# COMP2610/6261 - Information Theory

## Lecture 21: Computing Capacities, Coding in Practice, & Review


**Mark Reid** and Aditya Menon

Research School of Computer Science
The Australian National University


October 14, 2013

# Computing Capacities

Recall the definition of capacity for a channel $Q$ with inputs $\mathcal{A}_X$ and ouputs $\mathcal{A}_Y$

$$C = \max_{\mathbf{p}_X} I(X; Y)$$

How do we actually calculate this quantity?

# Computing Capacities

Recall the definition of capacity for a channel $Q$ with inputs $\mathcal{A}_X$ and ouputs $\mathcal{A}_Y$

$$C = \max_{\mathbf{p}_X} I(X;Y)$$

How do we actually calculate this quantity?

1. Compute the mutual information $I(X;Y)$ for a general $\mathbf{p}_X = (p_0, p_1)$
2. Determine which choice of $\mathbf{p}_X$ maximises $I(X;Y)$
3. Use that maximising value to determine $C$

# Computing Capacities

Recall the definition of capacity for a channel $Q$ with inputs $\mathcal{A}_X$ and ouputs $\mathcal{A}_Y$

$$C = \max_{\mathbf{p}_X} I(X; Y)$$

How do we actually calculate this quantity?

1. Compute the mutual information $I(X; Y)$ for a general $\mathbf{p}_X = (p_0, p_1)$
2. Determine which choice of $\mathbf{p}_X$ maximises $I(X; Y)$
3. Use that maximising value to determine $C$

**Binary Symmetric Channel**:
We first consider the *binary symmetric channel* with $\mathcal{A}_X = \mathcal{A}_Y = \{0, 1\}$ and flip probability $f$. It has transition matrix

$$Q = \begin{bmatrix} 1-f & f \\ f & 1-f \end{bmatrix}$$

The mutual information can be expressed as $I(X;Y) = H(Y) - H(Y|X)$. We therefore need to compute two terms: $H(Y)$ and $H(Y|X)$ so we need the distributions $P(y)$ and $P(y|x)$.

**Computing** $H(Y)$:

- $P(y=0) = (1-f)P(x=0) + fP(x=1) = (1-f)p_0 + fp_1$
- $P(y=1) = (1-f)P(x=1) + fP(x=0) = fp_0 + (1-f)p_1$

The mutual information can be expressed as $I(X;Y) = H(Y) - H(Y|X)$. We therefore need to compute two terms: $H(Y)$ and $H(Y|X)$ so we need the distributions $P(y)$ and $P(y|x)$.

**Computing** $H(Y)$:

- $P(y=0) = (1-f)P(x=0) + fP(x=1) = (1-f)p_0 + fp_1$
- $P(y=1) = (1-f)P(x=1) + fP(x=0) = fp_0 + (1-f)p_1$

In general, $\mathbf{q} := \mathbf{p}_Y = Q\mathbf{p}_X$, so above calculation is just

$$\mathbf{q} = \mathbf{p}_Y = \begin{bmatrix} (1-f) & f \\ f & (1-f) \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \end{bmatrix}$$

The mutual information can be expressed as $I(X;Y) = H(Y) - H(Y|X)$. We therefore need to compute two terms: $H(Y)$ and $H(Y|X)$ so we need the distributions $P(y)$ and $P(y|x)$.

**Computing** $H(Y)$:

- $P(y=0) = (1-f)P(x=0) + fP(x=1) = (1-f)p_0 + fp_1$
- $P(y=1) = (1-f)P(x=1) + fP(x=0) = fp_0 + (1-f)p_1$

In general, $\mathbf{q} := \mathbf{p}_Y = Q\mathbf{p}_X$, so above calculation is just

$$\mathbf{q} = \mathbf{p}_Y = \begin{bmatrix} (1-f) & f \\ f & (1-f) \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \end{bmatrix}$$

