |

## SOLUTION

$d_{min}(\mathcal{C}) = 3$, so the code can correct all erasures of weight $p < d_{min}(\mathcal{C}) = 3$.

For $y_1$, $p = 2 < d_{min}(\mathcal{C})$: correction is guaranteed. $y_1$ is decoded to $c_1$.

For $y_2$, $p = 3 \not< d_{min}(\mathcal{C})$: correction is not guaranteed in general. In fact, $y_2$ cannot be corrected by a minimum-distance decoder, because $c_3$ and $c_5$ are at the same distance from $y$.

For $y_3$, $p = 3 \not< d_{min}(\mathcal{C})$: correction is not guaranteed in general, but only one codeword is compatible with this $y$, namely $c_7$.

| code $\mathcal{C}$ |
|---|
| $c_0 = 0000000$ |
| $c_1 = 0011100$ |
| $c_2 = 0111011$ |
| $c_3 = 1110100$ |
| $c_4 = 0100111$ |
| $c_5 = 1101000$ |
| $c_6 = 1001111$ |
| $c_7 = 1010011$ |

## Exercise (cont.)

How many errors can $\mathcal{C}$ correct?

If $y_1 = c_1 + 0100000$, what codeword is decoded?

If $y_2 = c_4 + 0010100 = 0110011$, what codeword is decoded?

| code $\mathcal{C}$ |
|---|
| $c_0 = 0000000$ |
| $c_1 = 0011100$ |
| $c_2 = 0111011$ |
| $c_3 = 1110100$ |
| $c_4 = 0100111$ |
| $c_5 = 1101000$ |
| $c_6 = 1001111$ |
| $c_7 = 1010011$ |

## SOLUTION

$d_{min}(\mathcal{C}) = 3$, so the code can correct all errors of weight $p < \frac{d_{min}(\mathcal{C})}{2} = 1$.

$p(y_1) = 1 \quad \Rightarrow \arg\min_{\hat{c} \in \mathcal{C}} d(y_1, \hat{c}) = 0011100 = c_1.$

$p(y_2) = 2 \quad \Rightarrow \arg\min_{\hat{c} \in \mathcal{C}} d(y_2, \hat{c}) = c_2 \neq c_4.$

| code $\mathcal{C}$ |
|---|
| $c_0 = 0000000$ |
| $c_1 = 0011100$ |
| $c_2 = 0111011$ |
| $c_3 = 1110100$ |
| $c_4 = 0100111$ |
| $c_5 = 1101000$ |
| $c_6 = 1001111$ |
| $c_7 = 1010011$ |

## EXERCISE

Let $|\mathcal{C}| = M$. How many distances do we have to check to determine $d_{min}(\mathcal{C})$ via a brute-force approach?

## SOLUTION

Consider the following $M \times M$ matrix, with as many rows and columns as the number of codewords.

| | 1 | 2 | 3 | | $M-1$ | $M$ |
|---|---|---|---|---|---|---|
| 1 | $x$ | $\surd$ | $\surd$ | $\cdots$ | $\surd$ | $\surd$ |
| 2 | $x$ | $x$ | $\surd$ | $\cdots$ | $\surd$ | $\surd$ |
| | $\vdots$ | $\vdots$ | | | | $\vdots$ |
| $M-1$ | $x$ | $x$ | | $\cdots$ | $x$ | $\surd$ |
| $M$ | $x$ | $x$ | | $\cdots$ | $x$ | $x$ |

A "$\surd$" at position $(i, j)$ means that $c_i$ and $c_j$ need to be compared, whereas "$x$" means that they don't need to be compared.

There are $\frac{1}{2}(M^2 - M) = \frac{1}{2}M(M-1) = \binom{M}{2}$ of them.

- The code used in CDs has $M = 2^{1024} \approx 10^{300}$ codewords.
- A brute-force approach requires on the order of $10^{600}$ comparisons.
- (There are about $10^{50}$ atoms on Earth, about $10^{80}$ atoms in the universe, and about $5 \times 10^{29}$ picoseconds since the big bang.)
- We need codes that have structure, for which we can tell the minimum distance via analytical means, rather than by brute-force computation.
- First, we derive an upper bound to $d_{min}(\mathcal{C})$.

# UPPER BOUND TO $d_{min}(\mathcal{C})$

Recall the important parameters of a block code $\mathcal{C}$ over a $D$-ary alphabet:

- $n$, the block length.
- $k = \log_D |\mathcal{C}|$, the number of information symbols carried by a codeword. (Equivalently, $|\mathcal{C}| = D^k$.)
- $d_{min}$ is the minimum distance.

