Information Theory

Lecture 2: Inequalities & Other Results

Mark Reid

Research School of Computer Science
The Australian National University

1st December, 2014

Outline

1. Inequalities
   - Probabilisitic Inequalities
   - Convex Inequalities
   - Information Theoretic Inequalities
2. Key Results
   - The Source Coding Theorem for Lossy Uniform-Length Coding
   - The Source Coding Theorem for Lossless Variable-Length Coding
   - The Noisy-Channel Coding Theorem

Expectation and Variance

Let X be a random variable over $\mathcal{X}$, with probability distribution p

Expected value:

$$\mathbb{E}[X] = \sum_{x \in \mathcal{X}} x \cdot p(x).$$

Variance:

$$\begin{aligned} \mathbb{V}[X] &= \mathbb{E}[(X - \mathbb{E}[X])^2] \\ &= \mathbb{E}[X^2] - (\mathbb{E}[X])^2. \end{aligned}$$

Standard deviation is $\sqrt{\mathbb{V}[X]}$

Properties of expectation and variance

Expectation: A key property of expectations is linearity:

$$\mathbb{E}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbb{E}\left[X_i\right].$$

This holds even if the variables are dependent!

Variance: We have linearity of variance for independent random variables:

$$\mathbb{V}\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} \mathbb{V}\left[X_i\right].$$

Does not hold if the variables are dependent

Also, for any $a \in \mathbb{R}$ we have $\mathbb{E}[aX] = a \cdot \mathbb{E}[X]$ and $\mathbb{V}[aX] = a^2 \cdot \mathbb{V}[X]$.

Markov's Inequality

Theorem

Let X be a nonnegative random variable. Then, for any $\lambda > 0$,

$$p(X \geq \lambda \cdot \mathbb{E}[X]) \leq \frac{1}{\lambda}.$$

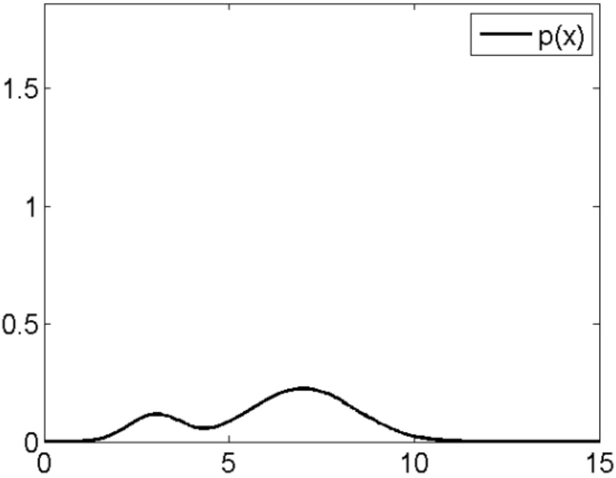
Values from nonnegative r.v. unlikely to be much larger than expectation

Proof: Let $\alpha = \lambda \mathbb{E}[X]$.

$$\begin{aligned}
\mathbb{E}[X] &= \sum_{x \in \mathcal{X}} x \cdot p(x) \\
&= \sum_{x < \alpha} x \cdot p(x) + \sum_{x \geq \alpha} x \cdot p(x) \\
&\geq \sum_{x \geq \alpha} x \cdot p(x) \quad \text{nonneg. of random variable} \\
&\geq \sum_{x \geq \alpha} \alpha \cdot p(x) = \alpha \cdot p(X \geq \alpha)
\end{aligned}$$

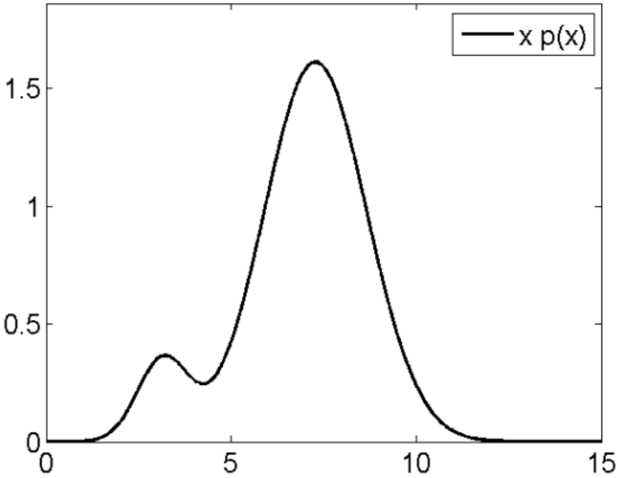
Markov's Inequality

Illustration from `http://justindomke.wordpress.com/`



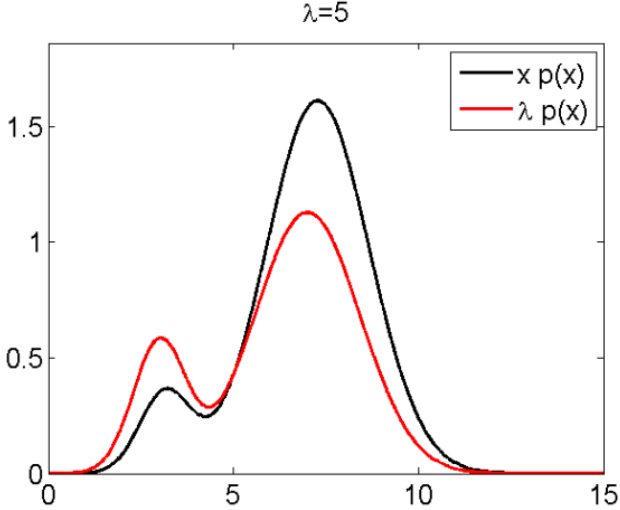
Markov's Inequality

Illustration from `http://justindomke.wordpress.com/`



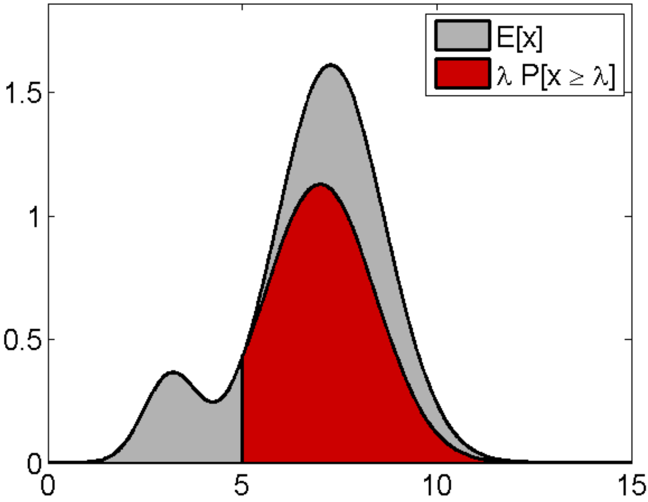
Markov's Inequality

Illustration from `http://justindomke.wordpress.com/`



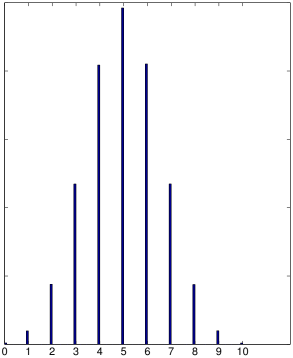
Markov's Inequality

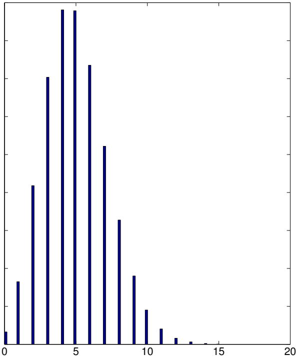
Illustration from `http://justindomke.wordpress.com/`



Chebyshev's Inequality

Markov's inequality only uses the mean of the distribution What about the spread of the distribution (variance)?





Chebyshev's Inequality

Theorem

Let X be a random variable with $\mathbb{E}[X] < \infty$. Then, for any $\lambda > 0$,

$$p(|X - \mathbb{E}[X]| \geq \lambda) \leq \frac{\mathbb{V}[X]}{\lambda^2}.$$

Bounds probability of "unexpected" outcome in terms of variance.

Note: Does not require non negativity; two-sided bound.

Corollary

Let X be a random variable with $\mathbb{E}[X] < \infty$. Then, for any $\lambda > 0$,

$$p(|X - \mathbb{E}[X]| \geq \lambda \cdot \sqrt{\mathbb{V}[X]}) \leq \frac{1}{\lambda^2}.$$

Observations unlikely several standard deviations away from the mean.

Chebyshev's Inequality
Proof

Define

$$Y = (X - \mathbb{E}[X])^2.$$

Then, by Markov's inequality, for any $\nu > 0$,

$$p(Y \geq \nu) \leq \frac{\mathbb{E}[Y]}{\nu}.$$

But,

$$\mathbb{E}[Y] = \mathbb{V}[X].$$

Also,

$$Y \geq \nu \iff |X - \mathbb{E}[X]| \geq \sqrt{\nu}.$$

Thus, setting $\lambda = \sqrt{\nu}$,

$$p(|X - \mathbb{E}[X]| \geq \lambda) \leq \frac{\mathbb{V}[X]}{\lambda^2}.$$

Law of Large Numbers

Theorem

Let $X_1, \ldots, X_n$ be a sequence of iid random variables, with

$$\mathbb{E}[X_i] = \mu$$

and $\mathbb{V}[X_i] < \infty$. Define

$$\bar{X}_n = \frac{X_1 + \ldots + X_n}{n}.$$

Then, for any $\epsilon > 0$,

$$\lim_{n \to \infty} p(|\bar{X}_n - \mu| > \epsilon) = 0.$$

Given enough trials, the empirical "success frequency" will be close to the expected value

Since X_i's are identically distributed,

$$\mathbb{E}[\bar{X}_n] = \mu.$$

Since the X_i's are independent,

$$\begin{aligned} \mathbb{V}[\bar{X}_n] &= \mathbb{V}\left[\frac{X_1 + \ldots + X_n}{n}\right] \\ &= \frac{\mathbb{V}\left[X_1 + \ldots + X_n\right]}{n^2} \\ &= \frac{n\sigma^2}{n^2} \\ &= \frac{\sigma^2}{n}. \end{aligned}$$

Applying Chebyshev's inequality to $\bar{X}_n$,

$$p(|\bar{X}_n - \mu| \geq \epsilon) \leq \frac{\mathbb{V}[\bar{X}_n]}{\epsilon^2} = \frac{\sigma^2}{n\epsilon^2}.$$

As $n \to \infty$, the right hand side $\to 0$.

Thus,

$$p(|\bar{X}_n - \mu| < \epsilon) \to 1.$$

Outline

Definition

A function $f(x)$ is convex $\smile$ over $\mathbb{R}^N$ if for all $x_1, x_2 \in \mathbb{R}^N$ and $0 \leq \lambda \leq 1$:

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda)f(x_2)$$

We say f is strictly convex $\smile$ if for all $x_1, x_2 \in \mathbb{R}^N$ the equality holds only for $\lambda = 0$ and $\lambda = 1$.

Similarly, a function f is concave $\frown$ if $-f$ is convex $\smile$, i.e. if every cord of the function lies below the function.

