# Notes for ELEMENTS OF INFORMATION THEORY

## CHAPTER 1 INTRODUCTION AND PREVIEW

Information theory answers two fundamental questions in communication theory: What is the ultimate data compression (answer: the entropy $H$), and what is the ultimate transmission rate of communication (answer: the channel capacity $C$).

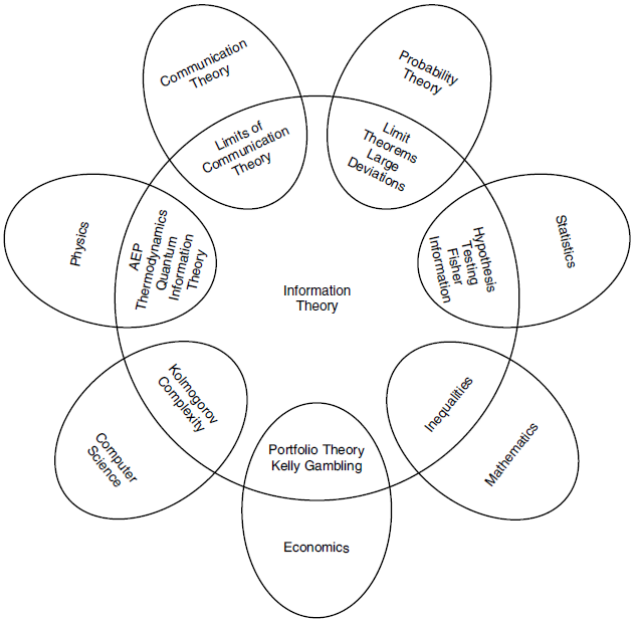



Figure 1. 1 Relationship of information theory to other fields.

**Electrical Engineering (Communication Theory).**

1.1 PREVIEW OF THE BOOK

## CHAPTER 2 ENTROPY, RELATIVE ENTROPY, AND MUTUAL INFORMATION

For any probability distribution, we define a quantity called the ***entropy***, which has many properties that agree with the intuitive notion of what a measure of information should be. This notion is extended to define ***mutual information***, which is a measure of the amount of information one random variable contains about another. **Entropy** then becomes the self-information of a random variable. **Mutual information** is a special case of a more general quantity called **relative entropy**, which is a measure of the **distance** between two probability distributions.

2.1 ENTROPY

We first introduce the concept of *entropy*, which is a measure of the uncertainty of a random variable. Let $X$ be a discrete random variable with **alphabet** $\mathcal{X}$ and probability mass function $p(x) = \Pr\{X = x\}$, $x \in \mathcal{X}$. We denote the probability mass function by $p(x)$ rather than $p_X(x)$, for convenience.

**Definition** The entropy $H(X)$ of a discrete random variable $X$ is defined by

$$H(X) = -\sum_{x\in\mathcal{X}} p(x)\log p(x).$$

We also write $H(p)$ for the above quantity. The log is to the base 2 and entropy is expressed in **bits**. For example, the entropy of a fair coin toss is 1 bit. We will use the convention that $0\log 0 = 0$, which is easily justified by continuity since $x\log x \to 0$ as $x \to 0$.

If the base of the logarithm is $b$, we denote the entropy as $H_b(X)$. If the base of the logarithm is $e$, the entropy is measured in *nats*. Unless otherwise specified, we will take all logarithms to base 2, and hence all the entropies will be measured in bits. Note that entropy is a function of the distribution of $X$. It does not depend on the actual values taken by the random variable $X$, but only on the probabilities.

We denote expectation by $\mathrm{E}$. Thus, if $X\sim p(x)$, the expected value of the random variable $g(X)$ is written

$$\mathrm{E}_p[g(X)] = \sum_{x\in\mathcal{X}} g(x)p(x),$$

**Remark** The entropy of $X$ can **also** be interpreted as the expected value of the random variable $\log\frac{1}{p(X)}$, (bit->**Information of each symbol**) where $X$ is drawn according to probability mass function $p(x)$. Thus,

$$H(X) = \mathrm{E}_p\left[\log\frac{1}{p(X)}\right].$$

First, we derive some immediate consequences of the definition.

**Lemma 2.1.1** $H(X) \geq 0$.

**Lemma 2.1.2** $H_b(X) = (\log_b a)H_a(X)$.

**Example 2.1.1** Let

$$X = \begin{cases} 1 & \text{with probability } p, \\ 0 & \text{with probability } 1-p. \end{cases}$$

Then

$$H(X) = -p\log p - (1-p)\log(1-p) \overset{\text{def}}{=} H(p).$$

In particular, $H(X) = 1$ bit when $p = 1/2$. The graph of the function $H(p)$ is shown in Figure 2.1.

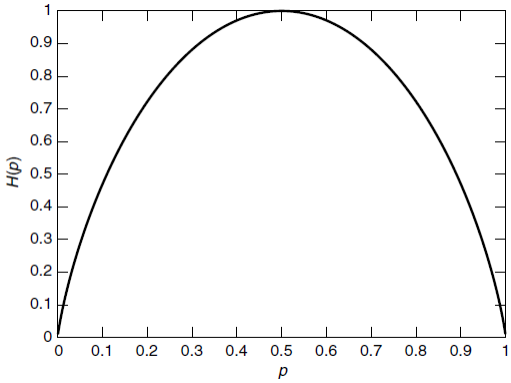


Figure 2. 1 $H(p)$ vs. $p$.

**Example 2.1.2** Let

$$X = \begin{cases} a & \text{with probability } 1/2, \\ b & \text{with probability } 1/4, \\ c & \text{with probability } 1/8, \\ d & \text{with probability } 1/8. \end{cases}$$

The entropy of $X$ is

$$H(X) = -\frac{1}{2}\log\frac{1}{2} - \frac{1}{4}\log\frac{1}{4} - \frac{1}{8}\log\frac{1}{8} - \frac{1}{8}\log\frac{1}{8} = \frac{7}{4} \text{ bits.}$$

2.2 JOINT ENTROPY AND CONDITIONAL ENTROPY

We now extend the definition to a pair of random variables. There is nothing really new in this definition because $(X, Y)$ can be considered to be a single vector-valued random variable.

**Definition** The joint entropy $H(X, Y)$ of a pair of discrete random variables $(X, Y)$ with a joint distribution $p(x, y)$ is defined as

$$H(X, Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x, y)\log p(x, y),$$

which can also be expressed as

$$H(X, Y) = -\mathrm{E}[\log p(x, y)].$$

We also define the conditional entropy of a random variable given another as the expected value of the entropies of the conditional distributions, averaged over the conditioning random variable.

**Definition** If $(X, Y)\sim p(x, y)$, the conditional entropy $H(Y|X)$ is defined as

$$\begin{aligned} H(Y|X) &= \sum_{x\in\mathcal{X}} p(x)H(Y|X = x) \\ &= -\sum_{x\in\mathcal{X}} p(x)\sum_{y\in\mathcal{Y}} p(y|x)\log p(y|x) \\ &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x, y)\log p(y|x) \\ &= -\mathrm{E}[\log p(Y|X)]. \end{aligned}$$

The naturalness of the definition of joint entropy and conditional entropy is exhibited by the fact that the entropy of a pair of random variables is the entropy of one plus the conditional entropy of the other.

**Theorem 2.2.1** (**Chain rule**)

$$H(X, Y) = H(X) + H(Y|X).$$

**Proof**

$$\begin{aligned} H(X, Y) &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x, y)\log p(x, y) \\ &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x, y)\log p(x)p(y|x) \end{aligned}$$

$$= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(x) - \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(y|x)$$
$$= -\sum_{x\in\mathcal{X}} p(x)\log p(x) - \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(y|x)$$
$$= H(X) + H(Y|X).$$

Equivalently, we can write

$$\log p(X,Y) = \log p(X) + \log p(Y|X)$$

and take the expectation of both sides of the equation to obtain the theorem above.

**Corollary**

$$H(X,Y|Z) = H(X|Z) + H(Y|X,Z).$$

**Example 2.2.1** Let $(X,Y)$ have the following joint distribution:

| $Y$ \ $X$ | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | $\frac{1}{8}$ | $\frac{1}{16}$ | $\frac{1}{32}$ | $\frac{1}{32}$ |
| 2 | $\frac{1}{16}$ | $\frac{1}{8}$ | $\frac{1}{32}$ | $\frac{1}{32}$ |
| 3 | $\frac{1}{16}$ | $\frac{1}{16}$ | $\frac{1}{16}$ | $\frac{1}{16}$ |
| 4 | $\frac{1}{4}$ | 0 | 0 | 0 |

The marginal distribution of $X$ is $\left(\frac{1}{2},\frac{1}{4},\frac{1}{8},\frac{1}{8}\right)$ and the marginal distribution of $Y$ is $\left(\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4}\right)$, and hence $H(X) = \frac{7}{4}$ bits and $H(Y) = 2$ bits. Also,

$$H(X|Y) = \sum_{i=1}^{4} p(Y=i)H(X|Y=i)$$
$$= \frac{1}{4}H\left(\frac{1}{8}\times\frac{1}{1/4},\frac{1}{16}\times\frac{1}{1/4},\frac{1}{32}\times\frac{1}{1/4},\frac{1}{32}\times\frac{1}{1/4}\right) + \frac{1}{4}H\left(\frac{1}{4},\frac{1}{2},\frac{1}{8},\frac{1}{8}\right)$$
$$+\frac{1}{4}H\left(\frac{1}{4},\frac{1}{4},\frac{1}{4},\frac{1}{4}\right) + \frac{1}{4}H(1,0,0,0)$$
$$= \frac{1}{4}\times\frac{7}{4} + \frac{1}{4}\times\frac{7}{4} + \frac{1}{4}\times 2 + \frac{1}{4}\times 0$$
$$= \frac{11}{8} \text{ bits.}$$

Similarly, $H(Y|X) = \frac{13}{8}$ bits and $H(X,Y) = \frac{27}{8}$ bits.

**Remark** Note that $H(Y|X) \neq H(X|Y)$.

2.3 RELATIVE ENTROPY AND MUTUAL INFORMATION

The entropy of a random variable is a measure of the uncertainty of the random variable; it is a measure of the amount of information required on the average to describe the random variable.

The ***relative entropy*** is a measure of the distance between two distributions. In statistics, it arises as an expected logarithm of the likelihood ratio. The relative entropy $D(p \parallel q)$ is a measure of the **inefficiency** of assuming that the distribution is $q$ when the true distribution is $p$. For example, if we knew the true distribution $p$ of the random variable, we could construct **a code** with average description length $H(p)$. If, **instead**, we used **the code** for a distribution $q$, we would **need** $H(p) + D(p \parallel q)$ bits on the average to describe the random variable.

**Definition** The *relative entropy* or *Kullback–Leibler distance* between two probability mass functions $p(x)$ and $q(x)$ is defined as

$$D(p \parallel q) = \sum_{x\in\mathcal{X}} p(x)\log\frac{p(x)}{q(x)}$$
$$= \mathrm{E}_p\left[\log\frac{p(X)}{q(X)}\right].$$

In the above definition, we use the convention that $0\log\frac{0}{0} = 0$ and the convention (based on continuity

arguments) that $0 \log \frac{0}{q} = 0$ and $p \log \frac{p}{0} = \infty$.

We will soon show that **relative entropy** is **always nonnegative** and is zero **if and only if** $p = q$. However, it is not a true distance between distributions since it is not symmetric and does not satisfy the triangle inequality. **Nonetheless**, it is often useful to think of relative entropy as a "distance" between distributions.

We now introduce mutual information, which is a measure of the amount of information that one random variable contains about another random variable. It is the **reduction** in the uncertainty of one random variable due to the knowledge of the other.

**Definition** Consider two random variables $X$ and $Y$ with a joint probability mass function $p(x,y)$ and marginal probability mass functions $p(x)$ and $p(y)$. The ***mutual information*** $I(X;Y)$ is the **relative entropy** between the joint distribution and the product distribution $p(x)p(y)$:

$$\begin{aligned} I(X;Y) &= \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \\ &= D\big(p(x,y) \parallel p(x)p(y)\big) \\ &= \mathrm{E}_{p(x,y)} \left[ \log \frac{p(X,Y)}{p(X)p(Y)} \right]. \end{aligned}$$

**Example 2.3.1** Let $\mathcal{X} = \{0,1\}$ and consider two distributions $p$ and $q$ on $\mathcal{X}$. Let $p(0) = 1 - r$, $p(1) = r$, and let $q(0) = 1 - s$, $q(1) = s$. Then

$$D(p \parallel q) = (1-r) \log \frac{1-r}{1-s} + r \log \frac{r}{s}$$

and

$$D(q \parallel p) = (1-s) \log \frac{1-s}{1-r} + s \log \frac{s}{r}.$$

If $r = s$, then $D(p \parallel q) = D(q \parallel p) = 0$. If $r = \frac{1}{2}$, $s = \frac{1}{4}$, we can calculate

$$D(p \parallel q) = 0.2075 \text{ bit},$$

whereas

$$D(q \parallel p) = 0.1887 \text{ bit}.$$

Note that $D(p \parallel q) \neq D(q \parallel p)$ in general.

2.4 RELATIONSHIP BETWEEN ENTROPY AND MUTUAL INFORMATION

We can rewrite the definition of mutual information $I(X;Y)$ as

$$\begin{aligned} I(X;Y) = D\big(p(x,y) \parallel p(x)p(y)\big) &= \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \\ &= \sum_{x,y} p(x,y) \log \frac{p(x|y)}{p(x)} \\ &= -\sum_{x,y} p(x,y) \log p(x) + \sum_{x,y} p(x,y) \log p(x|y) \\ &= -\sum_{x} p(x) \log p(x) - \left( -\sum_{x,y} p(x,y) \log p(x|y) \right) \\ &= H(X) - H(X|Y). \end{aligned}$$

Thus, the mutual information $I(X;Y)$ is the reduction in the uncertainty of $X$ due to the knowledge of $Y$.

By symmetry, it also follows that

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

Thus, $X$ says as much about $Y$ as $Y$ says about $X$.

Since $H(X,Y) = H(X) + H(Y|X)$, we have

$$I(X;Y) = H(X) + H(Y) - H(X,Y).$$

Finally, we note that

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

Thus, the mutual information of a random variable with itself is the entropy of the random variable. This is

the reason that entropy is sometimes referred to as self-information.