Using $H_2(q) = -q \log_2 q - (1-q)\log_2(1-q)$ and letting $q = q_1 = P(y=1)$ we see the entropy

$$H(Y) = H_2(q_1) = H_2(fp_0 + (1-f)p_1)$$

**Computing** $H(Y|X)$:

Since $P(y|x)$ is described by the matrix $Q$, we have

$H(Y|x=0) = H_2(P(y=1|x=0)) = H_2(Q_{1,0}) = H_2(f)$

and similarly,

$H(Y|x=1) = H_2(P(y=1|x=1)) = H_2(Q_{0,1}) = H_2(f)$

**Computing** $H(Y|X)$:
Since $P(y|x)$ is described by the matrix $Q$, we have
$H(Y|x=0) = H_2(P(y=1|x=0)) = H_2(Q_{1,0}) = H_2(f)$
and similarly,
$H(Y|x=1) = H_2(P(y=1|x=1)) = H_2(Q_{0,1}) = H_2(f)$
So,

$$H(Y|X) = \sum_x H(Y|x)P(x)$$

**Computing** $H(Y|X)$:
Since $P(y|x)$ is described by the matrix $Q$, we have
$H(Y|x=0) = H_2(P(y=1|x=0)) = H_2(Q_{1,0}) = H_2(f)$
and similarly,
$H(Y|x=1) = H_2(P(y=1|x=1)) = H_2(Q_{0,1}) = H_2(f)$
So,

$$H(Y|X) = \sum_x H(Y|x)P(x) = \sum_x H_2(f)P(x) = H_2(f)\sum_x P(x) = H_2(f)$$

**Computing** $I(X;Y)$:
Putting it all together gives

$$\boxed{I(X;Y) = H(Y) - H(Y|X) = H_2(fp_0 + (1-f)p_1) - H_2(f)}$$

Binary Symmetric Channel (BSC) with flip probability $f \in [0, 1]$:

$$I(X; Y) = H_2(fp_0 + (1 - f)p_1) - H_2(f)$$

Binary Symmetric Channel (BSC) with flip probability $f \in [0, 1]$:

$$I(X; Y) = H_2(fp_0 + (1 - f)p_1) - H_2(f)$$

**Examples**:

- BSC ($f = 0$) and $\mathbf{p}_X = (0.5, 0.5)$:
  $I(X; Y) = H_2(0.5) - H_2(0) = 1$

Binary Symmetric Channel (BSC) with flip probability $f \in [0, 1]$:

$$I(X; Y) = H_2(fp_0 + (1 - f)p_1) - H_2(f)$$

**Examples**:

- BSC ($f = 0$) and $\mathbf{p}_X = (0.5, 0.5)$:
  $I(X; Y) = H_2(0.5) - H_2(0) = 1$
- BSC ($f = 0.15$) and $\mathbf{p}_X = (0.5, 0.5)$:
  $I(X; Y) = H_2(0.5) - H_2(0.15) \approx 0.39$

Binary Symmetric Channel (BSC) with flip probability $f \in [0, 1]$:

$$I(X; Y) = H_2(fp_0 + (1 - f)p_1) - H_2(f)$$

**Examples**:

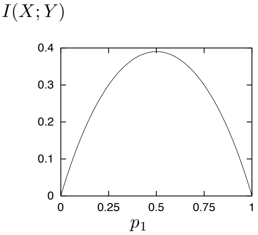
- BSC ($f = 0$) and $\mathbf{p}_X = (0.5, 0.5)$:
  $I(X; Y) = H_2(0.5) - H_2(0) = 1$
- BSC ($f = 0.15$) and $\mathbf{p}_X = (0.5, 0.5)$:
  $I(X; Y) = H_2(0.5) - H_2(0.15) \approx 0.39$
- BSC ($f = 0.15$) and $\mathbf{p}_X = (0.9, 0.1)$:
  $I(X; Y) = H_2(0.22) - H_2(0.15) \approx 0.15$