Jensen's Inequality for Convex Functions

Theorem: Jensen's Inequality

If $f : \mathbb{R}^N \to \mathbb{R}$ is a convex $\smile$ function and X is a $\mathbb{R}^N$-valued r.v. then:

$$f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)].$$

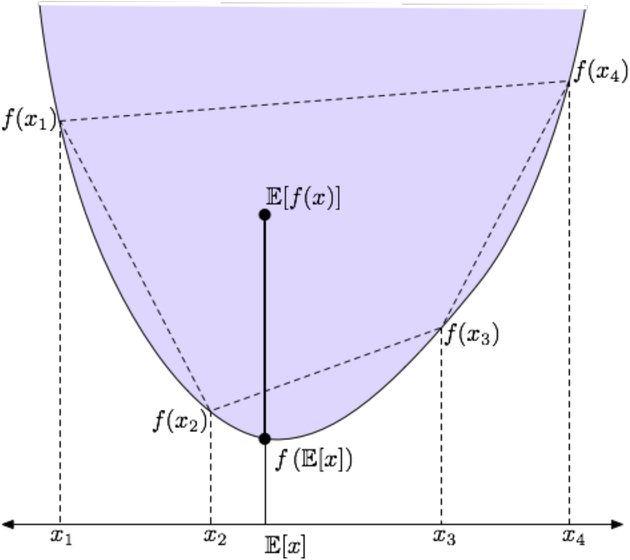
Moreover, if f is strictly convex $\smile$, the equality implies that $X = \mathbb{E}[X]$ with probability 1, i.e X is a constant.

In other words, for a probability vector $\mathbf{p}$,

$$f\left(\sum_{i=1}^{N} p_i x_i\right) \leq \sum_{i=1}^{N} p_i f(x_i).$$

Similarly for a concave $\frown$ function: $\mathbb{E}[f(X)] \leq f(\mathbb{E}[X])$.

Jensen's Inequality: "Proof"



Gibb's Inequality

Theorem

The relative entropy (or KL divergence) between two distributions $p(X)$ and $q(X)$ with $X \in \mathcal{X}$ is non-negative:

$$D_{\mathrm{KL}}(p\|q) \geq 0$$

with equality if and only if $p(x) = q(x)$ for all x.

Recall that: $D_{\mathrm{KL}}(p\|q) = \sum_{x\in\mathcal{X}} p(x) \log \frac{p(x)}{q(x)} = \mathbb{E}_{p(X)}\left[\log \frac{p(X)}{q(X)}\right]$

Let $\mathcal{A} = \{x : p(x) > 0\}$. Then:

$$\begin{aligned} -D_{\mathrm{KL}}(p\|q) = \sum_{x\in\mathcal{A}} p(x) \log \frac{q(x)}{p(x)} &\leq \log \sum_{x\in\mathcal{A}} p(x) \frac{q(x)}{p(x)} && \text{Jensen's inequality} \\ &\leq \log \sum_{x\in\mathcal{X}} q(x) = \log 1 = 0 \end{aligned}$$

Non-Negativity of Mutual Information

Corollary

For any two random variables X, Y:

$$I(X; Y) \geq 0,$$

with equality if and only if X and Y are statistically independent.

Proof: We simply use the definition of mutual information and Gibbs' inequality:

$$I(X; Y) = D_{\mathrm{KL}}(p(X, Y) \| p(X)p(Y)) \geq 0,$$

with equality if and only if $p(X, Y) = p(X)p(Y)$, i.e. X and Y are independent.

Conditioning Reduces Entropy

Information Cannot Hurt — Proof

Theorem

For any two random variables X, Y,

$$H(X|Y) \leq H(X),$$

with equality if and only if X and Y are independent.

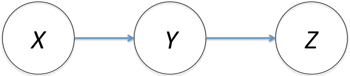
Proof: We simply use the non-negativity of mutual information:

$$\begin{aligned} I(X;Y) &\geq 0 \\ H(X) - H(X|Y) &\geq 0 \\ H(X|Y) &\leq H(X) \end{aligned}$$

with equality if and only if $p(X,Y) = p(X)p(Y)$, i.e X and Y are independent.

Data are helpful, they don't increase uncertainty on average.

Markov Chain



Definition

Random variables X, Y, Z are said to form a Markov chain in that order (denoted by $X \to Y \to Z$) if their joint probability distribution can be written as:

$$p(X, Y, Z) = p(X)p(Y|X)p(Z|Y)$$

Consequences:

- $X \to Y \to Z$ if and only if X and Z are conditionally independent given Y.
- $X \to Y \to Z$ implies that $Z \to Y \to X$.
- If $Z = f(Y)$, then $X \to Y \to Z$

Data-Processing Inequality

Definition

Theorem

if $X \to Y \to Z$ then: $I(X;Y) \geq I(X;Z)$

- X is the state of the world, Y is the data gathered and Z is the processed data
- No "clever" manipulation of the data can improve the inferences that can be made from the data
- No processing of Y, deterministic or random, can increase the information that Y contains about X

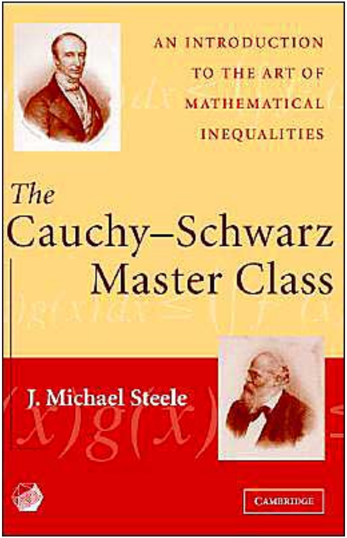
Recall that the chain rule for mutual information states that:

$$\begin{aligned} I(X;Y,Z) &= I(X;Y) + I(X;Z|Y) \\ &= I(X;Z) + I(X;Y|Z) \end{aligned}$$

Therefore:

$$\begin{aligned} I(X;Y) + \underbrace{I(X;Z|Y)}_{0} &= I(X;Z) + I(X;Y|Z) && \text{Markov chain assumption} \\ I(X;Y) &= I(X;Z) + I(X;Y|Z) && \text{but } I(X;Y|Z) \geq 0 \\ I(X;Y) &\geq I(X;Z) \end{aligned}$$

More on inequalities



Outline

1. Inequalities

2. Key Results
   - The Source Coding Theorem for Lossy Uniform-Length Coding
   - The Source Coding Theorem for Lossless Variable-Length Coding
   - The Noisy-Channel Coding Theorem

Key Results: Overview

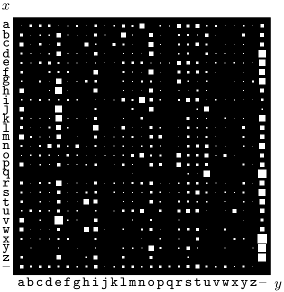
What is Compression?

Cn y rd ths mssg wtht ny vwls?

What is Compression?

Cn y rd ths mssg wtht ny vwls?

It is not too difficult to read as there is redundancy in English text.
(Estimates of 1-1.5 bits per character, compared to $\log_2 26 \approx 4.7$)



(a) $P(y \mid x)$

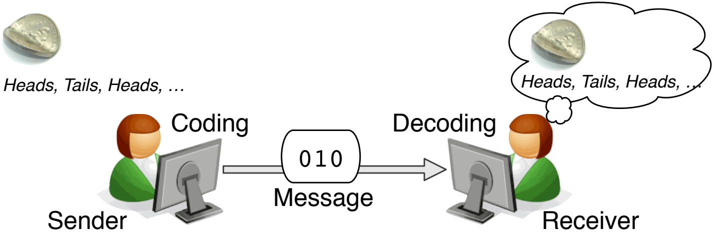
- If you see a "q", it is very likely to be followed with a "u"
- The letter "e" is much more common than "j"
- Compression exploits differences in relative probability of symbols or blocks of symbols

We will breifly look at two types of compression: **lossy** (trade off size and reliability) and **lossless** (unambiguous decoding).

A General Communication Game

Data compression is the process of replacing a message with a smaller message which can be reliably converted back to the original.

- Sender & Receiver agree on code for each outcome ahead of time (e.g., 0 for *Heads*; 1 for *Tails*)
- Sender observes outcomes then codes and sends message
- Receiver decodes message and recovers outcome sequence
- Want small messages on average when outcomes are from a fixed, known, but uncertain source (e.g., coin flips with known bias)



Codes for Compression

Notation:

- If $\mathcal{A}$ is a finite set then $\mathcal{A}^N$ is the set of all *strings of length N*.
- $\mathcal{A}^+ = \bigcup_N \mathcal{A}^N$ is the set of *all finite strings*

Examples:

- $\{0,1\}^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$
- $\{0,1\}^+ = \{0, 1, 00, 01, 10, 11, 000, 001, 010, \ldots\}$

Codes for Compression

Notation:

- If $\mathcal{A}$ is a finite set then $\mathcal{A}^N$ is the set of all *strings of length* N.
- $\mathcal{A}^+ = \bigcup_N \mathcal{A}^N$ is the set of *all finite strings*

Examples:

- $\{0,1\}^3 = \{000, 001, 010, 011, 100, 101, 110, 111\}$
- $\{0,1\}^+ = \{0, 1, 00, 01, 10, 11, 000, 001, 010, \ldots\}$

Binary Symbol Code

Let X be an ensemble with $\mathcal{A}_X = \{a_1, \ldots, a_I\}$.
A function $c : \mathcal{A}_X \to \{0,1\}^+$ is a **code** for X.

- The binary string $c(x)$ is the **codeword** for $x \in \mathcal{A}_X$
- The **length** of the codeword for for x is denoted $\ell(x)$.
  Shorthand: $\ell_i = \ell(a_i)$ for $i = 1 \ldots, I$.
- The **extension** of c assigns codewords to any sequence $x_1 x_2 \ldots x_N$ from $\mathcal{A}^+$ by $c(x_1 \ldots x_N) = c(x_1) \ldots c(x_N)$

X is an ensemble with $\mathcal{A}_X = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$

Example 1 (Uniform Code):

- Let $c(\mathtt{a}) = 0001$, $c(\mathtt{b}) = 0010$, $c(\mathtt{c}) = 0100$, $c(\mathtt{d}) = 1000$
- Shorthand: $C_1 = \{0001, 0010, 0100, 1000\}$
- All codewords have *length* 4. That is, $\ell_1 = \ell_2 = \ell_3 = \ell_4 = 4$
- The *extension* of c maps $\mathtt{aba} \in \mathcal{A}_X^3 \subset \mathcal{A}_X^+$ to 000100100001

X is an ensemble with $\mathcal{A}_X = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$

Example 1 (Uniform Code):

- Let $c(\mathtt{a}) = 0001$, $c(\mathtt{b}) = 0010$, $c(\mathtt{c}) = 0100$, $c(\mathtt{d}) = 1000$
- Shorthand: $C_1 = \{0001, 0010, 0100, 1000\}$
- All codewords have *length* 4. That is, $\ell_1 = \ell_2 = \ell_3 = \ell_4 = 4$
- The *extension* of c maps $\mathtt{aba} \in \mathcal{A}_X^3 \subset \mathcal{A}_X^+$ to 000100100001

Example 2 (Variable-Length Code):

- Let $c(\mathtt{a}) = 0$, $c(\mathtt{b}) = 10$, $c(\mathtt{c}) = 110$, $c(\mathtt{d}) = 111$
- Shorthand: $C_2 = \{0, 10, 110, 111\}$
- In this case $\ell_1 = 1$, $\ell_2 = 2$, $\ell_3 = \ell_4 = 3$
- The *extension* of c maps $\mathtt{aba} \in \mathcal{A}_X^3 \subset \mathcal{A}_X^+$ to 0100

Expected Code Length

Expected Code Length

The **expected length** for a code C for ensemble X with $\mathcal{A}_X = \{a_1, \ldots, a_I\}$ and $\mathbf{p} = (p_1, \ldots, p_I)$ is

$$L(C, X) = \mathbb{E}_{x\sim\mathbf{p}}[\ell(x)] = \sum_{x\in\mathcal{A}_X} p(x)\,\ell(x) = \sum_{i=1}^{I} p_i\,\ell_i$$

Example: X has $\mathcal{A}_X = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$ and $\mathcal{P} = \{\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}\}$

1. The code $C_1 = \{0001, 0010, 0100, 1000\}$ has

$$L(C_1, X) = \sum_{i=1}^{4} p_i\,\ell_i = 4$$

2. The code $C_2 = \{0, 10, 110, 111\}$ has

$$L(C_2, X) = \sum_{i=1}^{4} p_i\,\ell_i = \frac{1}{2} \times 1 + \frac{1}{4} \times 2 + \frac{1}{8} \times 3 + \frac{1}{8} \times 3 = 1.25$$

Uniform Lossy Codes: What is the best we can do?