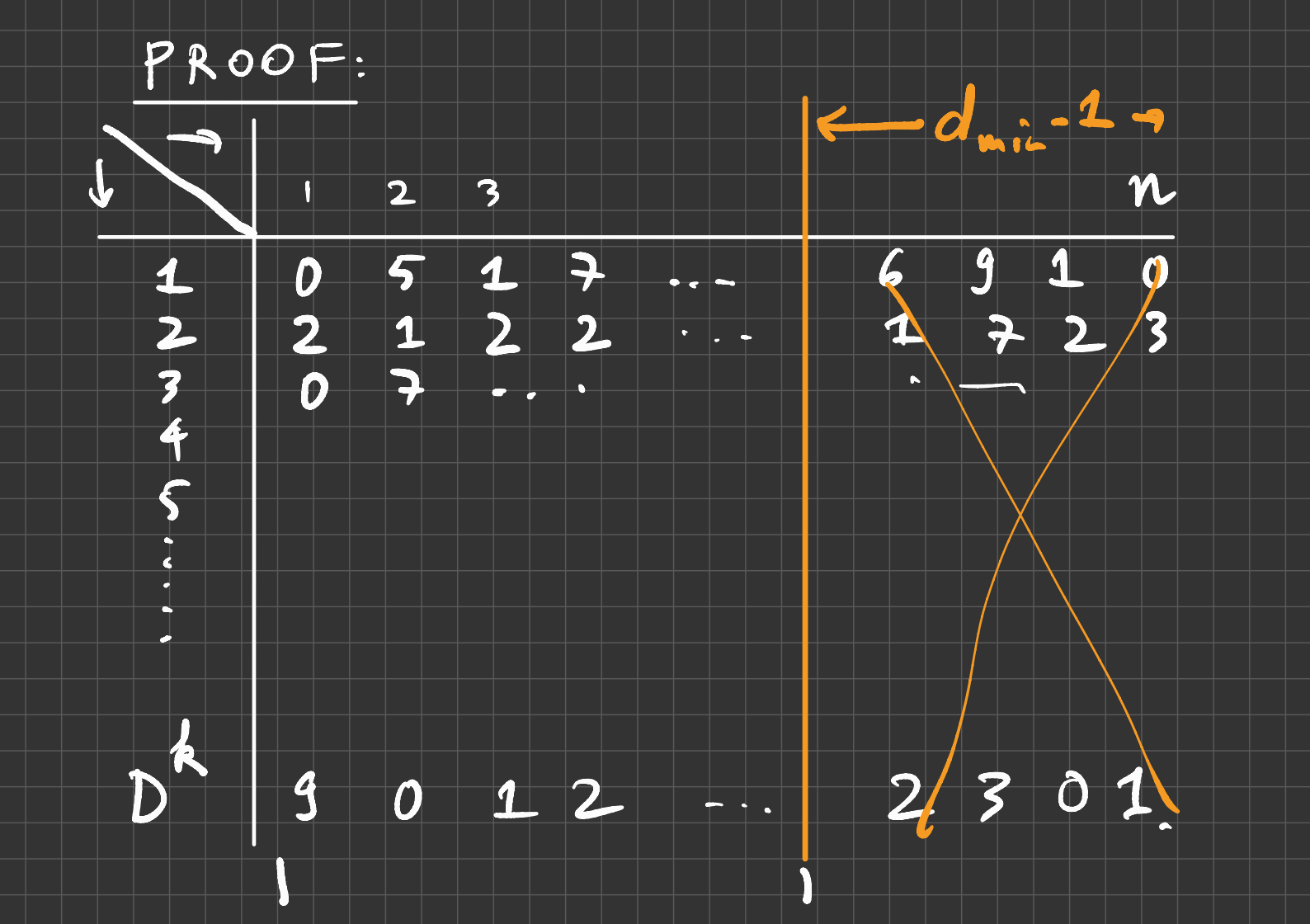
$D = |\mathcal{A}|$, $\mathcal{A}$ is the code alphabet.

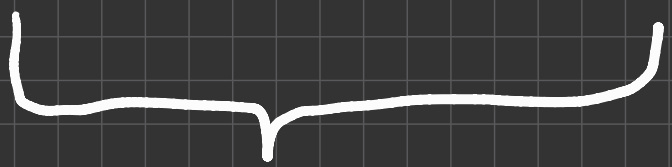
### Theorem (Singleton's bound: textbook theorem 11.5)

Regardless of the alphabet size, the minimum distance of a block code satisfies

$$d_{min} - 1 \leq n - k$$

Block codes that satisfy the Singleton bound with equality are called **Maximum Distance Separable** codes. (**MDS** codes.)