**Theorem 2.4.1** (*Mutual information and entropy*)

$$I(X;Y) = H(X) - H(X|Y)$$
$$I(X;Y) = H(Y) - H(Y|X)$$
$$I(X;Y) = H(X) + H(Y) - H(X,Y)$$
$$I(X;Y) = I(Y;X)$$
$$I(X;X) = H(X).$$

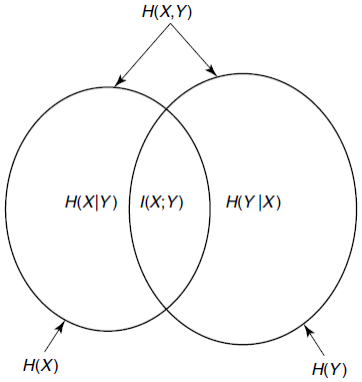



Figure 2. 2 Relationship between entropy and mutual information.

The relationship between $H(X)$, $H(Y)$, $H(X,Y)$, $H(X|Y)$, $H(Y|X)$, and $I(X;Y)$ is expressed in a Venn diagram (Figure 2.2).

2.5 CHAIN RULES FOR ENTROPY, RELATIVE ENTROPY, AND MUTUAL INFORMATION

We now show that the entropy of a collection of random variables is the sum of the conditional entropies.

**Theorem 2.5.1** (*Chain rule for entropy*) Let $X_1, X_2, \cdots, X_n$ be drawn according to $p(x_1, x_2, \cdots, x_n)$. Then

$$H(X_1, X_2, \cdots, X_n) = \sum_{i=1}^{n} H(X_i|X_{i-1}, \cdots, X_1).$$

We now define the conditional mutual information as the reduction in the uncertainty of $X$ due to knowledge of $Y$ when $Z$ is given.

**Definition** The *conditional mutual information* of random variables $X$ and $Y$ given $Z$ is defined by

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z)$$
$$= E_{p(x,y,z)}\left[\log \frac{p(X,Y|Z)}{p(X|Z)p(Y|Z)}\right].$$

Mutual information also satisfies a chain rule.

**Theorem 2.5.2** (*Chain rule for information*)

$$I(X_1, X_2, \cdots, X_n; Y) = \sum_{i=1}^{n} I(X_i; Y|X_{i-1}, X_{i-2}, \cdots, X_1).$$

We define a conditional version of the relative entropy.

**Definition** For joint probability mass functions $p(x,y)$ and $q(x,y)$, the *conditional relative entropy* $D\big(p(y|x) \parallel q(y|x)\big)$ is the average of the relative entropies between the conditional probability mass functions $p(y|x)$ and $q(y|x)$ averaged over the probability mass function $p(x)$. More precisely,

$$D\big(p(y|x) \parallel q(y|x)\big) = \sum_{x} p(x) \sum_{y} p(y|x) \log \frac{p(y|x)}{q(y|x)}$$
$$= E_{p(x,y)}\left[\log \frac{p(Y|X)}{q(Y|X)}\right].$$

The notation for conditional relative entropy is not explicit since it omits mention of the distribution $p(x)$ of the conditioning random variable. However, it is normally understood from the context.

The relative entropy between two joint distributions on a pair of random variables can be expanded as the

sum of a relative entropy and a conditional relative entropy.

**Theorem 2.5.3** (*Chain rule for relative entropy*)

$$D\big(p(x,y) \parallel q(x,y)\big) = D\big(p(x) \parallel q(x)\big) + D\big(p(y|x) \parallel q(y|x)\big).$$

2.6 JENSEN'S INEQUALITY AND ITS CONSEQUENCES

**Definition** A function $f(x)$ is said to be *convex* over an interval $(a,b)$ if for every $x_1, x_2 \in (a,b)$ and $0 \le \lambda \le 1$,

$$f(\lambda x_1 + (1-\lambda)x_2) \le \lambda f(x_1) + (1-\lambda) f(x_2).$$

A function $f$ is said to be *strictly convex* if equality holds only if $\lambda = 0$ or $\lambda = 1$.

**Definition** A function $f$ is *concave* if $-f$ is convex. A function is convex if it always lies below any chord. A function is concave if it always lies above any chord.

Examples of convex functions include $x^2$, $|x|$, $e^x$, $x \log x$ (for $x \ge 0$), and so on. Examples of concave functions include $\log x$ and $\sqrt{x}$ for $x \ge 0$. Figure 2.3 shows some examples of convex and concave functions. Note that **linear functions** $ax + b$ are both convex and concave.

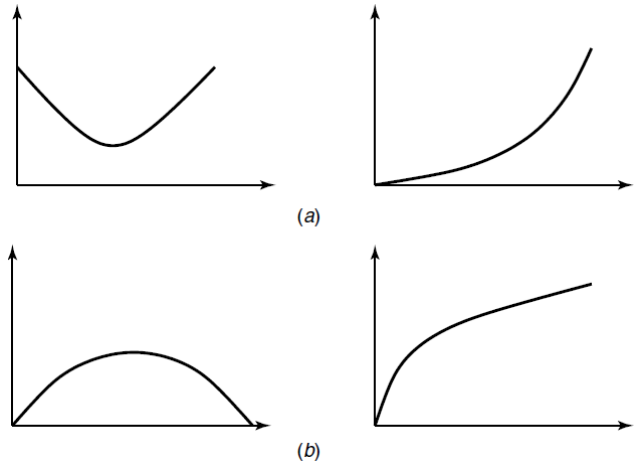



Figure 2. 3 Examples of (a) convex and (b) concave functions.

**Theorem 2.6.1** If the function $f$ has a second derivative that is nonnegative (positive) over an interval, the function is convex (strictly convex) over that interval.

**Theorem 2.6.2** (*Jensen's inequality*) If $f$ is a convex function and $X$ is a random variable,

$$\mathrm{E}[f(X)] \ge f(\mathrm{E}[X]).$$

Moreover, if $f$ is strictly convex, the equality above implies that $X = E[X]$ with probability 1 (i.e., $X$ is a constant).

**Theorem 2.6.3** (*Information inequality*) Let $p(x)$, $q(x)$, $x \in \mathcal{X}$, be two probability mass functions. Then

$$D(p \parallel q) \ge 0$$

with equality if and only if $p(x) = q(x)$ for all $x$. (See exercise 2.26)

**Corollary** (*Nonnegativity of mutual information*) For any two random variables, $X$, $Y$,

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

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

**Corollary**

$$D\big(p(y|x) \parallel q(y|x)\big) \ge 0,$$

with equality if and only if $p(y|x) = q(y|x)$ for all $y$ and $x$ such that $p(x) > 0$.

**Corollary**

$$I(X;Y|Z) \ge 0,$$

with equality if and only if $X$ and $Y$ are conditionally independent given $Z$.

We now show that the **uniform distribution** over the range $\mathcal{X}$ is the **maximum** entropy **distribution** over this range. It follows that any random variable with this range has an entropy no greater than $\log|\mathcal{X}|$.

**Theorem 2.6.4** $H(X) \le \log|\mathcal{X}|$, where $|\mathcal{X}|$ denotes the number of elements in the range of $X$, with equality

if and only if $X$ has a uniform distribution over $\mathcal{X}$. (for binary distribution, we can use derivative to find global maxima)

**Proof**:

Let $q(x) = \frac{1}{|\mathcal{X}|}$ be the uniform probability mass function over $\mathcal{X}$, and let $p(x)$ be the probability mass function for $X$. Then

$$\begin{aligned} D(p \parallel q) &= \sum_x p(x) \log \frac{p(x)}{q(x)} = \sum_x p(x) \log p(x) - \sum_x p(x) \log q(x) \\ &= -\sum_x p(x) \log \frac{1}{p(x)} - \sum_x p(x) \log \frac{1}{|\mathcal{X}|} \\ &= -\sum_x p(x) \log \frac{1}{p(x)} + \sum_x p(x) \log|\mathcal{X}| \\ &= -H(X) + \log|\mathcal{X}|. \end{aligned}$$

Hence by the nonnegativity of relative entropy,

$$\begin{aligned} 0 \le D(p \parallel q) &= -H(X) + \log|\mathcal{X}| \\ H(X) &\le \log|\mathcal{X}|. \end{aligned}$$

**Theorem 2.6.5** (*Conditioning reduces entropy*)(*Information can't hurt*)

$$H(X|Y) \le H(X)$$

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

Intuitively, the theorem says that knowing another random variable $Y$ can only reduce the uncertainty in $X$. Note that this is true only on the average. Specifically, $H(X|Y = y)$ may be greater than or less than or equal to $H(X)$, but on the average $H(X|Y) = \sum_y p(y)H(X|Y = y) \le H(X)$.

**Theorem 2.6.6** (*Independence bound on entropy*) Let $X_1, X_2, \cdots, X_n$ be drawn according to $p(x_1, x_2, \cdots, x_n)$. Then

$$H(X_1, X_2, \cdots, X_n) \le \sum_{i=1}^{n} H(X_i)$$

with equality if and only if the $X_i$ are independent.

2.7 LOG SUM INEQUALITY AND ITS APPLICATIONS

We now prove a simple consequence of the concavity of the logarithm, which will be used to prove some concavity results for the entropy.

**Theorem 2.7.1** (*Log sum inequality*) For nonnegative numbers, $a_1, a_2, \cdots, a_n$ and $b_1, b_2, \cdots, b_n$,

$$\sum_{i=1}^{n} a_i \log \frac{a_i}{b_i} \ge \left(\sum_{i=1}^{n} a_i\right) \log \frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} b_i}$$

with equality if and only if $\frac{a_i}{b_i} = const$.

**Theorem 2.7.2** (*Convexity of relative entropy*) $D(p \parallel q)$ is convex in the pair $(p, q)$; that is, if $(p_1, q_1)$ and $(p_2, q_2)$ are two pairs of probability mass functions, then

$$D(\lambda p_1 + (1-\lambda)p_2 \parallel \lambda q_1 + (1-\lambda)q_2) \le \lambda D(p_1 \parallel q_1) + (1-\lambda) D(p_2 \parallel q_2)$$

For all $0 \le \lambda \le 1$.

**Theorem 2.7.3** (*Concavity of entropy*) $H(p)$ is a concave function of $p$.

**Theorem 2.7.4** Let $(X, Y) \sim p(x, y) = p(x)p(y|x)$. The mutual information $I(X; Y)$ is a concave function of $p(x)$ for fixed $p(y|x)$ and a convex function of $p(y|x)$ for fixed $p(x)$.

2.8 DATA-PROCESSING INEQUALITY

The data-processing inequality can be used to show that no clever manipulation of the data can improve the inferences that can be made from the data.

**Definition** Random variables $X, Y, Z$ are said to form a ***Markov chain*** in that order (denoted by $X \to Y \to Z$)

if the conditional distribution of $Z$ depends only on $Y$ and is conditionally independent of $X$. Specifically, $X$, $Y$, and $Z$ form a Markov chain $X \to Y \to Z$ if the joint probability mass function can be written as

$$p(x, y, z) = p(x)p(y|x)p(z|y).$$

Some simple consequences are as follows:

• $X \to Y \to Z$ if and only if $X$ and $Z$ are conditionally independent given $Y$. Markovity implies conditional independence because

$$p(x, z|y) = \frac{p(x, y, z)}{p(y)} = \frac{p(x, y)p(z|y)}{p(y)} = p(x|y)p(z|y).$$

This is the characterization of Markov chains that can be extended to define Markov fields, which are $n$-dimensional random processes in which the interior and exterior are independent given the values on the boundary.

• $X \to Y \to Z$ implies that $Z \to Y \to X$. Thus, the condition is sometimes written $X \leftrightarrow Y \leftrightarrow Z$.

• If $Z = f(Y)$, then $X \to Y \to Z$.

We can now prove an important and useful theorem demonstrating that no processing of $Y$, deterministic or random, can increase the information that $Y$ contains about $X$.

**Theorem 2.8.1**: (*Data-processing inequality*) If $X \to Y \to Z$, then $I(X; Y) \geq I(X; Z)$.

**Corollary** In particular, if $Z = g(Y)$, we have $I(X; Y) \geq I\big(X; g(Y)\big)$.

**Corollary** If $X \to Y \to Z$, then $I(X; Y|Z) \leq I(X; Y)$.

2.9 SUFFICIENT STATISTICS

2.10 FANO'S INEQUALITY

**PROBLEMS**:

2.1 *Coin flips*. A fair coin is flipped until the first head occurs. Let $X$ denote the number of flips required.

(a) Find the entropy $H(X)$ in bits. The following expressions may be useful:

$$\sum_{n=0}^{\infty} r^n = \frac{1}{1-r}, (-1 < r < 1) \quad \sum_{n=0}^{\infty} nr^n = \frac{r}{(1-r)^2}.$$

**Solution**:

The number $X$ of tosses till the first head appears has the geometric distribution with parameter $p = 1/2$, where $P(X = n) = pq^{n-1}$, $n \in \{1,2, \cdots\}$. Hence the entropy of $X$ is

$$\begin{aligned} H(X) &= -\sum_{n=1}^{\infty} pq^{n-1} \log(pq^{n-1}) \\ &= -\left[\sum_{n=0}^{\infty} pq^n \log p + \sum_{n=0}^{\infty} npq^n \log q\right] \\ &= \frac{-p \log p}{1-q} - \frac{pq \log q}{p^2} \\ &= \frac{-p \log p - q \log q}{p} \\ &= H(p)/p \text{ bits}. \end{aligned}$$

If $p = 1/2$, then $H(X) = 2$ bits.

(b) A random variable $X$ is drawn according to this distribution. Find an "efficient" sequence of yes–no questions of the form, "Is $X$ contained in the set $S$?" Compare $H(X)$ to the expected number of questions required to determine $X$.

**Solution**:

Intuitively, it seems clear that the best questions are those that have equally likely chances of receiving a yes or a no answer. Consequently, one possible guess is that the most "efficient" series of questions is: Is $X = 1$ ? If not, is $X = 2$ ? If not, is $X = 3$ ? ⋯ with a resulting expected number of questions equal to $\sum_{n=1}^{\infty} n(1/2^n) = 2$. This should reinforce the intuition that $H(X)$ is a measure of the uncertainty of $X$. Indeed in this case, the entropy is exactly the same as the average number of questions needed to define $X$, and in general $E(\#\text{ of questions}) \geq H(X)$. This problem has an interpretation as a source coding problem. Let $0 =$ no, $1 =$ yes, $X =$ source, and $Y =$ Encoded Source. Then the set of questions in the above procedure can be written as a collection of $(X, Y)$ pairs: $(1,1), (2,01), (3,001)$, etc. In fact, this intuitively derived code is the optimal (Huffman) code minimizing the expected number of questions.