Binary Symmetric Channel (BSC) with flip probability $f \in [0,1]$:

$$I(X;Y) = H_2(fp_0 + (1-f)p_1) - H_2(f)$$

**Examples**:

- BSC ($f = 0$) and $\mathbf{p}_X = (0.5, 0.5)$:
  $I(X;Y) = H_2(0.5) - H_2(0) = 1$
- BSC ($f = 0.15$) and $\mathbf{p}_X = (0.5, 0.5)$:
  $I(X;Y) = H_2(0.5) - H_2(0.15) \approx 0.39$
- BSC ($f = 0.15$) and $\mathbf{p}_X = (0.9, 0.1)$:
  $I(X;Y) = H_2(0.22) - H_2(0.15) \approx 0.15$



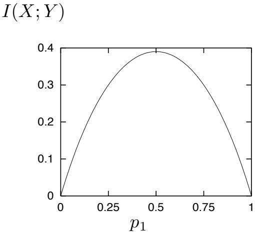

$I(X;Y)$ for $f = 0.15$

Binary Symmetric Channel (BSC) with flip probability $f \in [0, 1]$:

$$I(X;Y) = H_2(fp_0 + (1-f)p_1) - H_2(f)$$

**Examples**:

- BSC $(f = 0)$ and $\mathbf{p}_X = (0.5, 0.5)$:
  $I(X;Y) = H_2(0.5) - H_2(0) = 1$
- BSC $(f = 0.15)$ and $\mathbf{p}_X = (0.5, 0.5)$:
  $I(X;Y) = H_2(0.5) - H_2(0.15) \approx 0.39$
- BSC $(f = 0.15)$ and $\mathbf{p}_X = (0.9, 0.1)$:
  $I(X;Y) = H_2(0.22) - H_2(0.15) \approx 0.15$




$I(X;Y)$ for $f = 0.15$

**Maximise** $I(X;Y)$:
Since $I(X;Y)$ is symmetric in $p_1$ it is maximised when $p_0 = p_1 = 0.5$ in which case $C = 0.39$ for BSC with $f = 0.15$.

# Symmetry

## Symmetric Channel

A channel with input $\mathcal{A}_X$ and outputs $\mathcal{A}_Y$ and matrix $Q$ is **symmetric** if $\mathcal{A}_Y$ can be partitioned into subsets $Y' \subseteq Y$ so that each sub-matrix $Q'$ containing only rows for outputs $Y'$ has:

- Columns that are all permutations of each other
- Rows that are all permutations of each other

# Symmetry

## Symmetric Channel

A channel with input $\mathcal{A}_X$ and outputs $\mathcal{A}_Y$ and matrix $Q$ is **symmetric** if $\mathcal{A}_Y$ can be partitioned into subsets $Y' \subseteq Y$ so that each sub-matrix $Q'$ containing only rows for outputs $Y'$ has:

- Columns that are all permutations of each other
- Rows that are all permutations of each other

$\mathcal{A}_X = \mathcal{A}_Y = \{0, 1\}$

$$Q = \begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}$$

**Symmetric**

Subsets: $\{0, 1\}$

# Symmetry

## Symmetric Channel

A channel with input $\mathcal{A}_X$ and outputs $\mathcal{A}_Y$ and matrix $Q$ is **symmetric** if $\mathcal{A}_Y$ can be partitioned into subsets $Y' \subseteq Y$ so that each sub-matrix $Q'$ containing only rows for outputs $Y'$ has:

- Columns that are all permutations of each other
- Rows that are all permutations of each other

$\mathcal{A}_X = \mathcal{A}_Y = \{0, 1\}$

$$Q = \begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}$$

**Symmetric**

Subsets: $\{0, 1\}$

$\mathcal{A}_X = \{0, 1\}$, $\mathcal{A}_Y = \{0, ?, 1\}$

$$Q = \begin{bmatrix} 0.7 & 0.1 \\ 0.2 & 0.2 \\ 0.1 & 0.7 \end{bmatrix}$$

**Symmetric**

Subsets: $\{0, 1\}$, $\{?\}$

# Symmetry

## Symmetric Channel

A channel with input $\mathcal{A}_X$ and outputs $\mathcal{A}_Y$ and matrix $Q$ is **symmetric** if $\mathcal{A}_Y$ can be partitioned into subsets $Y' \subseteq Y$ so that each sub-matrix $Q'$ containing only rows for outputs $Y'$ has:

- Columns that are all permutations of each other
- Rows that are all permutations of each other

$\mathcal{A}_X = \mathcal{A}_Y = \{0,1\}$

$$Q = \begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}$$

**Symmetric**
Subsets: $\{0,1\}$

$\mathcal{A}_X = \{0,1\}$, $\mathcal{A}_Y = \{0,?,1\}$

$$Q = \begin{bmatrix} 0.7 & 0.1 \\ 0.2 & 0.2 \\ 0.1 & 0.7 \end{bmatrix}$$

**Symmetric**
Subsets: $\{0,1\}$, $\{?\}$

$\mathcal{A}_X = \mathcal{A}_Y = \{0,1\}$

$$Q = \begin{bmatrix} 0.9 & 0 \\ 0.1 & 1 \end{bmatrix}$$

**Not Symmetric**

# Symmetry

## Symmetric Channel

A channel with input $\mathcal{A}_X$ and outputs $\mathcal{A}_Y$ and matrix $Q$ is **symmetric** if $\mathcal{A}_Y$ can be partitioned into subsets $Y' \subseteq Y$ so that each sub-matrix $Q'$ containing only rows for outputs $Y'$ has:

- Columns that are all permutations of each other
- Rows that are all permutations of each other

$\mathcal{A}_X = \mathcal{A}_Y = \{0, 1\}$

$$Q = \begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}$$

**Symmetric**

Subsets: $\{0, 1\}$

$\mathcal{A}_X = \{0, 1\}$, $\mathcal{A}_Y = \{0, ?, 1\}$

$$Q = \begin{bmatrix} 0.7 & 0.1 \\ 0.2 & 0.2 \\ 0.1 & 0.7 \end{bmatrix}$$

**Symmetric**

Subsets: $\{0, 1\}$, $\{?\}$

$\mathcal{A}_X = \mathcal{A}_Y = \{0, 1\}$

$$Q = \begin{bmatrix} 0.9 & 0 \\ 0.1 & 1 \end{bmatrix}$$

**Not Symmetric**

# Symmetry

## Symmetric Channel

A channel with input $\mathcal{A}_X$ and outputs $\mathcal{A}_Y$ and matrix $Q$ is **symmetric** if $\mathcal{A}_Y$ can be partitioned into subsets $Y' \subseteq Y$ so that each sub-matrix $Q'$ containing only rows for outputs $Y'$ has:

- Columns that are all permutations of each other
- Rows that are all permutations of each other

$\mathcal{A}_X = \mathcal{A}_Y = \{0, 1\}$

$$Q = \begin{bmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{bmatrix}$$

**Symmetric**

Subsets: $\{0, 1\}$

$\mathcal{A}_X = \{0, 1\}$, $\mathcal{A}_Y = \{0, ?, 1\}$

$$Q = \begin{bmatrix} 0.7 & 0.1 \\ 0.2 & 0.2 \\ 0.1 & 0.7 \end{bmatrix}$$

**Symmetric**

Subsets: $\{0, 1\}$, $\{?\}$

$\mathcal{A}_X = \mathcal{A}_Y = \{0, 1\}$

$$Q = \begin{bmatrix} 0.9 & 0 \\ 0.1 & 1 \end{bmatrix}$$

**Not Symmetric**

(*Linear codes* achieve rates at the capacity of symmetric channels.)

# Computing Capacities in General

**Symmetric Channels**:

If the channel is symmetric, the maximising $\mathbf{p}_X$ – and thus the capacity – can be obtained via the uniform distribution over inputs (Exercise 10.10).

# Computing Capacities in General

**Symmetric Channels**:
If the channel is symmetric, the maximising $\mathbf{p}_X$ – and thus the capacity – can be obtained via the uniform distribution over inputs (Exercise 10.10).