Consider a coin with $P(Heads) = 0.9$. If we want perfect transmission:

- Coding single outcomes requires 1 bit/outcome
- Coding 10 outcomes at a time needs 1 bits/outcome

Uniform Lossy Codes: What is the best we can do?

Consider a coin with $P(Heads) = 0.9$. If we want perfect transmission:

- Coding single outcomes requires 1 bit/outcome
- Coding 10 outcomes at a time needs 1 bits/outcome

However, if we are happy to fail on up to 2% of the sequences we can ignore any sequence of 10 outcomes with more than 3 tails (*Why?*).

Uniform Lossy Codes: What is the best we can do?

Consider a coin with $P(Heads) = 0.9$. If we want perfect transmission:

- Coding single outcomes requires 1 bit/outcome
- Coding 10 outcomes at a time needs 1 bits/outcome

However, if we are happy to fail on up to 2% of the sequences we can ignore any sequence of 10 outcomes with more than 3 tails (*Why?*). But there are only $176 < 2^8$ sequences with 3 or fewer tails (*Why?*).

- Coding 10 outcomes with 2% failure doable with 0.8 bits/outcome
- *Smallest bits/outcome needed for 10,000 outcome sequences?*

Uniform Lossy Codes: What is the best we can do?

Consider a coin with $P(Heads) = 0.9$. If we want perfect transmission:

- Coding single outcomes requires 1 bit/outcome
- Coding 10 outcomes at a time needs 1 bits/outcome

However, if we are happy to fail on up to 2% of the sequences we can ignore any sequence of 10 outcomes with more than 3 tails (*Why?*). But there are only $176 < 2^8$ sequences with 3 or fewer tails (*Why?*).

- Coding 10 outcomes with 2% failure doable with 0.8 bits/outcome
- *Smallest bits/outcome needed for 10,000 outcome sequences?*

Source Coding Theorem (Informal Statement)

If you want to uniformly code large sequences of outcomes with any degree of reliability from a random source then the average number of bits per outcome you will **need** is roughly equal to the entropy of that source.

Uniform Lossy Codes: What is the best we can do?

Consider a coin with $P(Heads) = 0.9$. If we want perfect transmission:

- Coding single outcomes requires 1 bit/outcome
- Coding 10 outcomes at a time needs 1 bits/outcome

However, if we are happy to fail on up to 2% of the sequences we can ignore any sequence of 10 outcomes with more than 3 tails (*Why?*). But there are only $176 < 2^8$ sequences with 3 or fewer tails (*Why?*).

- Coding 10 outcomes with 2% failure doable with 0.8 bits/outcome
- *Smallest bits/outcome needed for 10,000 outcome sequences?*

Source Coding Theorem (Informal Statement)

If you want to uniformly code large sequences of outcomes with any degree of reliability from a random source then the average number of bits per outcome you will **need** is roughly equal to the entropy of that source.

To define: "Uniformly code", "large sequences", "degree of reliability", "average number of bits per outcome", "roughly equal"

Essential Bit Content

There is an inherent trade off between the number of bits required in a uniform lossy code and the probability of being able to code an outcome

Smallest δ-sufficient subset

Let X be an ensemble and for $\delta \geq 0$ define S_δ to be the smallest subset of $\mathcal{A}_X$ such that

$$P(x \in S_\delta) \geq 1 - \delta$$

Essential Bit Content

There is an inherent trade off between the number of bits required in a uniform lossy code and the probability of being able to code an outcome

Smallest δ-sufficient subset

Let X be an ensemble and for $\delta \geq 0$ define S_δ to be the smallest subset of $\mathcal{A}_X$ such that

$$P(x \in S_\delta) \geq 1 - \delta$$

x	$P(\mathbf{x})$
a	1/4
b	1/4
c	1/4
d	3/16
e	1/64
f	1/64
g	1/64
h	1/64

- Outcomes ranked (high–low) by $P(x = a_i)$ removed to make set S_δ with $P(x \in S_\delta) \geq 1 - \delta$

$$\delta = 0 \; : \; S_\delta = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}, \mathtt{f}, \mathtt{g}, \mathtt{h}\}$$

Essential Bit Content

There is an inherent trade off between the number of bits required in a uniform lossy code and the probability of being able to code an outcome

Smallest δ-sufficient subset

Let X be an ensemble and for $\delta \geq 0$ define S_δ to be the smallest subset of $\mathcal{A}_X$ such that

$$P(x \in S_\delta) \geq 1 - \delta$$

x	$P(\mathbf{x})$
a	1/4
b	1/4
c	1/4
d	3/16
e	1/64
f	1/64
g	1/64

- Outcomes ranked (high–low) by $P(x = a_i)$ removed to make set S_δ with $P(x \in S_\delta) \geq 1 - \delta$

$$\delta = 0 \; : \; S_\delta = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}, \mathtt{f}, \mathtt{g}, \mathtt{h}\}$$

$$\delta = 1/64 \; : \; S_\delta = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}, \mathtt{f}, \mathtt{g}\}$$

Essential Bit Content

There is an inherent trade off between the number of bits required in a uniform lossy code and the probability of being able to code an outcome

Smallest δ-sufficient subset

Let X be an ensemble and for $\delta \geq 0$ define S_δ to be the smallest subset of $\mathcal{A}_X$ such that

$$P(x \in S_\delta) \geq 1 - \delta$$

x	$P(\mathbf{x})$
a	1/4
b	1/4
c	1/4
d	3/16

- Outcomes ranked (high–low) by $P(x = a_i)$ removed to make set S_δ with $P(x \in S_\delta) \geq 1 - \delta$

$$\delta = 0 \; : \; S_\delta = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}, \mathtt{f}, \mathtt{g}, \mathtt{h}\}$$

$$\delta = 1/64 \; : \; S_\delta = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}, \mathtt{f}, \mathtt{g}\}$$

$$\delta = 1/16 \; : \; S_\delta = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$$

Essential Bit Content

There is an inherent trade off between the number of bits required in a uniform lossy code and the probability of being able to code an outcome

Smallest δ-sufficient subset

Let X be an ensemble and for $\delta \geq 0$ define S_δ to be the smallest subset of $\mathcal{A}_X$ such that

$$P(x \in S_\delta) \geq 1 - \delta$$

x	$P(\mathbf{x})$
a	1/4

- Outcomes ranked (high–low) by $P(x = a_i)$ removed to make set S_δ with $P(x \in S_\delta) \geq 1 - \delta$

$$\delta = 0 \;:\; S_\delta = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}, \mathtt{f}, \mathtt{g}, \mathtt{h}\}$$

$$\delta = 1/64 \;:\; S_\delta = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}, \mathtt{f}, \mathtt{g}\}$$

$$\delta = 1/16 \;:\; S_\delta = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$$

$$\delta = 3/4 \;:\; S_\delta = \{\mathtt{a}\}$$

Essential Bit Content

Trade off between a probability of δ of not coding an outcome and size of uniform code is captured by the essential bit content

Essential Bit Content

Let X be an ensemble then for $\delta \geq 0$ the **essential bit content** of X is

$$H_\delta(X) \stackrel{\text{def}}{=} \log_2 |S_\delta|$$

Essential Bit Content

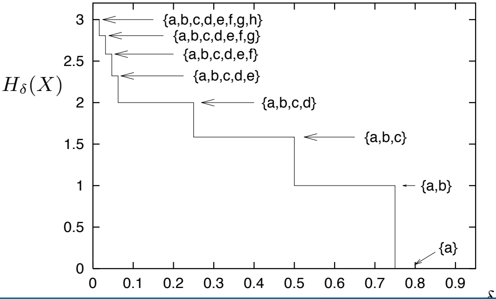
Trade off between a probability of δ of not coding an outcome and size of uniform code is captured by the essential bit content

Essential Bit Content

Let X be an ensemble then for $\delta \geq 0$ the **essential bit content** of X is

$$H_\delta(X) \stackrel{\text{def}}{=} \log_2 |S_\delta|$$

x	$P(\mathbf{x})$
a	1/4
b	1/4
c	1/4
d	3/16
e	1/64
f	1/64
g	1/64
h	1/64



The Source Coding Theorem for Uniform Codes

(Theorem 4.1 in MacKay)

The Source Coding Theorem for Uniform Codes

Let X be an ensemble with entropy $H = H(X)$ bits. Given $\epsilon > 0$ and $0 < \delta < 1$, there exists a positive integer N_0 such that for all $N > N_0$

$$\left| \frac{1}{N} H_\delta \left(X^N \right) - H \right| < \epsilon.$$

The Source Coding Theorem for Uniform Codes
(Theorem 4.1 in MacKay)

The Source Coding Theorem for Uniform Codes

Let X be an ensemble with entropy $H = H(X)$ bits. Given $\epsilon > 0$ and $0 < \delta < 1$, there exists a positive integer N_0 such that for all $N > N_0$

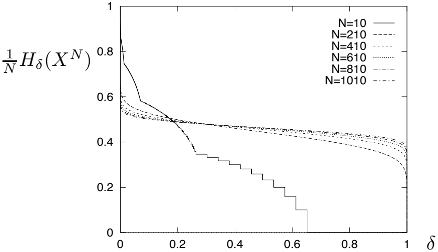
$$\left| \frac{1}{N} H_\delta \left(X^N \right) - H \right| < \epsilon.$$

- The term $\frac{1}{N} H_\delta(X^N)$ is the average number of bits per symbol required to uniformly code all but a proportion δ of length N sequences.
- Given a tiny probability of error δ, the average bits per symbol can be made as close to H as required.
- Even if we allow a large probability of error we cannot compress more than H bits ber symbol.

The Source Coding Theorem

Let X be an ensemble with entropy $H = H(X)$ bits. Given $\epsilon > 0$ and $0 < \delta < 1$, there exists a positive integer N_0 such that for all $N > N_0$

$$\left| \frac{1}{N} H_\delta \left(X^N \right) - H \right| < \epsilon.$$



- Given a tiny probability of error δ, the average bits per outcome can be made as close to H as required.
- Even if we allow a large probability of error we cannot compress more than H bits per outcome for large sequences.

Typical Sets and the AEP

Typical Set

For "closeness" $\beta > 0$ the typical set $T_{N\beta}$ for X^N is

$$T_{N\beta} \stackrel{\text{def}}{=} \left\{ \mathbf{x} : \left| -\frac{1}{N} \log_2 P(\mathbf{x}) - H(X) \right| < \beta \right\}$$

The name "typical" is used since $\mathbf{x} \in T_{N\beta}$ will have roughly $p_1 N$ occurences of symbol a_1, $p_2 N$ of a_2, ..., $p_K N$ of a_K.