**Addition**: (I think it's important)

Since

$$\sum_{n=0}^{\infty} r^n = \frac{1}{1-r}, (-1 < r < 1)$$

If $f(r) = \sum_{n=0}^{\infty} r^n = 1/(1-r)$, $(-1 < r < 1)$, then

$$f'(r) = \sum_{n=1}^{\infty} nr^{n-1} = \frac{1}{(1-r)^2}$$

Multiply both sides by $r$ we can get

$$rf'(r) = \sum_{n=1}^{\infty} nr^n = \frac{r}{(1-r)^2}$$

$$= \sum_{n=0}^{\infty} nr^n$$

So,

$$\sum_{n=0}^{\infty} nr^n = \frac{r}{(1-r)^2}, (-1 < r < 1)$$

2.2 *Entropy of functions*. Let $X$ be a random variable taking on a finite number of values. What is the (general) inequality relationship of $H(X)$ and $H(Y)$ if

(a) $Y = 2^X$?

(b) $Y = \cos X$?

**Solution**: Let $y = g(x)$. Then

$$p(y) = \sum_{x:y=g(x)} p(x).$$

Consider any set of $x$'s that map onto a single $y$. For this set

$$\sum_{x:y=g(x)} p(x) \log p(x) \leq \sum_{x:y=g(x)} p(x) \log p(y) = p(y) \log p(y),$$

since log is a monotone increasing function and $p(x) \leq \sum_{x:y=g(x)} p(x) = p(y)$. Extending this argument to the entire range of $X$ (and $Y$), we obtain

$$H(X) = -\sum_{x} p(x) \log p(x)$$

$$= -\sum_{y} \sum_{x:y=g(x)} p(x) \log p(x)$$

$$\geq -\sum_{y} p(y) \log p(y)$$

$$= H(Y),$$

with equality if $g$ is one-to-one with probability one.

So, since $Y = 2^X$ is one-to-one and hence the entropy, which is just a function of the probabilities (and not the values of a random variable) does not change, i.e., $H(X) = H(Y)$.

$Y = \cos X$ is not necessarily one-to-one. Hence all that we can say is that $H(X) \geq H(Y)$, with equality if cosine is one-to-one on the range of $X$.

2.3 *Minimum entropy*. What is the minimum value of $H(p_1, \cdots, p_n) = H(\mathbf{p})$ as $\mathbf{p}$ ranges over the set of $n$-dimensional probability vectors? Find all $\mathbf{p}$'s that achieve this minimum.

**Solution**:

We wish to find all probability vectors $\mathbf{p} = (p_1, \cdots, p_n)$ which minimize

$$H(\mathbf{p}) = -\sum_i p_i \log p_i.$$

Now $-p_i \log p_i \geq 0$, with equality if $p_i = 0$ or 1. Hence the only possible probability vectors which minimize $H(\mathbf{p})$ are those with $p_i = 1$ for some $i$ and $p_j = 0$, $j \neq i$. There are $n$ such vectors, i.e., $(1,0,\cdots,0)$, $(0,1,0,\cdots,0)$, $\cdots$, $(0,\cdots,0,1)$, and the minimum value of $H(\mathbf{p})$ is 0.

2.4 *Entropy of functions of a random variable*. Let $X$ be a discrete random variable. Show that the entropy of a function of $X$ is less than or equal to the entropy of $X$ by justifying the following steps:

$$\begin{aligned} H(X, g(X)) &\overset{(a)}{=} H(X) + H(g(X) \mid X) \\ &\overset{(b)}{=} H(X), \\ H(X, g(X)) &\overset{(c)}{=} H(g(X)) + H(X \mid g(X)) \\ &\overset{(d)}{\geq} H(g(X)). \end{aligned}$$

Thus, $H\big(g(X)\big) \leq H(X)$.

**Solution**:

Entropy of functions of a random variable,

(a) $H\big(X, g(X)\big) = H(X) + H(g(X)|X)$ by the chain rule for entropies.

(b) $H(g(X)|X) = 0$ since for any particular value of $X$, $g(X)$ is fixed, and hence $H(g(X)|X) = \sum_x p(x) H(g(X)|X = x) = \sum_x 0 = 0$.

(c) $H\big(X, g(X)\big) = H\big(g(X)\big) + H\big(X|g(X)\big)$ again by the chain rule.

(d) $H\big(X|g(X)\big) \geq 0$, with equality if $X$ is a function of $g(X)$, i.e., $g(\cdot)$ is one-to-one. Hence $H\big(X, g(X)\big) \geq H\big(g(X)\big)$.

Combining parts (b) and (d), we obtain $H(X) \geq H\big(g(X)\big)$.

2.13 *Inequality*. Show that $\ln x \geq 1 - \frac{1}{x}$ for $x > 0$.

**Solution**: *Inequality*.

Using the Remainder form of the Taylor expansion of $\ln x$ about $x = 1$, we have for some $c$ between 1 and $x$

$$\ln x = \ln(1) + \left(\frac{1}{t}\right)_{t=1}(x-1) + \left(\frac{-1}{t^2}\right)_{t=c}\frac{(x-1)^2}{2} \leq x - 1$$

since the second term is always negative. Hence letting $y = 1/x$, we obtain

$$-\ln y \leq \frac{1}{y} - 1$$

or

$$\ln y \geq 1 - \frac{1}{y}$$

with equality if $y = 1$.

2.21 *Markov's inequality for probabilities*. Let $p(x)$ be a probability mass function. Prove, for all $d \geq 0$, that

$$\Pr\{p(X) \leq d\} \log \frac{1}{d} \leq H(X).$$

**Solution**:

Markov inequality applied to entropy.

$$\begin{aligned}\Pr\{p(X) \le d\} \log \frac{1}{d} &= \sum_{x:p(x)\le d} p(x) \log \frac{1}{d} \\ &\le \sum_{x:p(x)\le d} p(x) \log \frac{1}{p(x)} \\ &\le \sum_{x} p(x) \log \frac{1}{p(x)} \\ &= H(X).\end{aligned}$$

2.26 *Another proof of nonnegativity of relative entropy*. In view of the fundamental nature of the result $D(p \parallel q) \ge 0$, we will give another proof.

(a) Show that $\ln x \le x - 1$ for $0 < x < \infty$.

See exercise 2.13.

(b) Justify the following steps:

$$\begin{aligned}-D(p \parallel q) &= \sum_{x} p(x) \ln \frac{q(x)}{p(x)} \\ &\le \sum_{x} p(x) \left( \frac{q(x)}{p(x)} - 1 \right) \\ &\le 0.\end{aligned}$$

We let $A$ be the **set** of $x$ such that $p(x) > 0$.

$$\begin{aligned}-D(p \parallel q) &= \sum_{x} p(x) \ln \frac{q(x)}{p(x)} \\ &\le \sum_{x} p(x) \left( \frac{q(x)}{p(x)} - 1 \right) \\ &= \sum_{x \in A} q(x) - \sum_{x \in A} p(x) \\ &\le 0\end{aligned}$$

The first step follows from the definition of $D$, the second step follows from the inequality $\ln t \le t - 1$, the third step from expanding the sum, and the last step from **the fact** that the $q(A) \le 1$ and $p(A) = 1$. (Should we really do it here?)

(c) What are the conditions for equality?

$$p(x) = q(x) \text{ for all } x.$$

## CHAPTER 3 ASYMPTOTIC EQUIPARTITION PROPERTY

In information theory, the analog of the **law of large numbers** is the **asymptotic equipartition property (AEP)**. It is a direct consequence of the weak law of large numbers. The law of large numbers states that for **independent, identically distributed (i.i.d.)** random variables, $\frac{1}{n}\sum_{i=1}^{n} X_i$ is close to its expected value $E[X]$ for large values of $n$. The **AEP** states that $\frac{1}{n}\log\frac{1}{p(X_1,X_2,\cdots,X_n)}$ is close to the entropy $H(X)$, where $X_1, X_2, \cdots, X_n$ are i.i.d. random variables and $p(X_1, X_2, \cdots, X_n)$ is the probability of observing the sequence $X_1, X_2, \cdots, X_n$. Thus, the probability $p(X_1, X_2, \cdots, X_n)$ assigned to an observed sequence will be close to $2^{-nH(X)}$.

This enables us to divide the set of all sequences into two sets, the *typical* set, where the sample entropy is close to the true entropy, and the nontypical set, which contains the other sequences. Most of our attention will be on the typical sequences. Any property that is proved for the typical sequences will then be true with high probability and will determine the average behavior of a large sample.

First, an example. Let the random variable $X \in \{0,1\}$ have a probability mass function defined by $p(1) = p$ and $p(0) = q$. If $X_1, X_2, \cdots, X_n$ are i.i.d. according to $p(x)$, the probability of a sequence $x_1, x_2, \cdots, x_n$ is $\prod_{i=1}^{n} p(x_i)$. For example, the probability of the sequence (1,0,1,1,0,1) is $p^{\sum X_i} q^{n-\sum X_i} = p^4 q^2$. Clearly, it is not true that all $2^n$ sequences of length $n$ have the same probability.

However, we might be able to predict the probability of the sequence that we actually observe. We ask for the probability $p(X_1, X_2, \cdots, X_n)$ of the outcomes $X_1, X_2, \cdots, X_n$, where $X_1, X_2, \cdots$ are i.i.d. $\sim p(x)$. This is insidiously self-referential, but well defined nonetheless. Apparently, we are asking for the probability of an event drawn according to the same probability distribution. Here it turns out that $p(X_1, X_2, \cdots, X_n)$ is close to $2^{-nH(X)}$ with high probability.

We summarize this by saying, "Almost all events are almost equally surprising." This is a way of saying that

$$\Pr\{(X_1, X_2, \cdots, X_n): p(X_1, X_2, \cdots, X_n) = 2^{-n(H(X)\pm\epsilon)}\} \approx 1$$

if $X_1, X_2, \cdots, X_n$ are i.i.d. $\sim p(x)$.

In the example just given, where $p(X_1, X_2, \cdots, X_n) = p^{\sum X_i} q^{n-\sum X_i}$, we are simply saying that the number of 1's in the sequence is close to $np$ (with high probability), and all such sequences have (roughly) the same probability $2^{-nH(p)}$. We use the idea of convergence in probability, defined as follows:

**Definition** (*Convergence of random variables*). Given a **sequence** of random variables, $X_1, X_2, \cdots$, we say that the sequence $X_1, X_2, \cdots$ **converges** to a random variable $X$:

1. In probability if for every $\epsilon > 0$, $\Pr\{|X_n - X| > \epsilon\} \to 0$
2. In mean square if $E[(X_n - X)^2] \to 0$
3. With probability 1 (also called almost surely) if $\Pr\left\{\lim_{n\to\infty} X_n = X\right\} = 1$

### 3.1 ASYMPTOTIC EQUIPARTITION PROPERTY THEOREM

The asymptotic equipartition property is formalized in the following theorem.

**Theorem 3.1.1** (*AEP*) If $X_1, X_2, \cdots$ are i.i.d. $\sim p(x)$, then

$$-\frac{1}{n}\log p(X_1, X_2, \cdots, X_n) \to H(X) \quad \text{in probability.}$$

**Proof**: Functions of independent random variables are also independent random variables. Thus, since the $X_i$ are i.i.d., so are $\log p(X_i)$. Hence, by the weak law of large numbers,

$$\begin{aligned} -\frac{1}{n}\log p(X_1, X_2, \cdots, X_n) &= -\frac{1}{n}\sum_i \log p(X_i) \\ &\to -E[\log p(X)] \quad \text{in probability} \\ &= H(X), \end{aligned}$$

**Definition** The *typical set* $A_\epsilon^{(n)}$ with respect to $p(x)$ is the set of sequences $(x_1, x_2, \cdots, x_n) \in \mathcal{X}^n$ with the property

$$2^{-n(H(X)+\epsilon)} \le p(x_1, x_2, \cdots, x_n) \le 2^{-n(H(X)-\epsilon)}.$$

As a consequence of the AEP, we can show that the set $A_\epsilon^{(n)}$ has the following properties:

**Theorem 3.1.2**

1. If $(x_1, x_2, \cdots, x_n) \in A_\epsilon^{(n)}$, then $H(X) - \epsilon \leq -\frac{1}{n}\log p(x_1, x_2, \cdots, x_n) \leq H(X) + \epsilon$.
2. $\Pr\left\{A_\epsilon^{(n)}\right\} > 1 - \epsilon$ for $n$ sufficiently large.
3. $\left|A_\epsilon^{(n)}\right| \leq 2^{n(H(X)+\epsilon)}$, where $|A|$ denotes the **number** of elements in the set $A$.
4. $\left|A_\epsilon^{(n)}\right| \geq (1-\epsilon)2^{n(H(X)-\epsilon)}$ for $n$ sufficiently large.

3.2 CONSEQUENCES OF THE AEP: DATA COMPRESSION

Let $X_1, X_2, \cdots, X_n$ be independent, identically distributed random variables drawn from the probability mass function $p(x)$. We wish to find short descriptions for such sequences of random variables. We divide all sequences in $\mathcal{X}^n$ into two sets: the typical set $A_\epsilon^{(n)}$ and its complement, as shown in Figure 3.1.

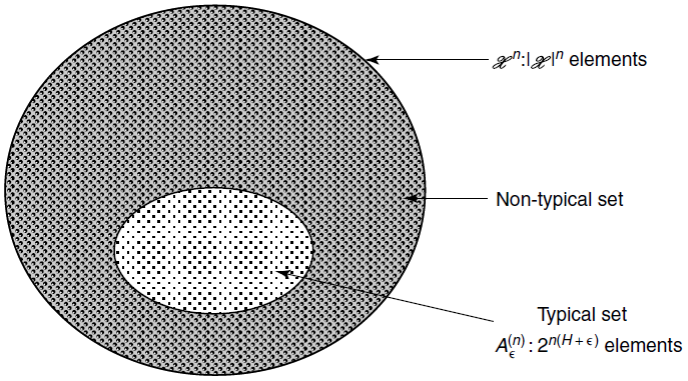



Figure 3. 1 Typical sets and source coding.