**Non-Symmetric Channels**:
What can we do if the channel is not symmetric?

# Computing Capacities in General

**Symmetric Channels**:
If the channel is symmetric, the maximising $\mathbf{p}_X$ – and thus the capacity – can be obtained via the uniform distribution over inputs (Exercise 10.10).

**Non-Symmetric Channels**:
What can we do if the channel is not symmetric?

- We **can** still calculate $I(X; Y)$ for a general input distribution $\mathbf{p}_X$
- Finding the maximising $\mathbf{p}_X$ is more challenging

# Computing Capacities in General

**Symmetric Channels**:
If the channel is symmetric, the maximising $\mathbf{p}_X$ – and thus the capacity – can be obtained via the uniform distribution over inputs (Exercise 10.10).

**Non-Symmetric Channels**:
What can we do if the channel is not symmetric?

- We **can** still calculate $I(X;Y)$ for a general input distribution $\mathbf{p}_X$
- Finding the maximising $\mathbf{p}_X$ is more challenging

**Example** (Z Channel with $P(y=0|x=1)=f$):

$$\begin{aligned}
H(Y) &= H_2(P(y=1)) = H_2(0p_0 + (1-f)p_1) \\
&= H_2((1-f)p_1) \\
H(Y|X) &= p_0 H_2(P(y=1|x=0)) + p_1 H_2(P(y=0|x=1)) \\
&= p_0 \underbrace{H_2(0)}_{=0} + p_1 H_2(f) \\
I(X;Y) &= H_2((1-f)p_1) - p_1 H_2(f)
\end{aligned}$$

# Computing Capacities in General

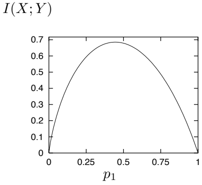
**Symmetric Channels**:
If the channel is symmetric, the maximising $\mathbf{p}_X$ – and thus the capacity – can be obtained via the uniform distribution over inputs (Exercise 10.10).

**Non-Symmetric Channels**:
What can we do if the channel is not symmetric?

- We **can** still calculate $I(X;Y)$ for a general input distribution $\mathbf{p}_X$
- Finding the maximising $\mathbf{p}_X$ is more challenging

**Example** (Z Channel with $P(y=0|x=1)=f$):

$$\begin{aligned}
H(Y) &= H_2(P(y=1)) = H_2(0p_0 + (1-f)p_1) \\
&= H_2((1-f)p_1) \\
H(Y|X) &= p_0 H_2(P(y=1|x=0)) + p_1 H_2(P(y=0|x=1)) \\
&= p_0 \underbrace{H_2(0)}_{=0} + p_1 H_2(f) \\
I(X;Y) &= H_2((1-f)p_1) - p_1 H_2(f)
\end{aligned}$$




$I(X;Y)$ for Z channel with $f = 0.15$

# Computing Capacities in General

What to do once we know $I(X;Y)$?

- $I(X;Y)$ is *concave* in $\mathbf{p}_X \implies$ single maximum
- For *binary* inputs, just look for stationary points (not for $|\mathcal{A}_X| > 2$) i.e., where $\frac{d}{dp} I(X;Y) = 0$ for $\mathbf{p}_X = (1-p, p)$

# Computing Capacities in General

What to do once we know $I(X;Y)$?

- $I(X;Y)$ is *concave* in $\mathbf{p}_X \implies$ single maximum
- For *binary* inputs, just look for stationary points (not for $|\mathcal{A}_X| > 2$) i.e., where $\frac{d}{dp}I(X;Y) = 0$ for $\mathbf{p}_X = (1-p, p)$

**Example** (Z Channel):
Showed earlier that $I(X;Y) = H_2((1-f)p) - pH_2(f)$ so solve

$$\begin{aligned}\frac{d}{dp}I(X;Y) = 0 &\iff (1-f)\log_2\left(\frac{1-(1-f)p}{(1-f)p}\right) - H_2(f) = 0 \\ &\iff \frac{1-(1-f)p}{(1-f)p} = 2^{H_2(f)/(1-f)} \\ &\iff p = \frac{1/(1-f)}{1+2^{H_2(f)/(1-f)}}\end{aligned}$$

For $f = 0.15$, we get $p = \frac{1/0.85}{1+2^{0.61/0.85}} \approx 0.44$ and so $C = H_2(0.38) - 0.44H_2(0.15) \approx 0.685$

# Computing Capacities in General

What to do once we know $I(X;Y)$?