Typical Sets and the AEP

Typical Set

For "closeness" $\beta > 0$ the typical set $T_{N\beta}$ for X^N is

$$T_{N\beta} \stackrel{\text{def}}{=} \left\{ \mathbf{x} : \left| -\frac{1}{N} \log_2 P(\mathbf{x}) - H(X) \right| < \beta \right\}$$

The name "typical" is used since $\mathbf{x} \in T_{N\beta}$ will have roughly $p_1 N$ occurences of symbol a_1, $p_2 N$ of a_2, ..., $p_K N$ of a_K.

Asymptotic Equipartition Property (Informal)

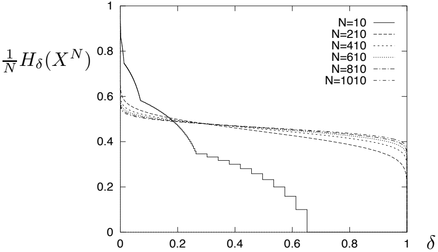
As $N \to \infty$, $-\frac{1}{N} \log_2 P(x_1, \ldots, x_N)$ is close to $H(X)$ with high probability.

For large block sizes "almost all sequences are typical" (i.e., in $T_{N\beta}$). This means $T_{N\beta}$ can be made to "look like" S_δ for any δ by choosing N large enough. This is useful since $T_{N\beta}$ is easy to count (size $\approx 2^{NH(X)}$) while S_δ is not (size varies with distribution)

The Source Coding Theorem

Let X be an ensemble with entropy $H = H(X)$ bits. Given $\epsilon > 0$ and $0 < \delta < 1$, there exists a positive integer N_0 such that for all $N > N_0$

$$\left| \frac{1}{N} H_\delta \left(X^N \right) - H \right| < \epsilon.$$



Proof Idea: As N increases

- $T_{N\beta}$ has $\sim 2^{NH(X)}$ elements
- almost all $\mathbf{x}$ are in $T_{N\beta}$
- S_δ and $T_{N\beta}$ increasingly overlap
- so $\log_2 |S_\delta| \sim NH$

Outline

1. Inequalities

2. Key Results
   - The Source Coding Theorem for Lossy Uniform-Length Coding
   - The Source Coding Theorem for Lossless Variable-Length Coding
   - The Noisy-Channel Coding Theorem

Lossless Codes: Unique Decodeability

Unique Decodeability

A code c for X is **uniquely decodeable** if no two strings from $\mathcal{A}_X^+$ have the same codeword. That is, for all $\mathbf{x}, \mathbf{y} \in \mathcal{A}_X^+$

$$\mathbf{x} \neq \mathbf{y} \implies c(\mathbf{x}) \neq c(\mathbf{y})$$

Lossless Codes: Unique Decodeability

Unique Decodeability

A code c for X is **uniquely decodeable** if no two strings from $\mathcal{A}_X^+$ have the same codeword. That is, for all $\mathbf{x}, \mathbf{y} \in \mathcal{A}_X^+$

$$\mathbf{x} \neq \mathbf{y} \implies c(\mathbf{x}) \neq c(\mathbf{y})$$

Examples:

- $C_1 = \{0001, 0010, 0100, 1000\}$ is uniquely decodeable Why?
- $C_2 = \{0, 10, 110, 111\}$ is uniquely decodeable
- $C_2' = \{1, 10, 110, 111\}$ is not uniquely decodeable because

$$c(\mathtt{aaa}) = c(\mathtt{d}) = 111 \quad \text{and} \quad c(\mathtt{ab}) = c(\mathtt{c}) = 110$$

Prefix Codes

a.k.a *prefix-free* or *instantaneous* codes

There is an simple property of codes that *guarantees* unique decodeability.

Prefix

A codeword $\mathbf{c} \in \{0,1\}^+$ is said to be a **prefix** of another codeword $\mathbf{c}' \in \{0,1\}^+$ if there exists a string $\mathbf{t} \in \{0,1\}^+$ such that $\mathbf{c}' = \mathbf{ct}$.

Prefix Codes

a.k.a *prefix-free* or *instantaneous* codes

There is an simple property of codes that *guarantees* unique decodeability.

Prefix

A codeword $\mathbf{c} \in \{0,1\}^+$ is said to be a **prefix** of another codeword $\mathbf{c}' \in \{0,1\}^+$ if there exists a string $\mathbf{t} \in \{0,1\}^+$ such that $\mathbf{c}' = \mathbf{ct}$.

Example: 01101 has prefixes 0, 01, 011, 0110.

Prefix Codes

a.k.a *prefix-free* or *instantaneous* codes

There is an simple property of codes that *guarantees* unique decodeability.

Prefix

A codeword $\mathbf{c} \in \{0,1\}^+$ is said to be a **prefix** of another codeword $\mathbf{c}' \in \{0,1\}^+$ if there exists a string $\mathbf{t} \in \{0,1\}^+$ such that $\mathbf{c}' = \mathbf{ct}$.

Example: 01101 has prefixes 0, 01, 011, 0110.

Prefix Codes

A code $C = \{c_1, \ldots, c_I\}$ is a **prefix code** if for every codeword $c_i \in C$ there is no prefix of c_i in C.

Prefix Codes

a.k.a *prefix-free* or *instantaneous* codes

There is an simple property of codes that *guarantees* unique decodeability.

Prefix

A codeword $\mathbf{c} \in \{0,1\}^+$ is said to be a **prefix** of another codeword $\mathbf{c}' \in \{0,1\}^+$ if there exists a string $\mathbf{t} \in \{0,1\}^+$ such that $\mathbf{c}' = \mathbf{ct}$.

Example: 01101 has prefixes 0, 01, 011, 0110.

Prefix Codes

A code $C = \{c_1, \ldots, c_I\}$ is a **prefix code** if for every codeword $c_i \in C$ there is no prefix of c_i in C.

Examples:

- $C_1 = \{0001, 0010, 0100, 1000\}$ is prefix-free
- $C_2 = \{0, 10, 110, 111\}$ is prefix-free
- $C_2' = \{1, 10, 110, 111\}$ is *not* prefix free since $c_3 = 110 = c_1 c_2$

Prefix Codes

a.k.a *prefix-free* or *instantaneous* codes

There is an simple property of codes that *guarantees* unique decodeability.

Prefix

A codeword $\mathbf{c} \in \{0,1\}^+$ is said to be a **prefix** of another codeword $\mathbf{c}' \in \{0,1\}^+$ if there exists a string $\mathbf{t} \in \{0,1\}^+$ such that $\mathbf{c}' = \mathbf{ct}$.

Example: 01101 has prefixes 0, 01, 011, 0110.

Prefix Codes

A code $C = \{c_1, \ldots, c_I\}$ is a **prefix code** if for every codeword $c_i \in C$ there is no prefix of c_i in C.

Examples:

- $C_1 = \{0001, 0010, 0100, 1000\}$ is prefix-free
- $C_2 = \{0, 10, 110, 111\}$ is prefix-free
- $C_2' = \{1, 10, 110, 111\}$ is *not* prefix free since $c_3 = 110 = c_1 c_2$
- $C_2'' = \{1, 01, 110, 111\}$ is *not* prefix free since $c_3 = 110 = c_1 10$

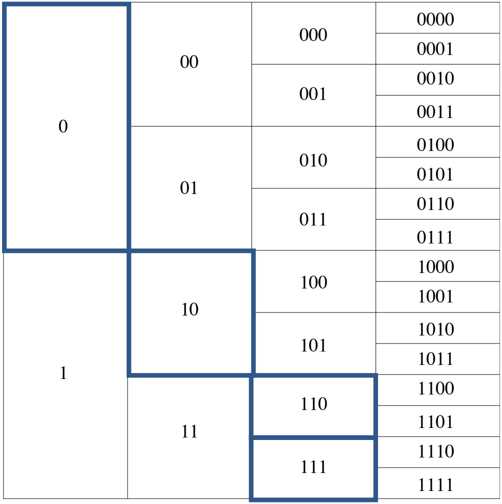
Prefix Codes as Trees

$$C_1 = \{0001, 0010, 0100, 1000\}$$

<table>
<tr><td rowspan="8">0</td><td rowspan="4">00</td><td rowspan="2">000</td><td>0000</td></tr>
<tr><td>$\boxed{0001}$</td></tr>
<tr><td rowspan="2">001</td><td>$\boxed{0010}$</td></tr>
<tr><td>0011</td></tr>
<tr><td rowspan="4">01</td><td rowspan="2">010</td><td>$\boxed{0100}$</td></tr>
<tr><td>0101</td></tr>
<tr><td rowspan="2">011</td><td>0110</td></tr>
<tr><td>0111</td></tr>
<tr><td rowspan="8">1</td><td rowspan="4">10</td><td rowspan="2">100</td><td>$\boxed{1000}$</td></tr>
<tr><td>1001</td></tr>
<tr><td rowspan="2">101</td><td>1010</td></tr>
<tr><td>1011</td></tr>
<tr><td rowspan="4">11</td><td rowspan="2">110</td><td>1100</td></tr>
<tr><td>1101</td></tr>
<tr><td rowspan="2">111</td><td>1110</td></tr>
<tr><td>1111</td></tr>
</table>

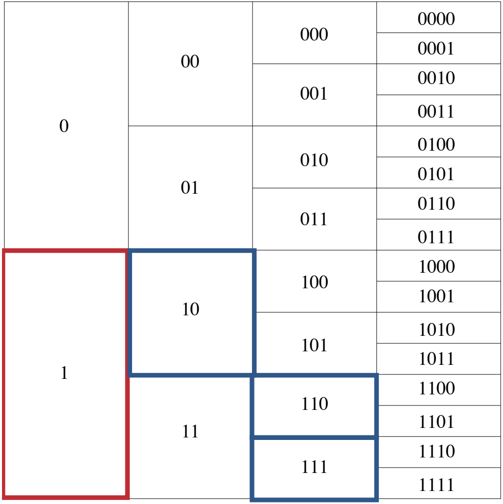
Prefix Codes as Trees

$$C_2 = \{0, 10, 110, 111\}$$



Prefix Codes as Trees

$$C_2' = \{1, 10, 110, 111\}$$



Lengths for Prefix Codes

Suppose someone said "I want codes with codewords lengths":

- $L_1 = \{4, 4, 4, 4\}$
- $L_2 = \{1, 2, 3, 3\}$
- $L_3 = \{2, 2, 3, 4, 4\}$
- $L_4 = \{1, 3, 3, 3, 3, 4\}$

Could you construct such codes? Uniquely Decodeable? Prefix-free?

Lengths for Prefix Codes

Suppose someone said “I want codes with codewords lengths”:

- $L_1 = \{4, 4, 4, 4\}$ — $C_1 = \{0001, 0010, 0100, 1000\}$
- $L_2 = \{1, 2, 3, 3\}$
- $L_3 = \{2, 2, 3, 4, 4\}$
- $L_4 = \{1, 3, 3, 3, 3, 4\}$

Could you construct such codes? Uniquely Decodeable? Prefix-free?