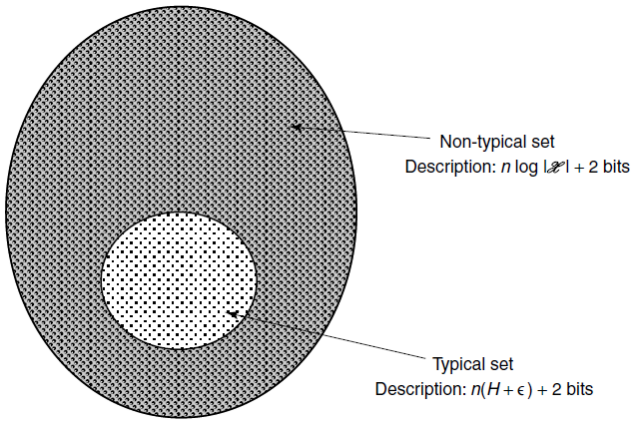



Figure 3. 2 Source code using the typical set.

We order all elements in each set according to some order (e.g., lexicographic order). Then we can represent each sequence of $A_\epsilon^{(n)}$ by giving the index of the sequence in the set. **Since** there are $\leq 2^{n(H+\epsilon)}$ **sequences** in $A_\epsilon^{(n)}$, the indexing requires no more than $n(H+\epsilon)+1$ bits. [The extra bit may be necessary because $n(H+\epsilon)$ may not be an integer.] We prefix all these sequences by a 0, giving a total length of $\leq n(H+\epsilon)+2$ bits to represent each sequence in $A_\epsilon^{(n)}$ (see Figure 3.2). Similarly, we can index each sequence not in $A_\epsilon^{(n)}$ by using **not more than** $n\log|\mathcal{X}|+1$ bits. Prefixing these indices by 1, we have a code for all the sequences in $\mathcal{X}^n$.

Note the following features of the above coding scheme:

- The code is one-to-one and easily decodable. The initial bit acts as a flag bit to indicate the length of the **codeword** that follows.
- We have used a brute-force enumeration of the atypical set $A_\epsilon^{(n)^c}$ without taking into account the fact that the number of elements in $A_\epsilon^{(n)^c}$ is less than the number of elements in $\mathcal{X}^n$. Surprisingly, this is good enough to yield an efficient description.

• The typical sequences have short descriptions of length $\approx nH$.

We use the notation $x^n$ to denote a sequence $x_1, x_2, \cdots, x_n$. Let $l(x^n)$ be the length of the codeword corresponding to $x^n$. If $n$ is sufficiently large so that $\Pr\left\{A_\epsilon^{(n)}\right\} > 1-\epsilon$, the expected **length** of the **codeword** is

$$\begin{aligned} E\big(l(X^n)\big) &= \sum_{x^n} p(x^n) l(x^n) \\ &= \sum_{x^n \in A_\epsilon^{(n)}} p(x^n) l(x^n) + \sum_{x^n \in A_\epsilon^{(n)^c}} p(x^n) l(x^n) \\ &\le \sum_{x^n \in A_\epsilon^{(n)}} p(x^n)(n(H+\epsilon)+2) + \sum_{x^n \in A_\epsilon^{(n)^c}} p(x^n)(n\log|\mathcal{X}|+2) \\ &= \Pr\left\{A_\epsilon^{(n)}\right\}(n(H+\epsilon)+2) + \Pr\left\{A_\epsilon^{(n)^c}\right\}(n\log|\mathcal{X}|+2) \\ &\le n(H+\epsilon) + \epsilon n(\log|\mathcal{X}|) + 2 \\ &= n(H+\epsilon'), \end{aligned}$$

where $\epsilon' = \epsilon + \epsilon\log|\mathcal{X}| + \frac{2}{n}$ can be made arbitrarily small by an appropriate choice of $\epsilon$ followed by an appropriate choice of $n$. Hence we have proved the following theorem.

**Theorem 3.2.1** Let $X^n$ be i.i.d. $\sim p(x)$. Let $\epsilon > 0$. Then there exists a code that maps sequences $x^n$ of length $n$ into binary strings such that the mapping is one-to-one (and therefore invertible) and

$$E\left(\frac{1}{n} l(X^n)\right) \le H(X) + \epsilon$$

for $n$ sufficiently large.

Thus, we can represent sequences $X^n$ using $nH(X)$ bits on the average.

3.3 HIGH-PROBABILITY SETS AND THE TYPICAL SET

From the definition of $A_\epsilon^{(n)}$, it is clear that $A_\epsilon^{(n)}$ is a fairly small set that contains most of the probability. But from the definition, it is not clear whether it is the smallest such set. We will prove that the typical set has essentially the same number of elements as the smallest set, to first order in the exponent.

**Definition** For each $n = 1,2,\cdots$, let $B_\delta^{(n)} \subset \mathcal{X}^n$ be the smallest set with

$$\Pr\left\{B_\delta^{(n)}\right\} \ge 1-\delta.$$

We argue that $B_\delta^{(n)}$ must have significant intersection with $A_\epsilon^{(n)}$ and therefore must have about as many elements.

**Theorem 3.3.1** Let $X_1, X_2, \cdots, X_n$ be i.i.d. $\sim p(x)$. For $\delta < \frac{1}{2}$ and any $\delta' > 0$, if $\Pr\left\{B_\delta^{(n)}\right\} > 1-\delta$, then

$$\frac{1}{n}\log\left|B_\delta^{(n)}\right| > H - \delta' \quad \text{for } n \text{ sufficiently large.}$$

Thus, $B_\delta^{(n)}$ must have at least $2^{nH}$ elements, to first order in the exponent. But $A_\epsilon^{(n)}$ has $2^{n(H\pm\epsilon)}$ elements. Therefore, $A_\epsilon^{(n)}$ s about the same size as the smallest high-probability set.

Addition:

**Ergodic process**: In econometrics and signal processing, a stochastic process is said to be ergodic if its statistical properties can be deduced from a single, sufficiently long, random sample of the process.

## CHAPTER 4 ENTROPY RATES OF A STOCHASTIC PROCESS

The asymptotic equipartition property in Chapter 3 establishes that $nH(X)$ bits suffice on the average to describe $n$ independent and identically distributed random variables. But what if the random variables are **dependent**? In particular, what if the random variables form a stationary process? We will show, just as in the i.i.d. case, that the entropy $H(X_1, X_2, \cdots, X_n)$ grows (asymptotically) linearly with $n$ at a rate $H(\mathcal{X})$, which we will call the ***entropy rate*** of the process. The interpretation of $H(\mathcal{X})$ as the best achievable data compression will await the analysis in Chapter 5.

4.1 MARKOV CHAINS

A **stochastic process** $\{X_i\}$ is an indexed sequence of random variables. In general, there can be an arbitrary **dependence** among the random variables. The process is characterized by the joint probability mass functions $\Pr\{(X_1, X_2, \cdots, X_n) = (x_1, x_2, \cdots, x_n)\} = p(x_1, x_2, \cdots, x_n)$, $(x_1, x_2, \cdots, x_n) \in \mathcal{X}^n$ for $n = 1,2, \cdots$.

**Definition** A **stochastic process** is said to be ***stationary*** if the joint distribution of any subset of the sequence of random variables is invariant with respect to shifts in the time index; that is,

$$\Pr\{X_1 = x_1, X_2 = x_2, \cdots, X_n = x_n\}$$
$$= \Pr\{X_{1+l} = x_1, X_{2+l} = x_2, \cdots, X_{n+l} = x_n\}$$

for every $n$ and every shift $l$ and for all $x_1, x_2, \cdots, x_n \in \mathcal{X}$.

A simple example of a stochastic process with **dependence** is one in which **each** random variable depends **only** on the **one** preceding it and is *conditionally* independent of all the other preceding random variables. Such a process is said to be **Markov**.

**Definition** A discrete stochastic process $X_1, X_2, \cdots$ is said to be a ***Markov chain*** or a ***Markov process*** if for $n = 1,2, \cdots$,

$$\Pr(X_{n+1} = x_{n+1}, |X_n = x_n, X_{n-1} = x_{n-1}, \cdots, X_1 = x_1)$$
$$= \Pr(X_{n+1} = x_{n+1}, |X_n = x_n)$$

for all $x_1, x_2, \cdots, x_n, x_{n+1} \in \mathcal{X}$.

In this case, the joint probability mass function of the random variables **can be** written as

$$p(x_1, x_2, \cdots, x_n) = p(x_1)p(x_2|x_1)p(x_3|x_2) \cdots p(x_n|x_{n-1}).$$

**Definition** The Markov chain is said to be ***time invariant*** if the conditional probability $p(x_{n+1}|x_n)$ does not depend on $n$; that is, for $n = 1,2, \cdots$,

$$\Pr\{X_{n+1} = b|X_n = a\} = \Pr\{X_2 = b|X_1 = a\} \quad \text{for all } a, b \in \mathcal{X}.$$

We will assume that the Markov chain is time invariant unless otherwise stated.

If $\{X_i\}$ is a Markov chain, $X_n$ is called the ***state*** at time $n$. A time invariant Markov chain is characterized by its initial state and a ***probability transition matrix*** $P = [P_{ij}]$, $i, j \in \{1,2, \cdots, m\}$, where $P_{ij} = \Pr\{X_{n+1} = j|X_n = i\}$.

If it is possible to go with positive probability from any state of the Markov chain to any other state in a finite number of steps, the Markov chain is said to be *irreducible*. If the largest common factor of the lengths of different paths from a state to itself is 1, the Markov chain is said to *aperiodic*.

**Addition**:

A Markov chain is said to be irreducible if its state space is a single communicating class; in other words, if it is possible to get to any state from any state.

If the probability mass function of the random variable at time $n$ is $p(x_n)$, the probability mass function at time $n + 1$ is

$$p(x_{n+1}) = \sum_{x_n} p(x_n) P_{x_n x_{n+1}}.$$

A distribution on the states such that the distribution at time $n + 1$ is the same as the distribution at time $n$ is called a **stationary distribution**. The stationary distribution is so called because if the initial state of a Markov chain is drawn according to a stationary distribution, the Markov chain forms a stationary process.

If the finite-state Markov chain is irreducible and aperiodic, the stationary distribution is unique, and from any starting distribution, the distribution of $X_n$ tends to the stationary distribution as $n \to \infty$.

**Example 4.1.1**: Consider a two-state Markov chain with a probability transition matrix

$$P = \begin{bmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{bmatrix}$$

as shown in Figure 4.1.

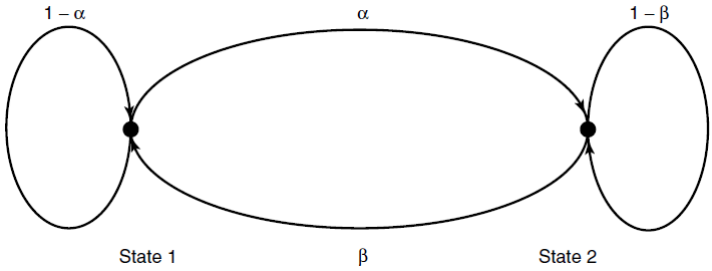



Figure 4. 1 Two-state Markov chain.

Let the stationary distribution be represented by a vector $\mu$ whose components are the stationary probabilities of states 1 and 2, respectively. Then the stationary probability can be found by solving the equation $\mu P = \mu$ or, more simply, by balancing probabilities. For the stationary distribution, the net probability flow across any cut set in the state transition graph is zero. Applying this to Figure 4.1, we obtain

$$\mu P = [\mu_1 \quad \mu_2] \begin{bmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{bmatrix} = [\mu_1 \quad \mu_2]$$

$$\mu_1 \alpha = \mu_2 \beta.$$

Since $\mu_1 + \mu_2 = 1$, the stationary distribution is

$$\mu_1 = \frac{\beta}{\alpha+\beta}, \quad \mu_2 = \frac{\alpha}{\alpha+\beta}.$$

If the Markov chain has an initial state drawn according to the stationary distribution, the resulting process will be stationary. The entropy of the state $X_n$ at time $n$ is

$$H(X_n) = H\left(\frac{\beta}{\alpha+\beta}, \frac{\alpha}{\alpha+\beta}\right).$$

However, this is not the rate at which entropy grows for $H(X_1, X_2, \cdots, X_n)$. The dependence among the $X_i$'s will take a steady toll.

4.2 ENTROPY RATE

If we have a sequence of $n$ random variables, a natural question to ask is: How does the entropy of the sequence grow with $n$? We define the ***entropy rate*** as this rate of growth as follows.

**Definition** The *entropy rate* of a stochastic process $\{X_i\}$ is defined by

$$H(\mathcal{X}) = \lim_{n\to\infty} \frac{1}{n} H(X_1, X_2, \cdots, X_n)$$

when the limit exists.

We now consider some simple examples of stochastic processes and their corresponding entropy rates.

1. Typewriter.

Consider the case of a typewriter that has $m$ equally likely output letters. The typewriter can produce $m^n$ sequences of length $n$, all of them equally likely. Hence $H(X_1, X_2, \cdots, X_n) = \log m^n$ and the entropy rate is $H(\mathcal{X}) = \log m$ bits per symbol.

2. $X_1, X_2, \cdots$ are i.i.d. random variables. Then

$$H(\mathcal{X}) = \lim \frac{H(X_1, X_2, \cdots, X_n)}{n} = \lim \frac{nH(X_1)}{n} = H(X_1),$$

which is what one would expect for the entropy rate per symbol.

3. Sequence of independent but not identically distributed random variables. (Page 74)

We can also define a related quantity for entropy rate:

$$H'(\mathcal{X}) = \lim_{n\to\infty} H(X_n | X_{n-1}, X_{n-2}, \cdots, X_1)$$

when the limit exists.

The two quantities $H(\mathcal{X})$ and $H'(\mathcal{X})$ correspond to two different notions of entropy rate. The first is the per symbol entropy of the $n$ random variables, and the second is the conditional entropy of the last random variable given the past.