- $I(X;Y)$ is *concave* in $\mathbf{p}_X \implies$ single maximum
- For *binary* inputs, just look for stationary points (not for $|\mathcal{A}_X| > 2$) i.e., where $\frac{d}{dp} I(X;Y) = 0$ for $\mathbf{p}_X = (1-p, p)$

**Example** (Z Channel):
Showed earlier that $I(X;Y) = H_2((1-f)p) - pH_2(f)$ so solve

$$\begin{aligned}
\frac{d}{dp} I(X;Y) = 0 &\iff (1-f) \log_2 \left( \frac{1-(1-f)p}{(1-f)p} \right) - H_2(f) = 0 \\
&\iff \frac{1-(1-f)p}{(1-f)p} = 2^{H_2(f)/(1-f)} \\
&\iff p = \frac{1/(1-f)}{1 + 2^{H_2(f)/(1-f)}}
\end{aligned}$$

For $f = 0.15$, we get $p = \frac{1/0.85}{1+2^{0.61/0.85}} \approx 0.44$ and so $C = H_2(0.38) - 0.44 H_2(0.15) \approx 0.685$

**Homework**: Show that $\frac{d}{dp} H_2(p) = \log_2 \frac{1-p}{p}$

# Theory and Practice

*The difference between theory and practice is that, in theory, there is no difference between theory and practice but, in practice, there is.*

— *Jan L. A. van de Snepscheut*

# Theory and Practice

*The difference between theory and practice is that, in theory, there is no difference between theory and practice but, in practice, there is.*

— *Jan L. A. van de Snepscheut*

**Theory vs. Practice**

- The NCCT theorem tells us that good block codes exist for any noisy channel (in fact, most random codes are good)
- However, the theorem is non-constructive: it does not tell us **how** to create *practical* codes for a given noisy channel
- The construction of practical codes that achieve rates up to the capacity for general channels is ongoing research

# Types of Codes

When we talk about types of codes we will be referring to schemes for creating $(N, K)$ codes for any size $N$. MacKay makes the following distinctions:

- **Very Good**: Can achieve arbitrarily small error at any rate up to the channel capacity (i.e., for any $\epsilon > 0$ a very good coding scheme can make a code with $K/N = C$ and $p_{BM} < \epsilon$)

# Types of Codes

When we talk about types of codes we will be referring to schemes for creating $(N, K)$ codes for any size $N$. MacKay makes the following distinctions:

- **Very Good**: Can achieve arbitrarily small error at any rate up to the channel capacity (i.e., for any $\epsilon > 0$ a very good coding scheme can make a code with $K/N = C$ and $p_{BM} < \epsilon$)
- **Good**: Can achieve arbitrarily small error up to some maximum rate strictly less than the channel capacity (i.e, for any $\epsilon$ a good coding scheme can make a code with $K/N = R_{max} < C$ and $p_{BM} < \epsilon$)

# Types of Codes

When we talk about types of codes we will be referring to schemes for creating $(N, K)$ codes for any size $N$. MacKay makes the following distinctions:

- **Very Good**: Can achieve arbitrarily small error at any rate up to the channel capacity (i.e., for any $\epsilon > 0$ a very good coding scheme can make a code with $K/N = C$ and $p_{BM} < \epsilon$)
- **Good**: Can achieve arbitrarily small error up to some maximum rate strictly less than the channel capacity (i.e, for any $\epsilon$ a good coding scheme can make a code with $K/N = R_{max} < C$ and $p_{BM} < \epsilon$)
- **Bad**: Cannot achieve arbitrarily small error, or only achieve it if the rate goes to zero (i.e., either $p_{BM} \to a > 0$ as $N \to \infty$ or $p_{BM} \to 0 \implies K/N \to 0$)