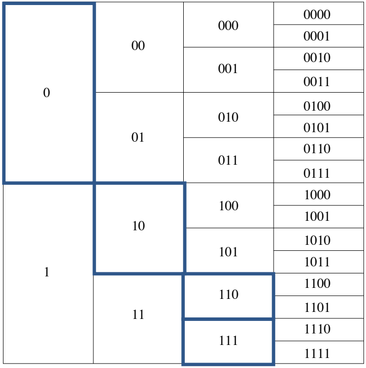
<table>
<tr><td rowspan="8">0</td><td rowspan="4">00</td><td rowspan="2">000</td><td>0000</td></tr>
<tr><td>$\boxed{0001}$</td></tr>
<tr><td rowspan="2">001</td><td>$\boxed{0010}$</td></tr>
<tr><td>0011</td></tr>
<tr><td rowspan="4">01</td><td rowspan="2">010</td><td>$\boxed{0100}$</td></tr>
<tr><td>0101</td></tr>
<tr><td rowspan="2">011</td><td>0110</td></tr>
<tr><td>0111</td></tr>
<tr><td rowspan="8">1</td><td rowspan="4">10</td><td rowspan="2">100</td><td>$\boxed{1000}$</td></tr>
<tr><td>1001</td></tr>
<tr><td rowspan="2">101</td><td>1010</td></tr>
<tr><td>1011</td></tr>
<tr><td rowspan="4">11</td><td rowspan="2">110</td><td>1100</td></tr>
<tr><td>1101</td></tr>
<tr><td rowspan="2">111</td><td>1110</td></tr>
<tr><td>1111</td></tr>
</table>

Lengths for Prefix Codes

Suppose someone said "I want codes with codewords lengths":

- $L_1 = \{4, 4, 4, 4\}$ — $C_1 = \{0001, 0010, 0100, 1000\}$
- $L_2 = \{1, 2, 3, 3\}$ — $C_2 = \{0, 10, 110, 111\}$
- $L_3 = \{2, 2, 3, 4, 4\}$
- $L_4 = \{1, 3, 3, 3, 3, 4\}$

Could you construct such codes? Uniquely Decodeable? Prefix-free?



Suppose someone said "I want codes with codewords lengths":

- $L_1 = \{4, 4, 4, 4\}$ — $C_1 = \{0001, 0010, 0100, 1000\}$
- $L_2 = \{1, 2, 3, 3\}$ — $C_2 = \{0, 10, 110, 111\}$
- $L_3 = \{2, 2, 3, 4, 4\}$ — $C_3 = \{00, \quad , \quad , \quad , \quad \}$
- $L_4 = \{1, 3, 3, 3, 3, 4\}$

Could you construct such codes? Uniquely Decodeable? Prefix-free?

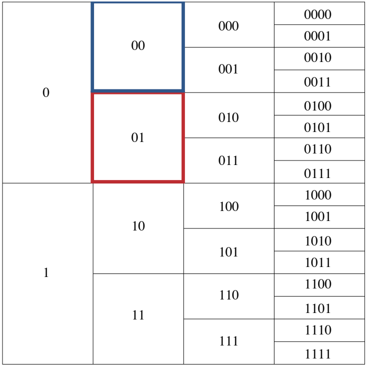
<table>
<tr><td rowspan="8">0</td><td rowspan="4">00</td><td rowspan="2">000</td><td>0000</td></tr>
<tr><td>0001</td></tr>
<tr><td rowspan="2">001</td><td>0010</td></tr>
<tr><td>0011</td></tr>
<tr><td rowspan="4">01</td><td rowspan="2">010</td><td>0100</td></tr>
<tr><td>0101</td></tr>
<tr><td rowspan="2">011</td><td>0110</td></tr>
<tr><td>0111</td></tr>
<tr><td rowspan="8">1</td><td rowspan="4">10</td><td rowspan="2">100</td><td>1000</td></tr>
<tr><td>1001</td></tr>
<tr><td rowspan="2">101</td><td>1010</td></tr>
<tr><td>1011</td></tr>
<tr><td rowspan="4">11</td><td rowspan="2">110</td><td>1100</td></tr>
<tr><td>1101</td></tr>
<tr><td rowspan="2">111</td><td>1110</td></tr>
<tr><td>1111</td></tr>
</table>

Lengths for Prefix Codes

Suppose someone said "I want codes with codewords lengths":

- $L_1 = \{4, 4, 4, 4\}$ — $C_1 = \{0001, 0010, 0100, 1000\}$
- $L_2 = \{1, 2, 3, 3\}$ — $C_2 = \{0, 10, 110, 111\}$
- $L_3 = \{2, 2, 3, 4, 4\}$ — $C_3 = \{00, 01, \quad, \quad, \quad\}$
- $L_4 = \{1, 3, 3, 3, 3, 4\}$

Could you construct such codes? Uniquely Decodeable? Prefix-free?

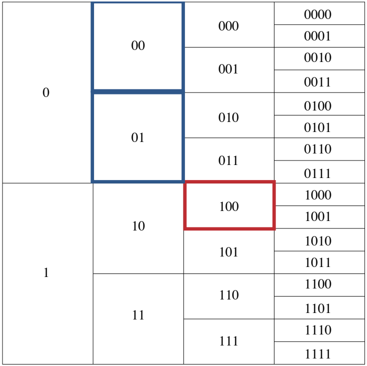


Lengths for Prefix Codes

Suppose someone said "I want codes with codewords lengths":

- $L_1 = \{4, 4, 4, 4\}$ — $C_1 = \{0001, 0010, 0100, 1000\}$
- $L_2 = \{1, 2, 3, 3\}$ — $C_2 = \{0, 10, 110, 111\}$
- $L_3 = \{2, 2, 3, 4, 4\}$ — $C_3 = \{00, 01, 100, \quad , \quad \}$
- $L_4 = \{1, 3, 3, 3, 3, 4\}$

Could you construct such codes? Uniquely Decodeable? Prefix-free?

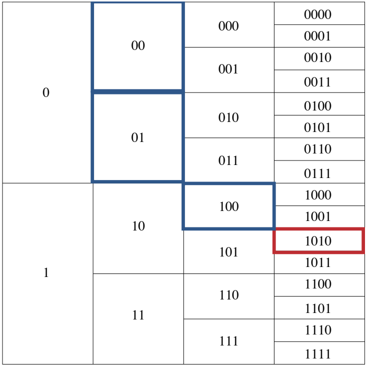


Lengths for Prefix Codes

Suppose someone said "I want codes with codewords lengths":

- $L_1 = \{4, 4, 4, 4\}$ — $C_1 = \{0001, 0010, 0100, 1000\}$
- $L_2 = \{1, 2, 3, 3\}$ — $C_2 = \{0, 10, 110, 111\}$
- $L_3 = \{2, 2, 3, 4, 4\}$ — $C_3 = \{00, 01, 100, 1010, \quad\quad \}$
- $L_4 = \{1, 3, 3, 3, 3, 4\}$

Could you construct such codes? Uniquely Decodeable? Prefix-free?

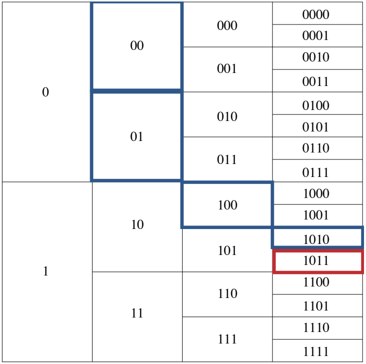


Lengths for Prefix Codes

Suppose someone said "I want codes with codewords lengths":

- $L_1 = \{4, 4, 4, 4\}$ — $C_1 = \{0001, 0010, 0100, 1000\}$
- $L_2 = \{1, 2, 3, 3\}$ — $C_2 = \{0, 10, 110, 111\}$
- $L_3 = \{2, 2, 3, 4, 4\}$ — $C_3 = \{00, 01, 100, 1010, 1011\}$
- $L_4 = \{1, 3, 3, 3, 3, 4\}$

Could you construct such codes? Uniquely Decodeable? Prefix-free?

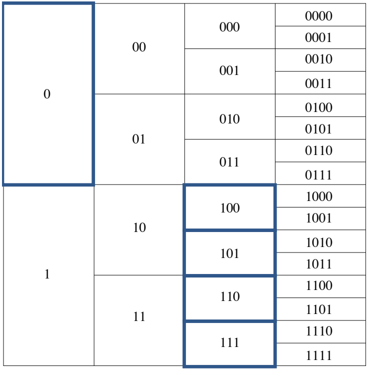


Lengths for Prefix Codes

Suppose someone said "I want codes with codewords lengths":

- $L_1 = \{4, 4, 4, 4\}$ — $C_1 = \{0001, 0010, 0100, 1000\}$
- $L_2 = \{1, 2, 3, 3\}$ — $C_2 = \{0, 10, 110, 111\}$
- $L_3 = \{2, 2, 3, 4, 4\}$ — $C_3 = \{00, 01, 100, 1010, 1011\}$
- $L_4 = \{1, 3, 3, 3, 3, 4\}$ — Impossible!

Could you construct such codes? Uniquely Decodeable? Prefix-free?



Prefixes Exclude Codes

Choosing a prefix codeword of length 1 — e.g., $c(\mathrm{a}) = 0$ — excludes:

- 2 x 2-bit codewords: $\{00, 01\}$
- 4 x 3-bit codewords: $\{000, 001, 010, 011\}$
- 8 x 4-bit codewords: $\{0000, 0001, \ldots, 0111\}$
- In general, an ℓ-bit codeword excludes $2^{k-\ell}$ x k-bit codewords

For lengths $L = \{\ell_1, \ldots, \ell_I\}$ and $\ell^* = \max\{\ell_1, \ldots, \ell_I\}$, there will be

$$\sum_{i=1}^{I} 2^{\ell^* - \ell_i} \leq 2^{\ell^*}$$

excluded ℓ^*-bit codewords. But there are only 2^{ℓ^*} possible ℓ^*-bit codewords

Prefixes Exclude Codes

Choosing a prefix codeword of length 1 — e.g., $c(\mathrm{a}) = 0$ — excludes:

- 2 x 2-bit codewords: $\{00, 01\}$
- 4 x 3-bit codewords: $\{000, 001, 010, 011\}$
- 8 x 4-bit codewords: $\{0000, 0001, \ldots, 0111\}$
- In general, an ℓ-bit codeword excludes $2^{k-\ell}$ x k-bit codewords

For lengths $L = \{\ell_1, \ldots, \ell_I\}$ and $\ell^* = \max\{\ell_1, \ldots, \ell_I\}$, there will be

$$\frac{1}{2^{\ell^*}} \sum_{i=1}^{I} 2^{\ell^* - \ell_i} \leq 1$$

excluded ℓ^*-bit codewords. But there are only 2^{ℓ^*} possible ℓ^*-bit codewords

Prefixes Exclude Codes

Choosing a prefix codeword of length 1 — e.g., $c(\text{a}) = 0$ — excludes:

- 2 x 2-bit codewords: $\{00, 01\}$
- 4 x 3-bit codewords: $\{000, 001, 010, 011\}$
- 8 x 4-bit codewords: $\{0000, 0001, \ldots, 0111\}$
- In general, an ℓ-bit codeword excludes $2^{k-\ell}$ x k-bit codewords

For lengths $L = \{\ell_1, \ldots, \ell_I\}$ and $\ell^* = \max\{\ell_1, \ldots, \ell_I\}$, there will be

$$\sum_{i=1}^{I} 2^{-\ell_i} \leq 1$$

excluded ℓ^*-bit codewords. But there are only 2^{ℓ^*} possible ℓ^*-bit codewords

The Kraft Inequality

a.k.a. The Kraft-McMillan Inequality

Kraft Inequality

For any prefix (binary) code C, its codeword lengths $\{\ell_1, \ldots, \ell_I\}$ satisfy

$$\sum_{i=1}^{I} 2^{-\ell_i} \leq 1 \tag{1}$$

Conversely, if the set $\{\ell_1, \ldots, \ell_I\}$ satisfy (1) then there exists a prefix code C with those codeword lengths.

The Kraft Inequality

a.k.a. The Kraft-McMillan Inequality

Kraft Inequality

For any prefix (binary) code C, its codeword lengths $\{\ell_1, \ldots, \ell_I\}$ satisfy

$$\sum_{i=1}^{I} 2^{-\ell_i} \leq 1 \tag{1}$$

Conversely, if the set $\{\ell_1, \ldots, \ell_I\}$ satisfy (1) then there exists a prefix code C with those codeword lengths.