**Theorem 4.2.1** For a stationary stochastic process, the limits in $H(\mathcal{X})$ and $H'(\mathcal{X})$ exist and are equal:

$$H(\mathcal{X}) = H'(\mathcal{X})$$

**Theorem 4.2.2** For a stationary stochastic process, $H(X_n|X_{n-1}, \cdots, X_1)$ is nonincreasing in $n$ and has a limit $H'(\mathcal{X})$.

**Theorem 4.2.3** (Cesáro mean) If $a_n \to a$ and $b_n = \frac{1}{n}\sum_{i=1}^{n} a_i$, then $b_n \to a$.

The significance of the entropy rate of a stochastic process arises from the AEP for a stationary ergodic process. We prove the general AEP in Section 16.8, where we show that for any stationary ergodic process,

$$-\frac{1}{n}\log p(X_1, X_2, \cdots, X_n) \to H(\mathcal{X})$$

with probability 1. Using this, the theorems of Chapter 3 can easily be extended to a general stationary ergodic process. We can define a typical set in the same way as we did for the i.i.d. case in Chapter 3. By the same arguments, we can show that the typical set has a probability close to 1 and that there are about $2^{nH(\mathcal{X})}$ typical sequences of length $n$, each with probability about $2^{-nH(\mathcal{X})}$. We can therefore represent the typical sequences of length $n$ using approximately $nH(\mathcal{X})$ bits. This shows the significance of the entropy rate as the average description length for a stationary ergodic process.

**Markov Chains**. For a stationary Markov chain, the entropy rate is given by

$$H(\mathcal{X}) = H'(\mathcal{X}) = \lim H(X_n|X_{n-1}, \cdots, X_1) = \lim H(X_n|X_{n-1})$$
$$= H(X_2|X_1),$$

where the conditional entropy is calculated using the given stationary distribution. Recall that the stationary distribution $\mu$ is the solution of the equations

$$\mu_j = \sum_{i=1}^{M} \mu_i P_{ij} \quad \text{for all } j = 1,2, \cdots M.$$

**Theorem 4.2.4** Let $\{X_i\}$ be a stationary Markov chain with stationary distribution $\mu$ and transition matrix $P$. Let $X_1 \sim \mu$. Then the entropy rate is

$$H(\mathcal{X}) = -\sum_{ij} \mu_i P_{ij} \log P_{ij} = \sum_i \mu_i \left( \sum_j -P_{ij} \log P_{ij} \right).$$

**Example 4.2.1**: (Two-state Markov chain) The entropy rate of the two-state Markov chain in Figure 4.1 is

$$H(\mathcal{X}) = H(X_2|X_1) = \frac{\beta}{\alpha+\beta} H(\alpha) + \frac{\alpha}{\alpha+\beta} H(\beta).$$

$$H(\mathcal{X}) = H(X_2|X_1) = \frac{\beta}{\alpha+\beta}\left(\alpha \log\frac{1}{\alpha} + (1-\alpha)\log\frac{1}{1-\alpha}\right) + \frac{\alpha}{\alpha+\beta}\left(\beta \log\frac{1}{\beta} + (1-\beta)\log\frac{1}{1-\beta}\right).$$

4.3 EXAMPLE: ENTROPY RATE OF A RANDOM WALK ON AWEIGHTED GRAPH

As an example of a stochastic process, let us consider a random walk on a connected graph (Figure 4.2). (Page 78)

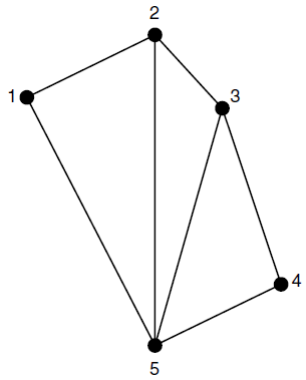


Figure 4. 2 Random walk on a graph.

**Remark** It is easy to see that a stationary random walk on a graph is ***time-reversible***; that is, the probability of any sequence of states is the same forward or backward:

$$\Pr(X_1 = x_1, X_2 = x_2, \cdots, X_n = x_n) = \Pr(X_n = x_1, X_{n-1} = x_2, \cdots, X_1 = x_n).$$

Rather surprisingly, the converse is also true; that is, any time-reversible Markov chain can be represented as a random walk on an undirected weighted graph.

4.4 SECOND LAW OF THERMODYNAMICS

**Definition** A probability transition matrix $[P_{ij}]$, $P_{ij} = \Pr\{X_{n+1} = j | X_n = i\}$, is called doubly stochastic if

$$\sum_i P_{ij} = 1, \quad j = 1,2,\cdots$$

and

$$\sum_j P_{ij} = 1, \quad i = 1,2,\cdots$$

**Remark** The uniform distribution is a stationary distribution of $P$ if and only if the probability transition matrix is doubly stochastic

4.5 FUNCTIONS OF MARKOV CHAINS

**PROBLEMS**

4.30 *Markov chain transitions*

$$P = [P_{ij}] = \begin{bmatrix} \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \end{bmatrix}.$$

Let $X_1$ be distributed uniformly over the states $\{0,1,2\}$. Let $\{X_i\}_1^\infty$ be a Markov chain with transition matrix $P$; thus, $P(X_{n+1} = j | X_n = i) = P_{ij}$, $i, j \in \{0,1,2\}$.

(a) Is $\{X_n\}$ stationary?

Let $\mu_n$ denote the probability mass function at time $n$. Since $\mu_1 = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$ and $\mu_2 = \mu_1 P = \mu_1$, $\mu_n = \mu_1 = \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$ for all $n$ and $\{X_n\}$ is stationary.

Alternatively, the observation $P$ is doubly stochastic will lead the same conclusion.

(b) Find $\lim_{n\to\infty} \frac{1}{n} H(X_1, X_2, \cdots, X_n)$.

Since $\{X_n\}$ is stationary Markov,

$$\lim_{n\to\infty}\frac{1}{n}H(X_1, X_2, \cdots, X_n) = H(X_2|X_1)$$
$$= \sum_{k=0}^{2} P(X_1 = k)H(X_2|X_1 = k)$$
$$= \frac{3}{2}.$$

## CHAPTER 5 DATA COMPRESSION

We now put content in the definition of entropy by establishing the fundamental limit for the compression of information. Data compression can be achieved by assigning short descriptions to the most frequent outcomes of the data source, and necessarily longer descriptions to the less frequent outcomes. For example, in Morse code, the most frequent symbol is represented by a single dot. In this chapter we find the shortest average description length of a random variable.

We first define the notion of an instantaneous code and then prove the important Kraft inequality, which asserts that the exponentiated codeword length assignments must look like a probability mass function. Elementary calculus then shows that the expected description length must be greater than or equal to the entropy, the first main result. Then Shannon's simple construction shows that the expected description length can achieve this bound asymptotically for repeated descriptions. This establishes the entropy as a natural measure of efficient description length. The famous Huffman coding procedure for finding minimum expected description length assignments is provided. Finally, we show that Huffman codes are competitively optimal and that it requires roughly $H$ fair coin flips to generate a sample of a random variable having entropy $H$. Thus, the entropy is the data compression limit as well as the number of bits needed in random number generation, and codes achieving $H$ turn out to be optimal from many points of view.

### 5.1 EXAMPLES OF CODES

**Definition** A ***source code*** $C$ for a random variable $X$ is a mapping from $\mathcal{X}$, the range of $X$, to $\mathcal{D}^*$, the set of finite-length strings of symbols from a $D$-ary alphabet. Let $C(x)$ denote the codeword (source code) corresponding to $x$ and let $l(x)$ denote the length of $C(x)$.

For example, $C(\text{red}) = 00$, $C(\text{blue}) = 11$ is a source code for $\mathcal{X} = \{\text{red,blue}\}$ with alphabet $\mathcal{D} = \{0,1\}$. ($\mathcal{D}^* = \{00,11\}$)

**Definition** The ***expected length*** $L(C)$ of a source code $C(x)$ for a random variable $X$ with probability mass function $p(x)$ is given by

$$L(C) = \sum_{x \in \mathcal{X}} p(x) l(x),$$

where $l(x)$ is the length of the codeword associated with $x$.

Without loss of generality, we can assume that the $D$-ary alphabet is $\mathcal{D} = \{0,1,\cdots,D-1\}$.

**Example 5.1.1**: Let $X$ be a random variable with the following distribution and codeword assignment:

$$\Pr(X = 1) = \frac{1}{2}, \quad \text{codeword } C(1) = 0$$
$$\Pr(X = 2) = \frac{1}{4}, \quad \text{codeword } C(2) = 10$$
$$\Pr(X = 3) = \frac{1}{8}, \quad \text{codeword } C(3) = 110$$
$$\Pr(X = 4) = \frac{1}{8}, \quad \text{codeword } C(4) = 111.$$

Note: $D = 2$, $\mathcal{D} = \{0,1\}$, $\mathcal{D}^* = \{0,10,110,111\}$.

The entropy $H(X)$ of $X$ is 1.75 bits, and the expected length $L(C) = E[l(X)]$ of this code is also 1.75 bits. Here we have a code that has the same average length as the entropy. We note that any sequence of bits can be uniquely decoded into a sequence of symbols of $X$. For example, the bit string $0110111100110$ is decoded as $134213$.

**Example 5.1.2**: Here $E[l(X)] > H(X)$.

We now define increasingly more stringent conditions on codes. Let $x^n$ denote $(x_1, x_2, \cdots, x_n)$.

**Definition** A code is said to be *nonsingular* if every element of the range of $X$ maps into a different string in $\mathcal{D}^*$; that

$$x \neq x' \Rightarrow (x) \neq C(x').$$

Nonsingularity suffices for an unambiguous description of a single value of $X$. But we usually wish to send a sequence of values of $X$. In such cases we can ensure decodability by adding a special symbol (a "comma")

between any two codewords. But this is an inefficient use of the special symbol; we can do better by developing the idea of **self-punctuating** or instantaneous codes. Motivated by the necessity to send sequences of symbols $X$, we define the extension of a code as follows:

**Definition** The ***extension*** $C^*$ of a code $C$ is the mapping from finite-length strings of $\mathcal{X}$ to finite-length strings of $\mathcal{D}$, defined by

$$C(x_1x_2\cdots x_n) = C(x_1)C(x_2)\cdots C(x_n),$$

where $C(x_1)C(x_2)\cdots C(x_n)$ indicates concatenation of the corresponding codewords.

**Example 5.1.4**: If $C(x_1) = 00$ and $C(x_2) = 11$, then $C(x_1x_2) = 0011$.

**Definition** A code is called *uniquely decodable* if its **extension** is **nonsingular**.

In other words, any encoded string in a uniquely decodable code has only one possible source string producing it. However, one may have to look at the entire string to determine even the first symbol in the corresponding source string.

**Definition** A code is called a ***prefix code*** or an ***instantaneous code*** if no codeword is a prefix of any other codeword.

An instantaneous code can be decoded without reference to future codewords since the end of a codeword is immediately recognizable. Hence, for an instantaneous code, the symbol $x_i$ can be decoded as soon as we come to the end of the codeword corresponding to it. We need not wait to see the codewords that come later. An instantaneous code is a *self-punctuating code*; we can look down the sequence of code symbols and add the commas to separate the codewords without looking at later symbols. For example, the binary string $01011111010$ produced by the code of Example 5.1.1 is parsed as $0,10,111,110,10$.

The nesting of these definitions is shown in Figure 5.1. To illustrate the differences between the various kinds of codes, consider the examples of codeword assignments $C(x)$ to $x \in \mathcal{X}$ in Figure 5.2. For the nonsingular code, the code string $010$ has three possible source sequences: $2$ or $14$ or $31$, and hence the code is not uniquely decodable. The uniquely decodable code is not prefix-free and hence is not instantaneous. To see that it is uniquely decodable, take any code string and start from the beginning. If the first two bits are $00$ or $10$, they can be decoded immediately. If the first two bits are $11$, we must look at the following bits. If the next bit is a $1$, the first source symbol is a $3$. If the length of the string of $0$'s immediately following the $11$ is odd, the first codeword must be $110$ and the first source symbol must be $4$; if the length of the string of $0$'s is even, the first source symbol is a $3$. By repeating this argument, we can see that this code is uniquely decodable. **Sardinas and Patterson** have devised a finite test for unique decodability, which involves forming sets of possible suffixes to the codewords and eliminating them systematically. The fact that the last code in Figure 5.2 is instantaneous is obvious since no codeword is a prefix of any other.

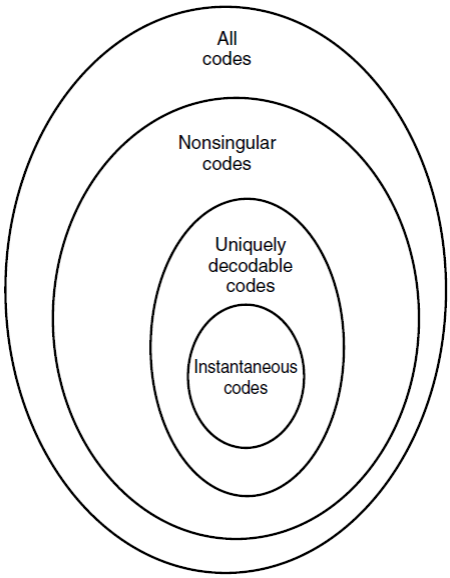


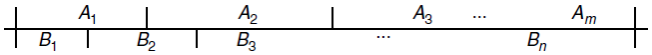

Figure 5. 1 Classes of codes.

| $X$ | Singular | Nonsingular, But Not Uniquely Decodable | Uniquely Decodable, But Not Instantaneous | Instantaneous |
|---|---|---|---|---|
| 1 | 0 | 0 | 10 | 0 |
| 2 | 0 | 010 | 00 | 10 |
| 3 | 0 | 01 | 11 | 110 |
| 4 | 0 | 10 | 110 | 111 |

Figure 5. 2 Classes of Codes

**PROBLEMS** 5.27

*Sardinas–Patterson test for unique decodability*. A code is not uniquely decodable if and only if there exists a finite sequence of code symbols which can be resolved into sequences of codewords in two different ways. That is, a situation such as




must occur where each $A_i$ and each $B_i$ is a codeword. Note that $B_1$ must be a prefix of $A_1$ with some resulting "dangling suffix." Each dangling suffix must in turn be either a prefix of a codeword or have another codeword as its prefix, resulting in another dangling suffix. Finally, the last dangling suffix in the sequence must also be a codeword. Thus, one can set up a test for unique decodability (which is essentially the Sardinas–Patterson test) in the following way: Construct a set $S$ of all possible dangling suffixes. The code is uniquely decodable if and only if $S$ contains no codeword.