PROOF:
$d_{min}-1$
1 2 3 n
1 0 5 1 7 ...
2 2 1 2 2 ...
3 0 7 ...
4
5
6 9 1 0
1 7 2 3
$D^k$ 9 0 1 2 ...
2 3 0 1
1
1



NO TWO OF THESE STRINGS OF LENGTH $n-(d_{min}-1)$ CAN BE EQUAL!

OTHERWISE, THE CODE CANNOT HAVE MINIMUM DISTANCE $d_{min}$.

BUT HOW MANY DIFFERENT STRINGS IF LENGTH $n-(d_{min}-1)$ CAN WE MAKE?

$$\longrightarrow D^{n-(d_{min}-1)}$$

HENCE WE MUST HAVE:

$$D^k \leq D^{n-(d_{min}-1)}$$

$$\Updownarrow$$

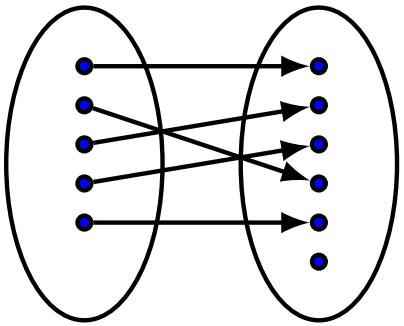
$$k \leq n - d_{min} + 1$$

D

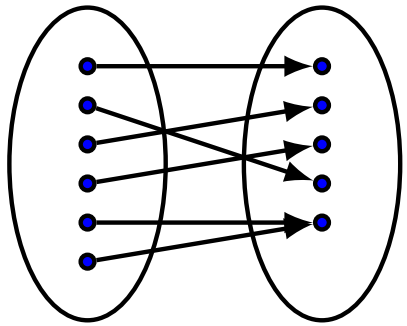
Recall that for a function $f : \mathcal{E} \to \mathcal{F}$

- $\mathcal{E}$ is the domain
- $\mathcal{F}$ is the codomain
- $f(\mathcal{E})$ is the image
- (range is sometimes used for the codomain, and sometimes for the image)

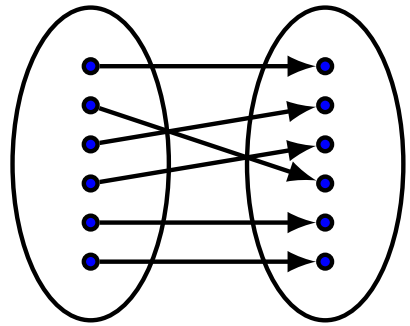
# PIGEONHOLE PRINCIPLE



injective
(one-to-one)



surjective
(onto)



bijective
(one-to-one and onto)

Let $f : \mathcal{E} \to \mathcal{F}$, where $\mathcal{E}$ and $\mathcal{F}$ are finite sets.

$$f \text{ injective} \Rightarrow |\mathcal{E}| \leq |\mathcal{F}|$$

$$f \text{ surjective} \Rightarrow |\mathcal{E}| \geq |\mathcal{F}|$$

$$f \text{ bijective} \Rightarrow |\mathcal{E}| = |\mathcal{F}|$$

## Proof of the Singleton Bound:

Consider the map $f : \mathcal{C} \to \mathcal{A}^{n-(d_{min}-1)}$ that removes the last $d_{min} - 1$ components of a codeword

$$f : (c_0, \ldots, c_{n-d_{min}}, \underbrace{c_{n-(d_{min}-1)}, \ldots, c_{n-1}}_{d_{min}-1 \text{ components}}) \mapsto (c_0, \ldots, c_{n-d_{min}})$$

The code has minimum distance $d_{min}$, so $f$ is injective (one-to-one).

By the pigeonhole principle, the cardinality of its domain cannot exceed the cardinality of the codomain:

$$\begin{aligned} |\mathcal{C}| &\leq |\mathcal{A}|^{n-(d_{min}-1)} \\ D^k &\leq D^{n-(d_{min}-1)} \\ k &\leq n - (d_{min} - 1). \end{aligned}$$

## Exercise

Write down $d_{min} - 1$ and $n - k$ for this code. Verify that Singleton's bound is satisfied.

| code $\mathcal{C}$ |
|---|
| $c_0 = 0000000$ |
| $c_1 = 0011100$ |
| $c_2 = 0111011$ |
| $c_3 = 1110100$ |
| $c_4 = 0100111$ |
| $c_5 = 1101000$ |
| $c_6 = 1001111$ |
| $c_7 = 1010011$ |

## Solution

$d_{min} - 1 = 2$.

$n - k = 7 - \log_2 8 = 4$.

Since $d_{min} - 1 \leq n - k$, Singleton's bound is satisfied.