Examples:

1. $C_1 = \{0001, 0010, 0100, 1000\}$ is prefix and $\sum_{i=1}^{4} 2^{-4} = \frac{1}{4} \leq 1$

The Kraft Inequality

a.k.a. The Kraft-McMillan Inequality

Kraft Inequality

For any prefix (binary) code C, its codeword lengths $\{\ell_1, \ldots, \ell_I\}$ satisfy

$$\sum_{i=1}^{I} 2^{-\ell_i} \leq 1 \tag{1}$$

Conversely, if the set $\{\ell_1, \ldots, \ell_I\}$ satisfy (1) then there exists a prefix code C with those codeword lengths.

Examples:

1. $C_1 = \{0001, 0010, 0100, 1000\}$ is prefix and $\sum_{i=1}^{4} 2^{-4} = \frac{1}{4} \leq 1$
2. $C_2 = \{0, 10, 110, 111\}$ is prefix and $\sum_{i=1}^{4} 2^{-\ell_i} = \frac{1}{2} + \frac{1}{4} + \frac{2}{8} = 1$

The Kraft Inequality

a.k.a. The Kraft-McMillan Inequality

Kraft Inequality

For any prefix (binary) code C, its codeword lengths $\{\ell_1, \ldots, \ell_I\}$ satisfy

$$\sum_{i=1}^{I} 2^{-\ell_i} \leq 1 \tag{1}$$

Conversely, if the set $\{\ell_1, \ldots, \ell_I\}$ satisfy (1) then there exists a prefix code C with those codeword lengths.

Examples:

1. $C_1 = \{0001, 0010, 0100, 1000\}$ is prefix and $\sum_{i=1}^{4} 2^{-4} = \frac{1}{4} \leq 1$
2. $C_2 = \{0, 10, 110, 111\}$ is prefix and $\sum_{i=1}^{4} 2^{-\ell_i} = \frac{1}{2} + \frac{1}{4} + \frac{2}{8} = 1$
3. Lengths $\{1, 2, 2, 3\}$ give $\sum_{i=1}^{4} 2^{-\ell_i} = \frac{1}{2} + \frac{2}{4} + \frac{1}{8} > 1$ so no prefix code

Code Lengths and Probabilities

The *Kraft inequality* says that $\{\ell_1, \ldots, \ell_I\}$ are prefix code lengths **iff**

$$\sum_{i=1}^{I} 2^{-\ell_i} \leq 1$$

Probabilities from Code Lengths

Given code lengths $\ell = \{\ell_1, \ldots, \ell_I\}$ such that $\sum_{i=1}^{I} 2^{-\ell_i} \leq 1$ we define $\mathbf{q} = \{q_1, \ldots, q_I\}$ the **probabilities for** ℓ by

$$q_i \stackrel{\text{def}}{=} \frac{1}{z} 2^{-\ell_i} \quad \text{where} \quad z \stackrel{\text{def}}{=} \sum_i 2^{-\ell_i}$$ ensure that q_i satisfy $\sum_i q_i = 1$

Note: this implies $\ell_i = \log_2 \frac{1}{z q_i}$

Code Lengths and Probabilities

The *Kraft inequality* says that $\{\ell_1, \ldots, \ell_I\}$ are prefix code lengths **iff**

$$\sum_{i=1}^{I} 2^{-\ell_i} \leq 1$$

Probabilities from Code Lengths

Given code lengths $\ell = \{\ell_1, \ldots, \ell_I\}$ such that $\sum_{i=1}^{I} 2^{-\ell_i} \leq 1$ we define $\mathbf{q} = \{q_1, \ldots, q_I\}$ the **probabilities for** ℓ by

$$q_i \stackrel{\text{def}}{=} \frac{1}{z} 2^{-\ell_i} \quad \text{where} \quad z \stackrel{\text{def}}{=} \sum_i 2^{-\ell_i}$$ ensure that q_i satisfy $\sum_i q_i = 1$

Note: this implies $\ell_i = \log_2 \frac{1}{zq_i}$

Examples:

1. Lengths $\{1, 2, 2\}$ give $z = 1$ so $q_1 = \frac{1}{2}$, $q_2 = \frac{1}{4}$, and $q_3 = \frac{1}{4}$

Code Lengths and Probabilities

The *Kraft inequality* says that $\{\ell_1, \ldots, \ell_I\}$ are prefix code lengths **iff**

$$\sum_{i=1}^{I} 2^{-\ell_i} \leq 1$$

Probabilities from Code Lengths

Given code lengths $\ell = \{\ell_1, \ldots, \ell_I\}$ such that $\sum_{i=1}^{I} 2^{-\ell_i} \leq 1$ we define $\mathbf{q} = \{q_1, \ldots, q_I\}$ the **probabilities for** ℓ by

$$q_i \stackrel{\text{def}}{=} \frac{1}{z} 2^{-\ell_i} \quad \text{where} \quad z \stackrel{\text{def}}{=} \sum_i 2^{-\ell_i} \text{ ensure that } q_i \text{ satisfy } \sum_i q_i = 1$$

Note: this implies $\ell_i = \log_2 \frac{1}{zq_i}$

Examples:

1. Lengths $\{1, 2, 2\}$ give $z = 1$ so $q_1 = \frac{1}{2}$, $q_2 = \frac{1}{4}$, and $q_3 = \frac{1}{4}$
2. Lengths $\{2, 2, 3\}$ give $z = \frac{5}{8}$ so $q_1 = \frac{2}{5}$, $q_2 = \frac{2}{5}$, and $q_3 = \frac{1}{5}$

Minimising Expected Code Length

Given an ensemble X with probabilities $\mathcal{P}_X = \mathbf{p} = \{p_1, \ldots, p_I\}$ how can we minimise the expected code length?

Minimising Expected Code Length

Given an ensemble X with probabilities $\mathcal{P}_X = \mathbf{p} = \{p_1, \ldots, p_I\}$ how can we minimise the expected code length?

- Suppose we use code C with lengths $\ell = \{\ell_1, \ldots, \ell_I\}$ and corresponding probabilities $\mathbf{q} = \{q_1, \ldots, q_I\}$ with $q_i = \frac{1}{z} 2^{-\ell_i}$. Then,

$$
\begin{aligned}
L(C, X) &= \sum_i p_i \ell_i = \sum_i p_i \log_2 \left(\frac{1}{z q_i} \right) \\
&= \sum_i p_i \log_2 \left(\frac{1}{z p_i} \frac{p_i}{q_i} \right) \\
&= \sum_i p_i \left[\log_2 \left(\frac{1}{p_i} \right) + \log_2 \left(\frac{p_i}{q_i} \right) + \log_2 \left(\frac{1}{z} \right) \right] \\
&= \sum_i p_i \log_2 \frac{1}{p_i} + \sum_i p_i \log_2 \frac{p_i}{q_i} + \log_2 \left(\frac{1}{z} \right) \sum_i p_i \\
&= \quad H(X) \quad + \quad D(\mathbf{p} \| \mathbf{q}) \quad + \quad \log_2 \frac{1}{z} \quad 1
\end{aligned}
$$

Minimising Expected Code Length

- So if $\mathbf{q} = \{q_1, \ldots, q_I\}$ are the probabilities for the code lengths of C then under ensemble X with probabilities $\mathbf{p} = \{p_1, \ldots, p_I\}$

$$L(C, X) = H(X) + D(p\|q) + \log_2 \frac{1}{z}$$

- Thus, $L(C, X)$ is minimal (and equal to the entropy $H(X)$) if we can choose code lengths so that $D(\mathbf{p}\|\mathbf{q}) = 0$ and $\log_2 \frac{1}{z} = 0$
- But the *relative entropy* $D(\mathbf{p}\|\mathbf{q}) \geq 0$ with $D(\mathbf{p}\|\mathbf{q}) = 0$ iff $\mathbf{q} = \mathbf{p}$ (Gibb's inequality)
- For $\mathbf{q} = \mathbf{p}$, we have $z \stackrel{\text{def}}{=} \sum_i q_i = \sum_i p_i = 1$ and so $\log_2 \frac{1}{z} = 0$

Minimising Expected Code Length

- So if $\mathbf{q} = \{q_1, \ldots, q_I\}$ are the probabilities for the code lengths of C then under ensemble X with probabilities $\mathbf{p} = \{p_1, \ldots, p_I\}$

$$L(C, X) = H(X) + D(p\|q) + \log_2 \frac{1}{z}$$

- Thus, $L(C, X)$ is minimal (and equal to the entropy $H(X)$) if we can choose code lengths so that $D(\mathbf{p}\|\mathbf{q}) = 0$ and $\log_2 \frac{1}{z} = 0$
- But the *relative entropy* $D(\mathbf{p}\|\mathbf{q}) \geq 0$ with $D(\mathbf{p}\|\mathbf{q}) = 0$ iff $\mathbf{q} = \mathbf{p}$ (Gibb's inequality)
- For $\mathbf{q} = \mathbf{p}$, we have $z \stackrel{\text{def}}{=} \sum_i q_i = \sum_i p_i = 1$ and so $\log_2 \frac{1}{z} = 0$

We have shown that for a code C with lengths corresponding to $\mathbf{q}$

$$L(C, X) \geq H(X)$$

with equality only when C has code lengths $\ell_i = \log_2 \frac{1}{p_i}$

Shannon Codes

But $\log_2 \frac{1}{p_i}$ is not always an integer—a problem for code lengths!

Shannon Codes

But $\log_2 \frac{1}{p_i}$ is not always an integer—a problem for code lengths!

Shannon Code

Given an ensemble X with $\mathcal{P}_X = \{p_1, \ldots, p_I\}$ define[a] codelengths $\ell = \{\ell_1, \ldots, \ell_I\}$ by

$$\ell_i = \left\lceil \log_2 \frac{1}{p_i} \right\rceil \geq \log_2 \frac{1}{p_i}.$$

A code C is called a **Shannon code** if it has codelengths ℓ.

[a] Here $\lceil x \rceil$ is "smallest integer not smaller than x". e.g., $\lceil 2.1 \rceil = 3$, $\lceil 5 \rceil = 5$.

This gives us code lengths that are "closest" to $\log_2 \frac{1}{p_i}$

Examples:

1. If $\mathcal{P}_X = \{\frac{1}{2}, \frac{1}{4}, \frac{1}{4}\}$ then $\ell = \{1, 2, 2\}$ so $C = \{0, 10, 11\}$ is a Shannon code (in fact, this is an *optimal* code)
2. If $\mathcal{P}_X = \{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\}$ then $\ell = \{2, 2, 2\}$ with Shannon code $C = \{00, 10, 11\}$ (or $C = \{01, 10, 11\}$ ...)

Since $\lceil x \rceil$ is the *smallest* integer bigger than or equal to x it must be the case that $x \leq \lceil x \rceil \leq x + 1$.

Source Coding Theorem for Symbol Codes

Since $\lceil x \rceil$ is the *smallest* integer bigger than or equal to x it must be the case that $x \leq \lceil x \rceil \leq x + 1$.