**Solution**: *Test for unique decodability*.

The proof of the Sardinas-Patterson test has two parts. In the first part, we will show that if there is a code string that has two different interpretations, then the code will fail the test. The simplest case is when the concatenation of two codewords yields another codeword. In this case, $S_2$ will contain a codeword, and hence the test will fail. In general, the code is not uniquely decodeable, if there exists a string that admits two different parsings into codewords, e.g.

$$x_1x_2x_3x_4x_5x_6x_7x_8 = x_1x_2, x_3x_4x_5, x_6x_7x_8 = x_1x_2x_3x_4, x_5x_6x_7x_8.$$

In this case, $S_2$ will contain the string $x_3x_4$ ( $x_1x_2$, $x_1x_2x_3x_4$), $S_3$ will contain $x_5$ ($x_3x_4$, $x_3x_4x_5$), $S_4$ will contain $x_6x_7x_8$ ($x_5$, $x_5x_6x_7x_8$), which is a codeword. It is easy to see that this procedure will work for any string that has two different parsings into codewords; a formal proof is slightly more difficult and using induction.

In the second part, we will show that if there is a codeword in one of the sets $S_i$, $i \geq 2$, then there exists a string with two different possible interpretations, thus showing that the code is not uniquely decodeable.

To do this, we essentially reverse the construction of the sets. We will not go into the details - the reader is referred to the original paper.

**(a)** State the precise rules for building the set $S$.

Let $S_1$ be the original set of codewords. We construct $S_i$ as follows: A string $y$ is in $S_i$ if there is a dangling suffix in $\{S_1 \cup S_{i-1}\}$, such that $xy$ is in $\{S_1 \cup S_{i-1}\}$ ($x$ is in $\{S_1 \cup S_{i-1}\}$). Then the code is uniquely decodable if none of the $S_i$, $i \geq 2$ contains a codeword. Thus the set $S = \cup_{i \geq 2} S_i$.

**(b)** Suppose that the codeword lengths are $l_i$, $i = 1,2,\cdots,m$. Find a good upper bound on the number of elements in the set $S$.

A simple upper bound can be obtained from the fact that all strings in the sets $S_i$ have length less than $l_{max}$, and therefore the maximum number of elements in $S$ is less than $2^{l_{max}}$.

**(c)** Determine which of the following codes is uniquely decodable:

i. $\{0,10,11\}$.

This code is instantaneous and hence uniquely decodable.

ii. $\{0,01,11\}$.

This code is a suffix code (see problem 11). It is therefore uniquely decodable. The sets in the Sardinas-Patterson test are $S_1 = \{0,01,11\}$, $S_2 = \{1\} = S_3 = S_4 = \cdots$.

iii. $\{0,01,10\}$.

This code is not uniquely decodable. The sets in the test are $S_1 = \{0,01,10\}$, $S_2 = \{1\}$, $S_3 = \{0\}$, $\cdots$. Since 0 is codeword, this code fails the test. It is easy to see otherwise that the code is not UD - the string 010 has two valid parsings.

iv. $\{0,01\}$.

This code is a suffix code and is therefore UD. The test produces sets $S_1 = \{0,01\}$, $S_2 = \{1\}$, $S_3 = \phi$.

v. $\{00,01,10,11\}$.

This code is instantaneous and therefore UD.

vi. $\{110,11,10\}$.

This code is uniquely decodable, by the Sardinas-Patterson test, since $S_1 = \{110,11,10\}$, $S_2 = \{0\}$, $S_3 = \phi$.

vii. $\{110,11,100,00,10\}$.

This code is UD, because by the Sardinas Patterson test, $S_1 = \{110,11,100,00,10\}$, $S_2 = \{0\}$, $S_3 = \{0\}$, etc.

**(d)** For each uniquely decodable code in part (c), construct, if possible, an infinite encoded sequence with a known starting point such that it can be resolved into codewords in two different ways. (This illustrates that unique decodability does not imply finite decodability.) Prove that such a sequence **cannot** arise in a prefix code.

We can produce infinite strings which can be decoded in two ways only for examples where the Sardinas Patterson test produces a repeating set. For example, in part (ii), the string $011111\cdots$ could be parsed either as $0,11,11,\cdots$ or as $01,11,11,\cdots$. Similarly for (vii), the string $10000\cdots$ could be parsed as $100,00,00,\cdots$ or as $10,00,00,\cdots$. For the instantaneous codes, it is not possible to construct such a string, since we can decode as soon as we see a codeword string, and there is no way that we would need to wait to decode.

5.2 KRAFT INEQUALITY

We wish to construct instantaneous codes of minimum expected length to describe a given source. It is clear that we cannot assign short codewords to all source symbols and still be prefix-free. The set of codeword lengths possible for instantaneous codes is limited by the following inequality.

**Theorem 5.2.1** (***Kraft inequality***) For any instantaneous code (prefix code) over an alphabet of size $D$, the codeword lengths $l_1, l_2, \cdots, l_m$ must satisfy the inequality

$$\sum_i D^{-l_i} \leq 1.$$

Conversely, given a set of codeword lengths that satisfy this inequality, there exists an instantaneous code with these word lengths.

**Proof**: Consider a $D$-ary tree in which each node has $D$ children. Let the branches of the tree represent the

symbols of the codeword. For example, the $D$ branches arising from the root node represent the $D$ possible values of the first symbol of the codeword. Then each codeword is represented by a leaf on the tree. The path from the root traces out the symbols of the codeword. A binary ($D = 2$) example of such a tree is shown in Figure 5.3. The prefix condition on the codewords implies that no codeword is an ancestor of any other codeword on the tree. Hence, each codeword eliminates its descendants as possible codewords.

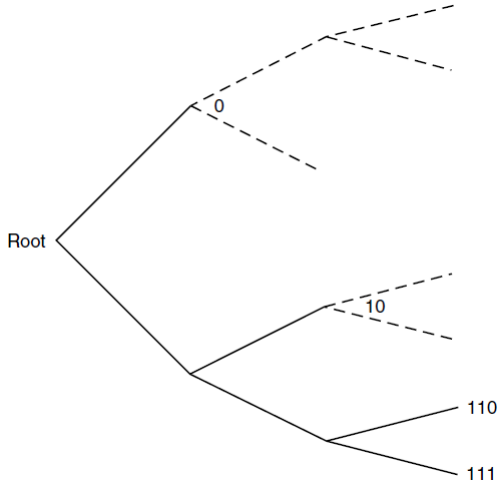



Figure 5. 3 Code tree for the Kraft inequality.

Let $l_{\max}$ be the length of the longest codeword of the set of codewords. Consider all nodes of the tree at level $l_{\max}$. Some of them are codewords, some are descendants of codewords, and some are neither. A codeword at level $l_i$ has $D^{l_{\max}-l_i}$ descendants at level $l_{\max}$. Each of these descendant sets must be disjoint. Also, the total number of nodes in these sets must be less than or equal to $D^{l_{\max}}$. Hence, summing over all the codewords, we have

$$\sum D^{l_{\max}-l_i} \leq D^{l_{\max}}$$

or

$$\sum D^{-l_i} \leq 1,$$

which is the Kraft inequality.

Conversely, given any set of codeword lengths $l_1, l_2, \cdots, l_m$ that satisfy the Kraft inequality, we can always construct a tree like the one in Figure 5.3. Label the first node (lexicographically) of depth $l_1$ as codeword 1, and remove its descendants from the tree. Then label the first remaining node of depth $l_2$ as codeword 2, and so on. Proceeding this way, we **construct** a prefix code with the specified $l_1, l_2, \cdots, l_m$.

**Theorem 5.2.2** (*Extended Kraft Inequality*) For any countably infinite set of codewords that form a prefix code, the codeword lengths satisfy the extended Kraft inequality,

$$\sum_{i=1}^{\infty} D^{-l_i} \leq 1.$$

Conversely, given any $l_1, l_2, \cdots$ satisfying the extended Kraft inequality, we can construct a prefix code with these codeword lengths.

In Section 5.5 we show that the lengths of codewords for a uniquely decodable code also satisfy the Kraft inequality. Before we do that, we consider the problem of finding the shortest instantaneous code.

5.3 OPTIMAL CODES

In Section 5.2 we proved that any codeword set that satisfies the prefix condition has to satisfy the Kraft inequality and that the Kraft inequality is a **sufficient condition** for the **existence** of a codeword set with the specified set of codeword lengths. We now consider the problem of finding the prefix code with the minimum expected length. From the results of Section 5.2, this is equivalent to finding the set of lengths $l_1$,

$l_2, \cdots, l_m$ satisfying the Kraft inequality and whose expected length $L = \sum p_i l_i$ is less than the expected length of any other prefix code. This is a standard optimization problem: Minimize

$$L = \sum p_i l_i$$

over all integers $l_1, l_2, \cdots, l_m$ satisfying

$$\sum D^{-l_i} \leq 1.$$

(Page 110)
We neglect the integer constraint on $l_i$ and will get

$$p_i = D^{-l_i},$$

yielding optimal code lengths,

$$l_i^* = -\log_D p_i.$$

This noninteger choice of codeword lengths yields expected codeword length

$$L^* = \sum p_i l_i^* = -\sum p_i \log_D p_i = H_D(X).$$

But since the $l_i$ must be integers, we will not always be able to set the codeword lengths as in $l_i^* = -\log_D p_i$. Instead, we should choose a set of codeword lengths $l_i$ "close" to the optimal set.

**Theorem 5.3.1** The expected length $L$ of any instantaneous $D$-ary code for a random variable $X$ is greater than or equal to the entropy $H_D(X)$; that is

$$L \geq H_D(X),$$

with equality if and only if $D^{-l_i} = p_i$. (i.e., if and only if $-\log_D p_i$ is an integer for all $i$).

**Definition** A probability distribution is called $D - adic$ if each of the probabilities is equal to $D^{-n}$ for some $n$. Thus, we have equality in the theorem if and only if the distribution of $X$ is $D - adic$.

5.4 BOUNDS ON THE OPTIMAL CODE LENGTH

We now demonstrate a code that achieves an expected description length $L$ within 1 bit of the lower bound; that is, (Page 112)

$$H(X) \leq L < H(X) + 1.$$

For remember: The choice of word lengths $l_i = \log_D \frac{1}{p_i}$ yields $L = H$. Since $\log_D \frac{1}{p_i}$ may not equal an integer, we round it up to give integer word-length assignments,

$$l_i = \left\lceil \log_D \frac{1}{p_i} \right\rceil,$$

This choice of codeword lengths satisfies

$$\log_D \frac{1}{p_i} \leq l_i < \log_D \frac{1}{p_i} + 1$$

Multiplying by $p_i$ and summing over $i$, we obtain

$$H_D(X) \leq L < H_D(X) + 1$$

**Theorem 5.4.1** Let $l_1^*, l_2^*, \cdots, l_m^*$ be optimal codeword lengths for a source distribution $\mathbf{p}$ and a $D$-ary alphabet, and let $L^*$ be the associated expected length of an optimal code ($L^* = \sum p_i l_i^*$). Then

$$H_D(X) \leq L^* < H_D(X) + 1.$$

In Theorem 5.4.1 there is an overhead that is at most 1 bit, due to the fact that $\log \frac{1}{p_i}$ is not always an integer.

We can reduce the overhead per symbol by **spreading** it out **over** many **symbols**: (Page 113)
We send a sequence of $n$ symbols from $X$. The symbols are assumed to be drawn **i.i.d**.
Define $L_n$ to be the expected codeword length per input symbol,

$$L_n = \frac{1}{n} \sum p(x_1, x_2, \cdots, x_n) l(x_1, x_2, \cdots, x_n) = \frac{1}{n} E[l(x_1, x_2, \cdots, x_n)].$$

$$H(X_1, X_2, \cdots, X_n) \leq E[l(x_1, x_2, \cdots, x_n)] < H(X_1, X_2, \cdots, X_n) + 1.$$

Dividing above by $n$,

$$H(X) \leq L_n < H(X) + \frac{1}{n}.$$

Hence, by using large block lengths we can achieve an expected code-length per symbol arbitrarily close

to the entropy.

We can use the same argument for a sequence of symbols from a stochastic process that is not necessarily i.i.d. In this case, we still have the bound

$$H(X_1, X_2, \cdots, X_n) \leq E[l(x_1, x_2, \cdots, x_n)] < H(X_1, X_2, \cdots, X_n) + 1.$$

$$\frac{H(X_1, X_2, \cdots, X_n)}{n} \leq L_n < \frac{H(X_1, X_2, \cdots, X_n)}{n} + \frac{1}{n}.$$

If the stochastic process is stationary, then $H(X_1, X_2, \cdots, X_n)/n \to H(\mathcal{X})$, and the expected description length tends to the entropy rate as $n \to \infty$. Thus, we have the following theorem:

**Theorem 5.4.2** The *minimum* expected codeword length per symbol satisfies

$$\frac{H(X_1, X_2, \cdots, X_n)}{n} \leq L_n^* < \frac{H(X_1, X_2, \cdots, X_n)}{n} + \frac{1}{n}.$$

Moreover, if $X_1, X_2, \cdots, X_n$ is a stationary stochastic process,

$$L_n^* \to H(\mathcal{X}),$$

where $H(\mathcal{X})$ is the entropy rate of the process.

Finally, we ask what happens to the expected description length if the code is designed for the wrong distribution. For example, the wrong distribution may be the best estimate that we can make of the unknown true distribution. We consider the Shannon code assignment $l(x) = \left\lceil \log \frac{1}{q(x)} \right\rceil$ designed for the probability mass function $q(x)$. Suppose that the true probability mass function is $p(x)$. (Page 115)

**Theorem 5.4.3** (***Wrong code***) The expected length under $p(x)$ of the code assignment $l(x) = \left\lceil \log \frac{1}{q(x)} \right\rceil$ satisfies

$$H(p) + D(p \parallel q) \leq E_p[l(X)] < H(p) + D(p \parallel q) + 1.$$

Thus, believing that the distribution is $q(x)$ when the true distribution is $p(x)$ incurs a penalty of $D(p \parallel q)$ in the average description length.