# Types of Codes

When we talk about types of codes we will be referring to schemes for creating $(N, K)$ codes for any size $N$. MacKay makes the following distinctions:

- **Very Good**: Can achieve arbitrarily small error at any rate up to the channel capacity (i.e., for any $\epsilon > 0$ a very good coding scheme can make a code with $K/N = C$ and $p_{BM} < \epsilon$)
- **Good**: Can achieve arbitrarily small error up to some maximum rate strictly less than the channel capacity (i.e, for any $\epsilon$ a good coding scheme can make a code with $K/N = R_{max} < C$ and $p_{BM} < \epsilon$)
- **Bad**: Cannot achieve arbitrarily small error, or only achieve it if the rate goes to zero (i.e., either $p_{BM} \to a > 0$ as $N \to \infty$ or $p_{BM} \to 0 \implies K/N \to 0$)
- **Practical**: Can be coded and decoded in time that is polynomial in the block length $N$.

# Random Codes

During the discussion of the Noisy-Channel Coding Theorem we saw how to construct very good **random codes** via expurgation and typical set decoding.

**Properties**:

- Very Good: Rates up to $C$ are achievable with arbitrarily small error
- Construction is easy
- Not Practical:
  - The $2^K$ codewords have no structure and must be "memorised"
  - Typical set decoding is expensive

# Linear Codes

## $(N, K)$ Block Code

An $(N, K)$ **block code** is a list of $S = 2^K$ codewords $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(S)}\}$, each of length $N$. A signal $s \in \{1, 2, \dots, 2^K\}$ is encoded as $\mathbf{x}^{(s)}$.

# Linear Codes

## $(N, K)$ Block Code

An $(N, K)$ **block code** is a list of $S = 2^K$ codewords $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(S)}\}$, each of length $N$. A signal $s \in \{1, 2, \dots, 2^K\}$ is encoded as $\mathbf{x}^{(s)}$.

## Linear $(N, K)$ Block Code

A **linear** $(N, K)$ **block code** is an $(N, K)$ block code where $s$ is first represented as a $K$-bit binary vector $\mathbf{s} \in \{0, 1\}^K$ and then encoded via multiplication by an $N \times K$ binary matrix $\mathbf{G}^\top$ to form $\mathbf{t} = \mathbf{G}^\top \mathbf{s}$ modulo 2.

Here linear means all $S = 2^K$ messages can be obtained by “adding” different combinations of the $K$ codewords $\mathbf{t}_i = \mathbf{G}^\top \mathbf{e}_i$ where $\mathbf{e}_i$ is $K$-bit string with single 1 in position $i$.

# Linear Codes

## $(N, K)$ Block Code

An $(N, K)$ **block code** is a list of $S = 2^K$ codewords $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(S)}\}$, each of length $N$. A signal $s \in \{1, 2, \dots, 2^K\}$ is encoded as $\mathbf{x}^{(s)}$.

## Linear $(N, K)$ Block Code

A **linear** $(N, K)$ **block code** is an $(N, K)$ block code where $s$ is first represented as a $K$-bit binary vector $\mathbf{s} \in \{0, 1\}^K$ and then encoded via multiplication by an $N \times K$ binary matrix $\mathbf{G}^\top$ to form $\mathbf{t} = \mathbf{G}^\top \mathbf{s}$ modulo 2.

Here linear means all $S = 2^K$ messages can be obtained by "adding" different combinations of the $K$ codewords $\mathbf{t}_i = \mathbf{G}^\top \mathbf{e}_i$ where $\mathbf{e}_i$ is $K$-bit string with single 1 in position $i$.

**Example**: Suppose $(N, K) = (7, 4)$. To send $s = 3$, first create $\mathbf{s} = 0011$ and send $\mathbf{t} = \mathbf{G}^\top \mathbf{s} = \mathbf{G}^\top (\mathbf{e}_0 + \mathbf{e}_1) = \mathbf{G}^\top \mathbf{e}_0 + \mathbf{G}^\top \mathbf{e}_1 = \mathbf{t}_0 + \mathbf{t}_1$ where $\mathbf{e}_0 = 0001$ and $\mathbf{e}_1 = 0010$.