Therefore, if we create a Shannon code C for $\mathbf{p} = \{p_1, \ldots, p_I\}$ with $\ell_i = \left\lceil \log_2 \frac{1}{p_i} \right\rceil \leq \log_2 \frac{1}{p_i} + 1$ it will satisfy

$$L(C, X) = \sum_i p_i \ell_i \leq \sum_i p_i \log_2 \frac{1}{p_i} + 1 = \sum_i p_i \log_2 \frac{1}{p_i} + \sum_i p_i$$
$$= H(X) + 1$$

Furthermore, since $\ell_i \geq -\log_2 p_i$ we have $2^{-\ell_i} \leq 2^{\log_2 p_i} = p_i$, so $\sum_i 2^{-\ell_i} \leq \sum_i p_i = 1$. By Kraft there is a *prefix code* with lengths ℓ_i

Source Coding Theorem for Symbol Codes

Since $\lceil x \rceil$ is the *smallest* integer bigger than or equal to x it must be the case that $x \leq \lceil x \rceil \leq x + 1$.

Therefore, if we create a Shannon code C for $\mathbf{p} = \{p_1, \ldots, p_I\}$ with $\ell_i = \left\lceil \log_2 \frac{1}{p_i} \right\rceil \leq \log_2 \frac{1}{p_i} + 1$ it will satisfy

$$L(C, X) = \sum_i p_i \ell_i \leq \sum_i p_i \log_2 \frac{1}{p_i} + 1 = \sum_i p_i \log_2 \frac{1}{p_i} + \sum_i p_i$$
$$= H(X) + 1$$

Furthermore, since $\ell_i \geq -\log_2 p_i$ we have $2^{-\ell_i} \leq 2^{\log_2 p_i} = p_i$, so $\sum_i 2^{-\ell_i} \leq \sum_i p_i = 1$. By Kraft there is a *prefix code* with lengths ℓ_i

Source Coding Theorem for Symbol Codes

For any ensemble X there exists a *prefix code* C such that

$$H(X) \leq L(C, X) \leq H(X) + 1.$$

Source Coding Theorem for Symbol Codes

Since $\lceil x \rceil$ is the *smallest* integer bigger than or equal to x it must be the case that $x \leq \lceil x \rceil \leq x + 1$.

Therefore, if we create a Shannon code C for $\mathbf{p} = \{p_1, \ldots, p_I\}$ with $\ell_i = \left\lceil \log_2 \frac{1}{p_i} \right\rceil \leq \log_2 \frac{1}{p_i} + 1$ it will satisfy

$$L(C, X) = \sum_i p_i \ell_i \leq \sum_i p_i \log_2 \frac{1}{p_i} + 1 = \sum_i p_i \log_2 \frac{1}{p_i} + \sum_i p_i$$
$$= H(X) + 1$$

Furthermore, since $\ell_i \geq -\log_2 p_i$ we have $2^{-\ell_i} \leq 2^{\log_2 p_i} = p_i$, so $\sum_i 2^{-\ell_i} \leq \sum_i p_i = 1$. By Kraft there is a *prefix code* with lengths ℓ_i

Source Coding Theorem for Symbol Codes

For any ensemble X there exists a *prefix code* C such that

$$H(X) \leq L(C, X) \leq H(X) + 1.$$

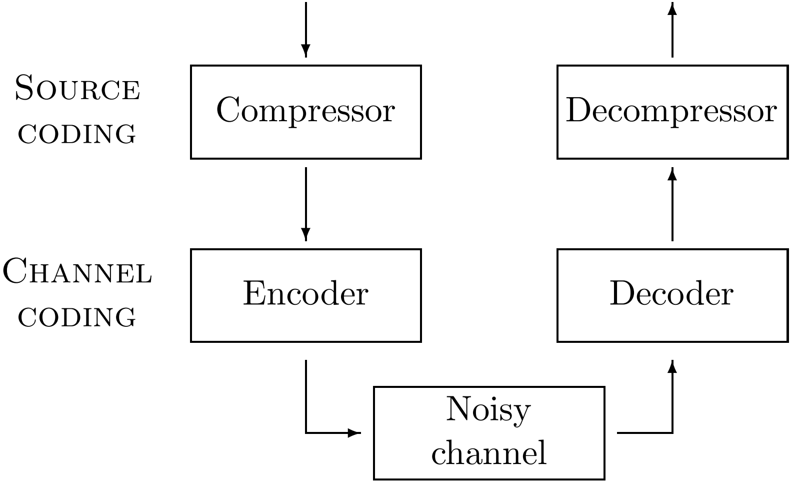
Shannon codes are "good" but not optimal — cf. Huffman coding

Outline

1. Inequalities

2. Key Results
   - The Source Coding Theorem for Lossy Uniform-Length Coding
   - The Source Coding Theorem for Lossless Variable-Length Coding
   - The Noisy-Channel Coding Theorem

The Big Picture



Channels

A **discrete channel** Q consists of an *input alphabet* $\mathcal{X} = \{a_1, \ldots, a_I\}$, an *output alphabet* $\mathcal{Y} = \{b_1, \ldots, b_J\}$ and *transistion probabilities* $P(y|x)$.
The channel Q can be expressed as a matrix

$$Q_{j,i} = P(y = b_j | x = a_i)$$

Channels

A **discrete channel** Q consists of an *input alphabet* $\mathcal{X} = \{a_1, \ldots, a_I\}$, an *output alphabet* $\mathcal{Y} = \{b_1, \ldots, b_J\}$ and *transistion probabilities* $P(y|x)$. The channel Q can be expressed as a matrix

$$Q_{j,i} = P(y = b_j | x = a_i)$$

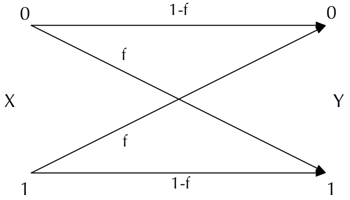
Example: A channel Q with inputs $\mathcal{X} = \{a_1, a_2, a_3\}$, outputs $\mathcal{Y} = \{b_1, b_2\}$, and transition probabilities expressed by the matrix

$$Q = \begin{bmatrix} 0.8 & 0.5 & 0.2 \\ 0.2 & 0.5 & 0.8 \end{bmatrix}$$

So $P(b_1|a_1) = 0.8 = P(b_2|a_3)$ and $P(b_1|a_2) = P(b_2|a_2) = 0.5$.

The Binary Symmetric Channel & The Z-Channel

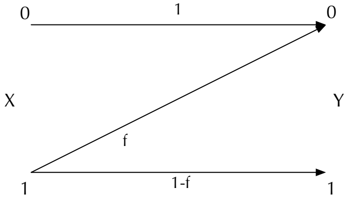
Binary Symmetric Channel



Inputs $\mathcal{X} = \{0, 1\}$; Outputs $\mathcal{Y} = \{0, 1\}$;
Transition probabilities with $P(\text{flip}) = f$

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

Z-Channel

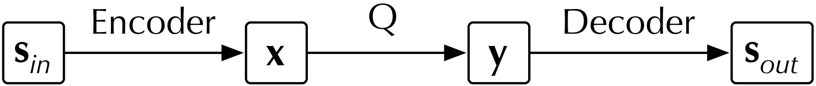


Inputs $\mathcal{X} = \{0, 1\}$; Outputs $\mathcal{Y} = \{0, 1\}$;
Transition probabilities

$$Q = \begin{bmatrix} 1 & f \\ 0 & 1-f \end{bmatrix}$$

Communicating over Noisy Channels

Suppose we know we have to communicate over some channel Q and we want build an *encoder/decoder* pair to reliably send a message **s** over Q.



Communicating over Noisy Channels

Suppose we know we have to communicate over some channel Q and we want build an *encoder/decoder* pair to reliably send a message $\mathbf{s}$ over Q.

$$\boxed{\mathbf{s}_{in}} \xrightarrow{\text{Encoder}} \boxed{\mathbf{x}} \xrightarrow{Q} \boxed{\mathbf{y}} \xrightarrow{\text{Decoder}} \boxed{\mathbf{s}_{out}}$$

Reliability is measured via **probability of error** — that is, the probability of incorrectly decoding $\mathbf{s}_{out}$ given $\mathbf{s}_{in}$ as input:

$$P(\mathbf{s}_{out} \neq \mathbf{s}_{in}) = \sum_{\mathbf{s}} P(\mathbf{s}_{out} \neq \mathbf{s}_{in} | \mathbf{s}_{in}) P(\mathbf{s}_{in})$$

Communicating over Noisy Channels

Suppose we know we have to communicate over some channel Q and we want build an *encoder/decoder* pair to reliably send a message $\mathbf{s}$ over Q.

$$\boxed{\mathbf{s}_{in}} \xrightarrow{\text{Encoder}} \boxed{\mathbf{x}} \xrightarrow{\quad Q \quad} \boxed{\mathbf{y}} \xrightarrow{\text{Decoder}} \boxed{\mathbf{s}_{out}}$$

Reliability is measured via **probability of error** — that is, the probability of incorrectly decoding $\mathbf{s}_{out}$ given $\mathbf{s}_{in}$ as input:

$$P(\mathbf{s}_{out} \neq \mathbf{s}_{in}) = \sum_{\mathbf{s}} P(\mathbf{s}_{out} \neq \mathbf{s}_{in} | \mathbf{s}_{in}) P(\mathbf{s}_{in})$$

Example:

Let $\mathcal{S} = \{\mathtt{a}, \mathtt{b}\}$, with *encoder*: $\mathtt{a} \to \mathtt{0}$; $\mathtt{b} \to \mathtt{1}$, *decoder*: $\mathtt{0} \to \mathtt{a}$; $\mathtt{1} \to \mathtt{b}$.
For binary symmetric Q with $f = 0.1$ and $(p_{\mathtt{a}}, p_{\mathtt{b}}) = (0.5, 0.5)$

$$P(\mathbf{s}_{in} \neq \mathbf{s}_{out}) = P(y = 1 | x = 0)\, p_{\mathtt{a}} + P(y = 0 | x = 1)\, p_{\mathtt{b}} = f = 0.1$$

A Simple Coding Scheme

Suppose $\mathbf{s} \in \{\mathtt{a}, \mathtt{b}\}$ and we encode by $\mathtt{a} \rightarrow \mathtt{000}$ and $\mathtt{b} \rightarrow \mathtt{111}$.
To decode we count the number of 1s and 0s and set all bits to the majority count to determine $\mathbf{s}$

$$\underbrace{\mathtt{000}, \mathtt{001}, \mathtt{010}, \mathtt{100}}_{A} \rightarrow \mathtt{a} \quad \text{and} \quad \underbrace{\mathtt{111}, \mathtt{110}, \mathtt{101}, \mathtt{011}}_{B} \rightarrow \mathtt{b}$$

A Simple Coding Scheme

Suppose $\mathbf{s} \in \{\mathtt{a}, \mathtt{b}\}$ and we encode by $\mathtt{a} \to 000$ and $\mathtt{b} \to 111$.
To decode we count the number of 1s and 0s and set all bits to the majority count to determine $\mathbf{s}$

$$\underbrace{000, 001, 010, 100}_{A} \to \mathtt{a} \quad \text{and} \quad \underbrace{111, 110, 101, 011}_{B} \to \mathtt{b}$$

If the channel Q is binary symmetric with $f = 0.1$ again

$$\begin{aligned} P(\mathbf{s}_{in} \neq \mathbf{s}_{out}) &= P(\mathbf{y} \in B|000)\, p_{\mathtt{a}} + P(\mathbf{y} \in A|111)\, p_{\mathtt{b}} \\ &= [f^3 + 3f^2(1-f)]p_{\mathtt{a}} + [f^3 + 3f^2(1-f)]p_{\mathtt{b}} \\ &= f^3 + 3f^2(1-f) = 0.028 \end{aligned}$$

A Simple Coding Scheme

Suppose $\mathbf{s} \in \{\mathtt{a}, \mathtt{b}\}$ and we encode by $\mathtt{a} \rightarrow \mathtt{000}$ and $\mathtt{b} \rightarrow \mathtt{111}$.
To decode we count the number of 1s and 0s and set all bits to the majority count to determine $\mathbf{s}$

$$\underbrace{\mathtt{000}, \mathtt{001}, \mathtt{010}, \mathtt{100}}_{A} \rightarrow \mathtt{a} \quad \text{and} \quad \underbrace{\mathtt{111}, \mathtt{110}, \mathtt{101}, \mathtt{011}}_{B} \rightarrow \mathtt{b}$$

If the channel Q is binary symmetric with $f = 0.1$ again

$$\begin{aligned} P(\mathbf{s}_{in} \neq \mathbf{s}_{out}) &= P(\mathbf{y} \in B | 000)\, p_{\mathrm{a}} + P(\mathbf{y} \in A | 111)\, p_{\mathrm{b}} \\ &= [f^3 + 3f^2(1-f)] p_{\mathrm{a}} + [f^3 + 3f^2(1-f)] p_{\mathrm{b}} \\ &= f^3 + 3f^2(1-f) = 0.028 \end{aligned}$$

So the *error* has dropped from 0.1 to 0.028
but so has the *rate*: from 1 symbol/bit to 1/3 symbol/bit.