5.5 KRAFT INEQUALITY FOR UNIQUELY DECODABLE CODES

We have proved that any instantaneous code must satisfy the Kraft inequality. The class of uniquely decodable codes is larger than the class of instantaneous codes, so one expects to achieve a lower expected codeword length if $L$ is minimized over all uniquely decodable codes. In this section we prove that the class of uniquely decodable codes does not offer any further possibilities for the set of codeword lengths than do instantaneous codes.

**Theorem 5.5.1** (*McMillan*) The codeword lengths of any uniquely decodable $D - ary$ code must satisfy the Kraft inequality

$$\sum_i D^{-l_i} \leq 1.$$

Conversely, given a set of codeword lengths that satisfy this inequality, it is possible to construct a uniquely decodable code with these codeword lengths.

**Corollary** A uniquely decodable code for an infinite source alphabet $\mathcal{X}$ also satisfies the Kraft inequality.

The theorem implies a rather surprising result-that the class of uniquely decodable codes does not offer any further choices for the set of codeword lengths than the class of prefix codes. The set of achievable codeword lengths is the same for uniquely decodable and instantaneous codes. Hence, the bounds derived on the optimal codeword lengths continue to hold even when we expand the class of allowed codes to the class of all uniquely decodable codes.

5.6 HUFFMAN CODES

An optimal (shortest expected length) prefix code for a given distribution can be constructed by a simple algorithm discovered by Huffman. We will prove that any **other code** for the **same** alphabet cannot have a **lower** expected length than the code constructed by the algorithm.

**Example 5.6.1**: Consider a random variable $X$ taking values in the set $\mathcal{X} = \{1,2,3,4,5\}$ with probabilities $0.25, 0.25, 0.2, 0.15, 0.15$, respectively. We expect the optimal binary code for $X$ to have the longest codewords assigned to the symbols 4 and 5. These two lengths must be equal, since otherwise we can

delete a bit from the longer codeword and still have a prefix code, but with a shorter expected length. In general, we can construct a code in which the two longest codewords differ only in the last bit. For this code, we can combine the symbols 4 and 5 into a single source symbol, with a probability assignment 0.30. Proceeding this way, combining the two least likely symbols into one symbol until we are finally left with only one symbol, and then assigning codewords to the symbols, we obtain the following table:

| Codeword Length | Codeword | $X$ | Probability |
|---|---|---|---|
| 2 | 01 | 1 | 0.25 0.3 0.45 0.55 1 |
| 2 | 10 | 2 | 0.25 0.25 0.3 0.45 |
| 2 | 11 | 3 | 0.2 0.25 0.25 |
| 3 | 000 | 4 | 0.15 0.2 |
| 3 | 001 | 5 | 0.15 |

This code has average length 2.3 bits.

**Example 5.6.2**: Consider a ternary code for the same random variable. Now we combine the three least likely symbols into one supersymbol and obtain the following table:

| Codeword | $X$ | Probability |
|---|---|---|
| 1 | 1 | 0.25 0.5 1 |
| 2 | 2 | 0.25 0.25 |
| 00 | 3 | 0.2 0.25 |
| 01 | 4 | 0.15 |
| 02 | 5 | 0.15 |

This code has an average length of 1.5 ternary digits.

**Example 5.6.3**: If $D \geq 3$, we may not have a sufficient number of symbols so that we can combine them $D$ at a time. In such a case, we add dummy symbols to the end of the set of symbols. The dummy symbols have probability 0 and are inserted to fill the tree. Since at each stage of the reduction, the number of symbols is reduced by $D-1$, we want the total number of symbols to be $1+k(D-1)$, where $k$ is the number of merges. Hence, we add enough dummy symbols so that the total number of symbols is of this form. For example:

| Codeword | $X$ | Probability |
|---|---|---|
| 1 | 1 | 0.25 0.25 0.5 1.0 |
| 2 | 2 | 0.25 0.25 0.25 |
| 01 | 3 | 0.2 0.2 0.25 |
| 02 | 4 | 0.1 0.2 |
| 000 | 5 | 0.1 0.1 |
| 001 | 6 | 0.1 |
| 002 | Dummy | 0.0 |

This code has an average length of 1.7 ternary digits.

5.7 SOME COMMENTS ON HUFFMAN CODES

1. *Equivalence of source coding and 20 questions*. (a sequence of questions is equivalent to a code for the object.)

The expected number of questions $E[Q]$ in this optimal scheme satisfies

$$H(X) \leq E[Q] < H(X) + 1.$$

2. *Huffman coding for weighted codewords*. Huffman's algorithm for minimizing $\sum p_i l_i$ can be applied to any set of numbers $p_i \geq 0$, regardless of $\sum p_i$. In this case, the Huffman code minimizes the sum of weighted code lengths $\sum w_i l_i$ rather than the average code length.

**Example 5.7.1**: We perform the weighted minimization using the same algorithm.

| $X$ | Codeword | Weights |
|---|---|---|
| 1 | 00 | 5 8 10 18 |
| 2 | 01 | 5 5 8 |
| 3 | 10 | 4 5 |
| 4 | 11 | 4 |

In this case the code minimizes the weighted sum of the codeword lengths, and the minimum weighted sum is 36.

3. *Huffman coding and "**slice**" questions* (Alphabetic codes).

4. *Huffman codes and Shannon codes*. Using codeword lengths of $\left\lceil \log \frac{1}{p_i} \right\rceil$ (which is called Shannon coding) may be much worse than the optimal code for some particular symbol.

Although either the Shannon code or the Huffman code can be shorter for individual symbols, the Huffman code is **shorter on average**. Also, the Shannon code and the Huffman code differ by less than 1 bit in expected codelength (since both lie between $H$ and $H + 1$.)

5. *Fano codes*.

5.8 OPTIMALITY OF HUFFMAN CODES

We prove by induction that the binary Huffman code is optimal. It is important to remember that there are **many optimal codes**: inverting all the bits or exchanging two codewords of the same length will give another optimal code. The Huffman procedure constructs one such optimal code.

Without loss of generality, we will assume that the probability masses are ordered, so that $p_1 \geq p_2 \geq \cdots p_m$. Recall that a code is optimal if $\sum p_i l_i$ is minimal. (And prefix code)

**Lemma 5.8.1** For any distribution, there exists an optimal instantaneous code (with minimum expected length) that satisfies the following properties:

1. The lengths are ordered inversely with the probabilities (i.e., if $p_j > p_k$, then $l_j \leq l_k$).
2. The two **longest** codewords have the same **length**.
3. Two of the longest codewords **differ** only in the last bit and correspond to the two least likely symbols.

**Proof**:

The proof amounts to swapping, trimming, and rearranging, as shown in Figure 5.4. Consider an optimal code $C_m$:

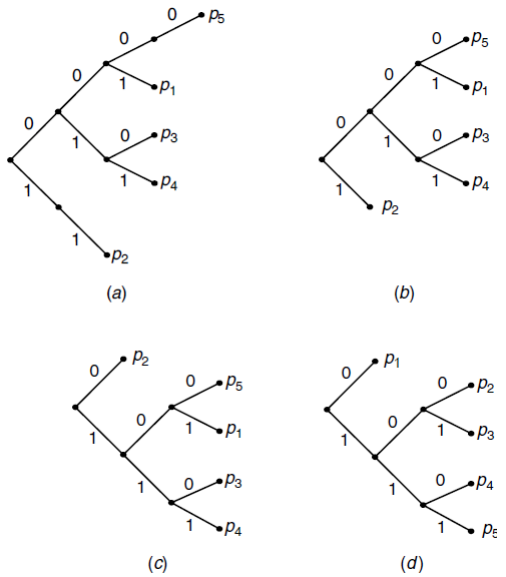



Figure 5. 4 Properties of optimal codes. We assume that $p_1 \geq p_2 \geq \cdots \geq p_m$. A possible instantaneous code is given in (a). By trimming branches without siblings, we improve the code to (b). We now rearrange the tree as shown in (c), so that the word lengths are ordered by increasing length from top to bottom. Finally, we swap probability assignments to improve the expected depth of the tree, as shown in (d). Every optimal code can be rearranged and swapped into canonical form as in (d), where $l_1 \leq l_2 \leq \cdots \leq l_m$ and $l_{m-1} = l_m$, and the last two codewords differ only in the last bit.

• If $p_j > p_k$, then $l_j \leq l_k$. Here we swap codewords. Consider $C'_m$, with the codewords $j$ and $k$ of $C_m$ interchanged. Then

$$L(C'_m) - L(C_m) = \sum p_i l'_i - \sum p_i l_i$$

$$= p_j l_k + p_k l_j - p_j l_j - p_k l_k$$
$$= (p_j - p_k)(l_k - l_j).$$

But $p_j - p_k > 0$, and since $C_m$ is optimal, $L(C'_m) - L(C_m) \geq 0$. Hence, we must have $l_k \geq l_j$. Thus, $C_m$ itself satisfies property 1.

• *The two longest codewords are of the same length*. Here we trim the codewords. If the two longest codewords are not of the same length, one can delete the last bit of the longer one, preserving the prefix property and achieving lower expected codeword length. Hence, the two longest codewords must have the same length. By property 1, the longest codewords must belong to the least probable source symbols.

• *The two longest codewords differ only in the last bit and correspond to the two least likely symbols*. Not all optimal codes satisfy this property, but by rearranging, we can find an optimal code that does. If there is a maximal-length codeword without a sibling, we can delete the last bit of the codeword and still satisfy the prefix property. This reduces the average codeword length and contradicts the optimality of the code. Hence, every maximal-length codeword in any optimal code has a sibling. Now we can exchange the longest codewords so that the two lowest-probability source symbols are associated with two siblings on the tree. This does not change the expected length, $\sum p_i l_i$. Thus, the codewords for the two lowest-probability source symbols have maximal length and agree in all but the last bit.

Summarizing, we have shown that if $\mathrm{p}_1 \geq \mathrm{p}_2 \geq \cdots \geq \mathrm{p}_\mathrm{m}$, there exists an optimal code with $\mathrm{l}_1 \leq \mathrm{l}_2 \leq \cdots \leq \mathrm{l}_{\mathrm{m}-1} = \mathrm{l}_\mathrm{m}$, and codewords $L(x_{m-1})$ and $L(x_m)$ that differ only in the last bit.

Thus, we have shown that there exists an optimal code satisfying the properties of the lemma. We call such codes *canonical codes*.

The proof of optimality: (Page 125)

**Theorem 5.8.1** Huffman coding is optimal; that is, if $C^*$ is a Huffman code and $C'$ is any other uniquely decodable code, $L(C^*) \leq L\left(C'\right)$.

**PROBLEMS**

5.4 *Huffman coding*. Consider the random variable

$$X = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ 0.49 & 0.26 & 0.12 & 0.04 & 0.04 & 0.03 & 0.02 \end{pmatrix}$$

(a) Find a binary Huffman code for $X$.

| Codeword | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | $x_1$ | 0.49 | 0.49 | 0.49 | 0.49 | 0.49 | 0.51 | 1 |
| 00 | $x_2$ | 0.26 | 0.26 | 0.26 | 0.26 | 0.26 | 0.49 | |
| 011 | $x_3$ | 0.12 | 0.12 | 0.12 | 0.13 | 0.25 | | |
| 01000 | $x_4$ | 0.04 | 0.05 | 0.08 | 0.12 | | | |
| 01001 | $x_5$ | 0.04 | 0.04 | 0.05 | | | | |
| 01010 | $x_6$ | 0.03 | 0.04 | | | | | |
| 01011 | $x_7$ | 0.02 | | | | | | |

(b) Find the expected code length for this encoding.

The expected length of the codewords for the binary Huffman code is 2.02 bits. ($H(X) = 2.01$ bits)

(c) Find a ternary Huffman code for $X$.

| Codeword | | | | | |
|---|---|---|---|---|---|
| 0 | $x_1$ | 0.49 | 0.49 | 0.49 | 1.0 |
| 1 | $x_2$ | 0.26 | 0.26 | 0.26 | |
| 20 | $x_3$ | 0.12 | 0.12 | 0.25 | |
| 22 | $x_4$ | 0.04 | 0.09 | | |
| 210 | $x_5$ | 0.04 | 0.04 | | |
| 211 | $x_6$ | 0.03 | | | |
| 212 | $x_7$ | 0.02 | | | |

This code has an expected length 1.34 ternary symbols. ($H_3(X) = 1.27$ ternary symbols).

5.6 *Bad codes*. Which of these codes cannot be Huffman codes for any probability assignment?

(a) {0,10,11}

$\{0,10,11\}$ is a Huffman code for the distribution $(1/2,1/4,1/4)$.

(b) $\{00,01,10,110\}$

The code $\{00,01,10,110\}$ can be shortened to $\{00,01,10,11\}$ without losing its instantaneous property, and therefore is not optimal, so it cannot be a Huffman code. Alternatively, it is not a Huffman code because there is a unique longest codeword.

(c) $\{01,10\}$

The code $\{01,10\}$ can be shortened to $\{0,1\}$ without losing its instantaneous property, and therefore is not optimal and not a Huffman code.

5.8 *Simple optimum compression* of a Markov source. Consider the three-state Markov process $U_1, U_2, \cdots$ having transition matrix

| $U_{n-1}$ \ $U_n$ | $S_1$ | $S_2$ | $S_3$ |
|---|---|---|---|
| $S_1$ | $\frac{1}{2}$ | $\frac{1}{4}$ | $\frac{1}{4}$ |
| $S_2$ | $\frac{1}{4}$ | $\frac{1}{2}$ | $\frac{1}{4}$ |
| $S_3$ | $0$ | $\frac{1}{2}$ | $\frac{1}{2}$ |

Thus, the probability that $S_1$ follows $S_3$ is equal to zero. Design three codes $C_1$, $C_2$, $C_3$ (one for each state 1, 2 and 3, each code mapping elements of the set of $S_i$'s into sequences of 0's and 1's, such that this Markov process can be sent with maximal compression by the following scheme:

(a) Note the present symbol $X_n = i$.

(b) Select code $C_i$.

(c) Note the next symbol $X_{n+1} = j$ and send the codeword in $C_i$ corresponding to $j$.