# Linear Codes: Examples

**(7,4) Hamming Code**

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

For $\mathbf{s} = 0011$,

$$\mathbf{G}^\top \mathbf{s} (\bmod 2) = [0\ 0\ 1\ 1\ 1\ 0\ 0]^\top$$

**(6,3) Repetition Code**

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

For $\mathbf{s} = 010$,

$$\mathbf{G}^\top \mathbf{s} (\bmod 2) = [0\ 1\ 0\ 0\ 1\ 0]^\top$$

# Decoding

We can construct codes with a relatively simple encoding but how do we decode them? That is, given the input distribution and channel model $Q$ how do we find the posterior distribution over $\mathbf{x}$ given we received $\mathbf{y}$?

# Decoding

We can construct codes with a relatively simple encoding but how do we decode them? That is, given the input distribution and channel model $Q$ how do we find the posterior distribution over $\mathbf{x}$ given we received $\mathbf{y}$?

Simple! Just compute

$$P(\mathbf{x}|\mathbf{y}) = \frac{P(\mathbf{y}|\mathbf{x})}{\sum_{\mathbf{x}' \in \mathcal{C}} P(\mathbf{y}|\mathbf{x})P(\mathbf{x})}$$

**But**:

- the number of codes $\mathbf{x} \in \mathcal{C}$ is $2^K$ so, naively, the sum is expensive
- linear codes provide structure that the above method doesn't exploit

# Types of Linear Code

Many commonly used codes are linear:

- Repetition Codes: e.g., $0 \to 000$ ; $1 \to 111$
- Convolution Codes: Linear coding plus bit shifts
- Concatenation Codes: Two or more levels of error correction
- Hamming Codes: Parity checking
- Low-Density Parity-Check Codes: Semi-random construction

# Types of Linear Code

Many commonly used codes are linear:

- Repetition Codes: e.g., $0 \to 000$ ; $1 \to 111$
- Convolution Codes: Linear coding plus bit shifts
- Concatenation Codes: Two or more levels of error correction
- Hamming Codes: Parity checking
- Low-Density Parity-Check Codes: Semi-random construction

An NCCT can be proved for linear codes (i.e., "there exists a linear code" replacing "there exists a code") but the proof is still non-constructive.

# Types of Linear Code

Many commonly used codes are linear:

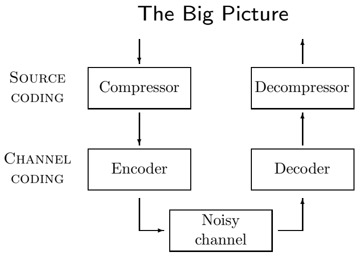
- Repetition Codes: e.g., $0 \rightarrow 000$ ; $1 \rightarrow 111$
- Convolution Codes: Linear coding plus bit shifts
- Concatenation Codes: Two or more levels of error correction
- Hamming Codes: Parity checking
- Low-Density Parity-Check Codes: Semi-random construction

An NCCT can be proved for linear codes (i.e., "there exists a linear code" replacing "there exists a code") but the proof is still non-constructive.

Practical linear codes:

- Use very large block sizes $N$
- Based on semi-random code constructions
- Apply probabilistic decoding techniques
- Used in wireless and satellite communication

# Coding: Review




**Source Coding for Compression**

- Shrink sequences
- Identify and remove redundancy
- Size limited by entropy
- Source Coding Theorems (Block & Variable Length)

**Channel Coding for Reliability**

- Protect sequences
- Add known form of redundancy
- Rate limited by capacity
- Noisy-Channel Coding Theorem

# Why "Entropy"?

# Why "Entropy"?

*You should call it entropy... no one really knows what entropy really is, so in a debate you will always have the advantage.*

— J. von Neumann to C. Shannon

# Why “Entropy”?

*You should call it entropy... no one really knows what entropy really is, so in a debate you will always have the advantage.*

— J. von Neumann to C. Shannon

**Thanks!**