A Simple Coding Scheme

Suppose $\mathbf{s} \in \{\mathtt{a}, \mathtt{b}\}$ and we encode by $\mathtt{a} \to \mathtt{000}$ and $\mathtt{b} \to \mathtt{111}$.
To decode we count the number of 1s and 0s and set all bits to the majority count to determine $\mathbf{s}$

$$\underbrace{\mathtt{000}, \mathtt{001}, \mathtt{010}, \mathtt{100}}_{A} \to \mathtt{a} \quad \text{and} \quad \underbrace{\mathtt{111}, \mathtt{110}, \mathtt{101}, \mathtt{011}}_{B} \to \mathtt{b}$$

If the channel Q is binary symmetric with $f = 0.1$ again

$$\begin{aligned} P(\mathbf{s}_{in} \neq \mathbf{s}_{out}) &= P(\mathbf{y} \in B|000)\, p_{\mathtt{a}} + P(\mathbf{y} \in A|111)\, p_{\mathtt{b}} \\ &= [f^3 + 3f^2(1-f)]p_{\mathtt{a}} + [f^3 + 3f^2(1-f)]p_{\mathtt{b}} \\ &= f^3 + 3f^2(1-f) = 0.028 \end{aligned}$$

So the *error* has dropped from 0.1 to 0.028
but so has the *rate*: from 1 symbol/bit to 1/3 symbol/bit.

Can we make the error arbitrarily small without the rate going to zero?

Channel Capacity

A key quantity when using a channel is the mutual information between its inputs X and outputs Y:

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

This measures how much what was received *reduces uncertainty* about what was transmitted.

Channel Capacity

A key quantity when using a channel is the mutual information between its inputs X and outputs Y:

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

This measures how much what was received *reduces uncertainty* about what was transmitted.

Examples (See MacKay §9.5)

- For noiseless channel $H(X|Y) = 0$ so $I(X;Y) = H(X)$.
  If $\mathbf{p}_X = (0.9, 0.1)$ then $I(X;Y) = 0.47$ bits.
- For binary symmetric channel with $f = 0.15$ and $\mathbf{p}_X$ as above we have $H(Y) = 0.76$ and $H(Y|X) = 0.61$ so $I(X;Y) = 0.15$ bits
- For Z channel with $f = 0.15$ and same $\mathbf{p}_X$ we have $H(Y) = 0.42$, $H(Y|X) = 0.061$ so $I(X;Y) = 0.36$ bits

Channel Capacity

A key quantity when using a channel is the mutual information between its inputs X and outputs Y:

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

This measures how much what was received *reduces uncertainty* about what was transmitted.

Examples (See MacKay §9.5)

- For noiseless channel $H(X|Y) = 0$ so $I(X;Y) = H(X)$.
  If $\mathbf{p}_X = (0.9, 0.1)$ then $I(X;Y) = 0.47$ bits.
- For binary symmetric channel with $f = 0.15$ and $\mathbf{p}_X$ as above we have $H(Y) = 0.76$ and $H(Y|X) = 0.61$ so $I(X;Y) = 0.15$ bits
- For Z channel with $f = 0.15$ and same $\mathbf{p}_X$ we have $H(Y) = 0.42$, $H(Y|X) = 0.061$ so $I(X;Y) = 0.36$ bits

So, intuitively, the reliability is "noiseless > Z > symmetric"

Channel Capacity

The mutual information measure for a channel depends on the choice of input distribution $\mathbf{p}_X$. If $H(X)$ is small then $I(X;Y) \leq H(X)$ is small.
The *largest possible* reduction in uncertainty achievable across a channel is its **capacity**.

Channel Capacity

The capacity C of a channel Q is the largest mutual information between its input and output for any choice of input ensemble. That is,

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

Example: For binary symmetric channel ($f = .15$), $I(X;Y)$ is maximal for $\mathbf{p}_X = (0.5, 0.5)$, so $C = 0.39$ bits (cf. $I(X;Y) = 0.15$ for $\mathbf{p}_X = (0.9, 0.1)$)

Block Codes

We now formalise codes that make repeated use of a noisy channel to communicate a predefined set of S messages.
Each $s \in \{1, 2, \ldots, S\}$ is paired with a unique *block* of symbols $\mathbf{x} \in \mathcal{X}^N$.

(N, K) Block Code

Given a channel Q with inputs $\mathcal{X}$ and outputs $\mathcal{Y}$, an integer $N > 0$, and $K > 0$, an (N, K) Block Code for Q is a list of $S = 2^K$ codewords

$$\mathcal{S} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(2^K)}\}$$

where each $\mathbf{x}^{(s)} \in \mathcal{X}^N$ consists of N symbols from $\mathcal{X}$. The **rate** of such a block code is K/N bits per channel use.

Examples (for Binary Symmetric Channel Q)

- A $(1, 1)$ block code: $\mathcal{S} = \{0, 1\}$ — Rate: 1
- A $(3, 2)$ block code: $\mathcal{S} = \{000, 001, 100, 111\}$ — Rate: $\frac{2}{3}$
- A $(3, \log_2 3)$ block code: $\mathcal{S} = \{001, 010, 100\}$ — Rate: $\frac{\log_2 3}{3} \approx 0.53$

Decoding Block Codes

An (N, K) block code sends each message $s \in \{1, 2, \ldots, 2^K\}$ over a channel Q as $\mathbf{x}^s \in \mathcal{X}^N$ and the block $\mathbf{y} \in \mathcal{Y}^N$ is received. How does the receiver determine which s was transmitted?

Block Decoder

A **decoder** for a (N, K) block code is a mapping that associates each $\mathbf{y} \in \mathcal{Y}^N$ with an $\hat{s} \in \{1, 2, \ldots, 2^K\}$.

Example The $(2, 1)$ block code $\mathcal{S} = \{000, 111\}$ and majority vote decoder $d : \{0, 1\}^3 \to \{1, 2\}$ defined by

$$d(000) = d(001) = d(010) = d(100) = 1$$
$$d(111) = d(110) = d(101) = d(011) = 2$$

Reliability

Want an *encoder/decoder* pair to reliably send a messages over channel Q.

$$\mathbf{s}_{in} \xrightarrow{\text{Encoder}} \mathbf{x} \xrightarrow{\quad Q \quad} \mathbf{y} \xrightarrow{\text{Decoder}} \mathbf{s}_{out}$$

Probability of (Block) Error

Given a channel Q the **probability of (block) error** for a code is

$$p_B = P(\mathbf{s}_{out} \neq \mathbf{s}_{in}) = \sum_{\mathbf{s}_{in}} P(\mathbf{s}_{out} \neq \mathbf{s}_{in} | \mathbf{s}_{in}) P(\mathbf{s}_{in})$$

and its **maximum probability of (block) error** is

$$p_{BM} = \max_{\mathbf{s}_{in}} P(\mathbf{s}_{out} \neq \mathbf{s}_{in} | \mathbf{s}_{in})$$

As $P(\mathbf{s}_{out} \neq \mathbf{s}_{in} | \mathbf{s}_{in}) \leq p_{BM}$ for all $\mathbf{s}_{in}$ we get $p_B \leq \sum_{\mathbf{s}_{in}} p_{BM} P(\mathbf{s}_{in}) = p_{BM}$ and so if $p_{BM} \to 0$ then $p_B \to 0$.

Achievable Rates

If it is possible to construct codes with rate R for a channel that can have arbitrarily small error the rate R is said to be *achievable*. Formally:

Achievable Rate

A rate R over a channel Q is said to be **achievable** if, for any $\epsilon > 0$ there is a (N, K) block code and decoder such that its rate $K/N \geq R$ and its maximum probability of block error satisfies

$$p_{BM} = \max_{\mathbf{s}_{in}} P(\mathbf{s}_{out} \neq \mathbf{s}_{in} | \mathbf{s}_{in}) < \epsilon$$

The main "trick" to minimising p_{BM} is to construct a (N, K) block code with (almost) **non-confusable** codes. That is, a code such that the set of $\mathbf{y}$ that each $\mathbf{x}^{(s)}$ are sent to by Q have low probability intersection.

The Noisy-Channel Coding Theorem

Informal Statement

Noisy-Channel Coding Theorem (Brief)

If Q is a channel with capacity C then the rate R is *achievable* **if and only if** $R \leq C$, that is, the rate is no greater than the channel capacity.

The Noisy-Channel Coding Theorem

Informal Statement

Noisy-Channel Coding Theorem (Brief)

If Q is a channel with capacity C then the rate R is *achievable* **if and only if** $R \leq C$, that is, the rate is no greater than the channel capacity.

Example:

- We saw that BSC Q with $f = 0.15$ has capacity $C = 0.39$ bits.
- Suppose we want error less than $\epsilon = 0.05$ and rate $R > 0.25$
- The NCCT tells us there should be, for N large enough, an (N, K) code with $K/N \geq 0.25$

Indeed, we showed the code $\mathcal{S} = \{000, 111\}$ with majority vote decoder has probability of error $0.028 < 0.05$ for Q and rate $1/3 > 0.25$.

Noisy-Channel Coding Theorem (Brief)

If Q is a channel with capacity C then the rate R is *achievable* **if and only if** $R \leq C$, that is, the rate is no greater than the channel capacity.

Example:

- We saw that BSC Q with $f = 0.15$ has capacity $C = 0.39$ bits.
- Suppose we want error less than $\epsilon = 0.05$ and rate $R > 0.25$
- The NCCT tells us there should be, for N large enough, an (N, K) code with $K/N \geq 0.25$

Indeed, we showed the code $\mathcal{S} = \{000, 111\}$ with majority vote decoder has probability of error $0.028 < 0.05$ for Q and rate $1/3 > 0.25$.

- For $N = 3$ there is a $(3, 1)$ code meeting the requirements.
- But there is *no code* with arbitrarily small ϵ and rate $1/2 > 0.39 = C$.

Summary

Inequalities

- Probabilistic: Markov, Chebyshev, Law of Large Numbers
- Information Theoretic: Gibbs, "Data doesn't hurt", Data-Processing
  - (Aside: All driven by concavity of entropy)

Main Results

- Source Coding Theorems
  - *For Lossy Block Coding*: Reliability/compression trade-off is asymptotically controlled by entropy of source.
  - *For Lossless Variable-Length Coding*: Can always find code with expected size within 1 bit of entropy of source
- Noisy-Channel Coding Theorem
  - The trade-off between reliability and rate of communication over a noisy channel is determined by capacity of channel (i.e., maxmimum mutual information between input and output).