(d) Repeat for the next symbol. What is the average message length of the next symbol conditioned on the previous state $X_n = i$ using this coding scheme? What is the unconditional average number of bits per source symbol? Relate this to the entropy rate $H(\mathcal{U})$ of the Markov chain.

It is easy to design an optimal code for each state. A possible solution is

| Next state | $S_1$ | $S_2$ | $S_3$ | |
|---|---|---|---|---|
| Code $C_1$ | 0 | 10 | 11 | $E[L(U_n\|U_{n-1} = S_1)] = 1.5$ bits/symbol |
| Code $C_2$ | 10 | 0 | 11 | $E[L(U_n\|U_{n-1} = S_2)] = 1.5$ bits/symbol |
| Code $C_3$ | – | 0 | 1 | $E[L(U_n\|U_{n-1} = S_3)] = 1$ bits/symbol |

The average message lengths of the next symbol conditioned on the previous state being $S_i$ are just the expected lengths of the codes $C_i$. Note that this code assignment achieves the conditional entropy lower bound.

To find the unconditional average, we have to find the stationary distribution on the states. Let $\mu$ be the stationary distribution. Then

$$\mu = \mu \begin{bmatrix} 1/2 & 1/4 & 1/4 \\ 1/4 & 1/2 & 1/4 \\ 0 & 1/2 & 1/2 \end{bmatrix}$$

We can solve this to find that $\mu = (2/9,4/9,1/3)$. Thus the unconditional average number of bits per source symbol

$$\begin{aligned} \mathrm{E}[L] &= \sum_{i=1}^{3} \mu_i \, E[L(U_n|U_{n-1} = S_i)] \\ &= \frac{2}{9} \times 1.5 + \frac{4}{9} \times 1.5 + \frac{1}{3} \times 1 \\ &= \frac{4}{3} \text{ bits/symbol.} \end{aligned}$$

The entropy rate $\mathcal{H}$ of the Markov chain is

$$\mathcal{H} = H(X_2|X_1)$$

$$= \sum_{i=1}^{3} \mu_i H(X_2|X_1 = S_i)$$
$$= 4/3 \text{ bits/symbol.}$$

Thus the unconditional average number of bits per source symbol and the entropy rate $\mathcal{H}$ of the Markov chain are equal, because the expected length of each code $C_i$ equals the entropy of the state after state $i$, $H(X_2|X_1 = S_i)$, and thus maximal compression is obtained.

## CHAPTER 7 CHANNEL CAPACITY

What do we mean when we say that $A$ communicates with $B$? We mean that the physical acts of $A$ have induced a desired physical state in $B$. This transfer of information is a physical process and therefore is subject to the uncontrollable ambient noise and imperfections of the physical signaling process itself. The communication is **successful** if the receiver $B$ and the transmitter $A$ agree on what was sent.

In this chapter we find the maximum number of distinguishable signals for $n$ uses of a communication channel. This number grows exponentially with $n$, and the exponent is known as the **channel capacity**. The characterization of the channel capacity (the logarithm of the number of distinguishable signals) as the maximum mutual information is the **central** and most famous **success** of information theory.

The mathematical analog of a physical signaling system is shown in Figure 7.1. Source symbols from some finite alphabet are mapped into some sequence of channel symbols, which then produces the output sequence of the channel. The output sequence is random but has a distribution that depends on the input sequence. From the output sequence, we attempt to recover the transmitted message.

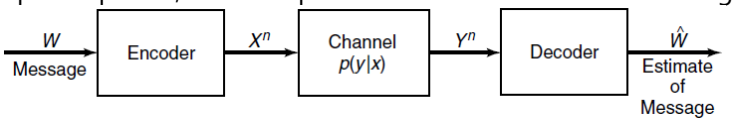



Figure 7. 1 Communication system.

**Each** of the possible input sequences **induces** a **probability distribution** on the output sequences. Since **two different** input sequences may give rise to the **same** output sequence, the inputs are confusable. In the next few sections, we show that we can choose a "nonconfusable" subset of input sequences so that with high probability there is only one highly likely input that could have caused the particular output. We can then **reconstruct** the input sequences at the output with a negligible probability of error. By mapping the source into the appropriate **"widely spaced"** input sequences to the channel, we can transmit a message with very low probability of error and reconstruct the source message at the output. The maximum rate at which this can be done is called the ***capacity*** of the channel.

**Definition** We define a *discrete channel* to be a system consisting of an input alphabet $\mathcal{X}$ and output alphabet $\mathcal{Y}$ and a probability transition matrix $p(y|x)$ that expresses the probability of **observing** the output symbol $y$ given that we **send** the symbol $x$. The channel is said to be ***memoryless*** if the probability distribution of the output depends only on the input **at that time** and is conditionally independent of previous channel inputs or outputs.

**Definition** We define the "*information*" *channel capacity* of a discrete memoryless channel as

$$C = \max_{p(x)} I(X;Y),$$

where the maximum is taken over all possible input distributions $p(x)$.

We shall soon give an operational definition of channel capacity as the **highest rate** in **bits** per channel use at which information can be sent with arbitrarily low probability of error. Shannon's second theorem establishes that the information channel capacity is equal to the operational channel capacity. Thus, we drop the word information in most discussions of channel capacity.

There is a duality between the problems of data compression and data transmission. During compression, we remove all the redundancy in the data to form the most compressed version possible, whereas during data transmission, we add redundancy in a controlled fashion to combat errors in the channel.

7.1 EXAMPLES OF CHANNEL CAPACITY

• Noiseless Binary Channel

Suppose that we have a channel whose the binary input is reproduced exactly at the output (Figure 7.2).

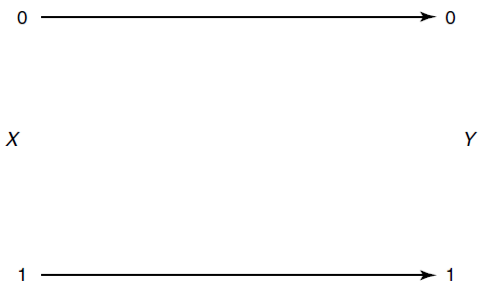


Figure 7. 2 Noiseless binary channel. $C = 1$ bit.

In this case, any transmitted bit is received without error. Hence, one error-free bit can be transmitted per use of the channel, and the capacity is 1 bit. We can also calculate the information capacity $C = \max I(X;Y) = 1$ bit, which is achieved by using $p(x) = \left(\frac{1}{2}, \frac{1}{2}\right)$. ($I(X;Y) = H(Y) - H(Y|X) = 1 - 0$, $P(Y) = \sum_{x_i} P(Y|X)P(X)$)

• Noisy Channel with Nonoverlapping Outputs

This channel has two possible outputs corresponding to each of the **two** inputs (Figure 7.3). The channel appears to be noisy, **but really** is not. Even though the output of the channel is a random consequence of the input, the input can be determined from the output (100%), and hence every transmitted bit can be recovered without error. The capacity of this channel is also 1 bit per transmission. We can also calculate the information capacity $C = \max I(X;Y) = 1$ bit, which is achieved by using $p(x) = \left(\frac{1}{2}, \frac{1}{2}\right)$.

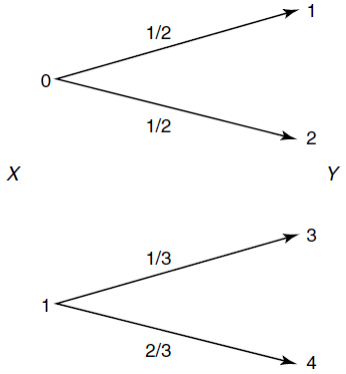


Figure 7. 3 Noisy channel with nonoverlapping outputs. $C = 1$ bit.

**Proof**:

We know, (when input is uniformly distribution)

$$(x, y) = \{(0,1), (0,2), (1,3), (1,4)\},$$

and

$$p(x = 0, y = 1) = p(y = 1|x = 0)p(x = 0) = \frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$$
$$p(x = 0, y = 2) = p(y = 2|x = 0)p(x = 0) = \frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$$
$$p(x = 1, y = 3) = p(y = 3|x = 1)p(x = 1) = \frac{1}{3} \times \frac{1}{2} = \frac{1}{6}$$
$$p(x = 1, y = 4) = p(y = 4|x = 1)p(x = 1) = \frac{2}{3} \times \frac{1}{2} = \frac{1}{3}$$

Then,

$$p(y = 1) = p(y = 1|x = 0)p(x = 0) + p(y = 1|x \neq 0)p(x \neq 0) = p(x = 0, y = 1)$$
$$p(y = 2) = p(x = 0, y = 2)$$
$$p(y = 3) = p(x = 1, y = 3)$$
$$p(y = 4) = p(x = 1, y = 4)$$

Note here we can rewrite $p(x=0,y=1)$ as $p(x=0,y=1)=p(x=0|y=1)p(y=1)=1\times\frac{1}{4}=\frac{1}{4}$.
Finally,

$$C=\max I(X;Y)=\frac{1}{4}\log\frac{\frac{1}{4}}{\frac{1}{2}\times\frac{1}{4}}+\frac{1}{4}\log\frac{\frac{1}{4}}{\frac{1}{2}\times\frac{1}{4}}+\frac{1}{6}\log\frac{\frac{1}{6}}{\frac{1}{2}\times\frac{1}{6}}+\frac{1}{3}\log\frac{\frac{1}{3}}{\frac{1}{2}\times\frac{1}{3}}=1.$$

• Noisy Typewriter

In this case the channel input is either received unchanged at the output with probability $\frac{1}{2}$ or is transformed into the next letter with probability $\frac{1}{2}$ (Figure 7.4). If the input has 26 symbols and we use every alternate input symbol, we can transmit one of 13 symbols without error with each transmission (only send 13 symbols: $\frac{\text{total number of } Y}{\text{total number of } Y \text{ given } X}=\frac{26}{2}$). Hence, the capacity of this channel is $\log 13$ bits per transmission. We can also calculate the information capacity $C=\max I(X;Y)=\max\big(H(Y)-H(Y|X)\big)=\max H(Y)-1=\log 26-1=\log 13$, achieved by using $p(x)$ distributed uniformly over all the inputs. (proof is same as above)

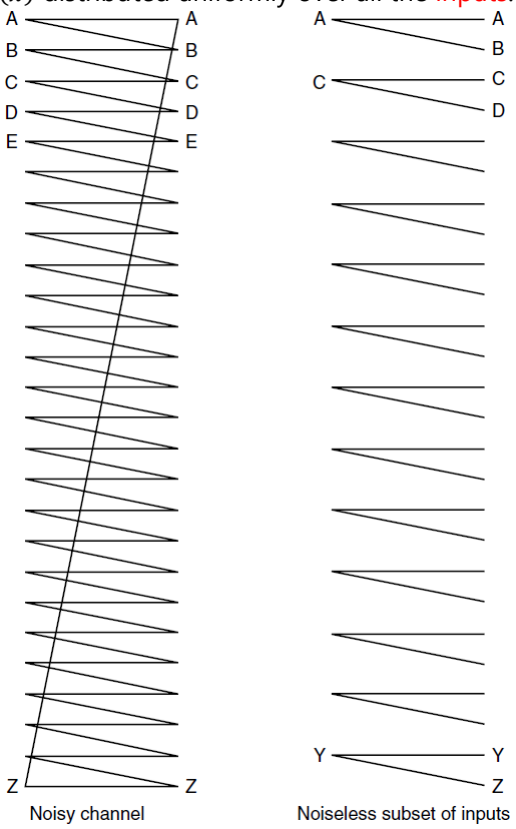


Figure 7. 4 Noisy Typewriter. $C=\log 13$ bits.

• Binary Symmetric Channel

Consider the binary symmetric channel (BSC), which is shown in Fig. 7.5. This is a binary channel in which the input symbols are complemented with probability $p$. This is the simplest model of a channel with errors, yet it **captures most of** the **complexity** of the general problem.

When an error occurs, a 0 is received as a 1, and **vice versa**. The bits received do not reveal where the errors have occurred. In a sense, all the bits received are unreliable. Later we show that we can still use such a communication channel to send information at a nonzero rate with an arbitrarily small probability of error. We bound the mutual information by

$$I(X;Y)=H(Y)-H(Y|X)$$
$$=H(Y)-\sum p(x)H(Y|X=x)$$

$$= H(Y) - \sum p(x)H(p)$$
$$= H(Y) - H(p)$$
$$\leq 1 - H(p),$$

where the last inequality follows because $Y$ is a binary random variable. Equality is achieved when the input distribution is uniform. (proof as above) Hence, the information capacity of a binary symmetric channel with parameter $p$ is

$$C = 1 - H(p) \quad \text{bits.}$$

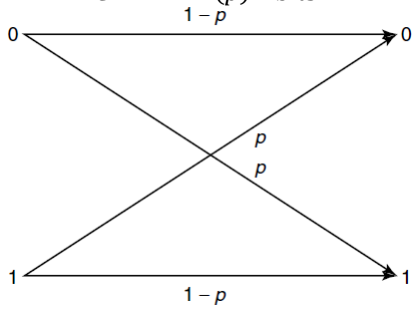



Figure 7. 5 Binary symmetric channel. $C = 1 - H(p)$ bits.

**Proof**:

We know, (when input is uniformly distribution)

$$(x, y) = \{(0,0), (0,1), (1,0), (1,1)\},$$

and

$$p(x = 0, y = 0) = p(y = 0|x = 0)p(x = 0) = (1 - p) \times \frac{1}{2} = \frac{(1-p)}{2}$$
$$p(x = 0, y = 1) = p(y = 1|x = 0)p(x = 0) = p \times \frac{1}{2} = \frac{p}{2}$$
$$p(x = 1, y = 0) = p(y = 0|x = 1)p(x = 1) = p \times \frac{1}{2} = \frac{p}{2}$$
$$p(x = 1, y = 1) = p(y = 1|x = 1)p(x = 1) = (1 - p) \times \frac{1}{2} = \frac{(1-p)}{2}$$

Then,

$$p(y = 0) = p(y = 0|x = 0)p(x = 0) + p(y = 0|x = 1)p(x = 1) = (1 - p) \times \frac{1}{2} + p \times \frac{1}{2} = \frac{1}{2}$$
$$p(y = 0) = \frac{1}{2}$$

Note here we can rewrite $p(x = 0, y = 0)$ as $p(x = 0, y = 0) = p(x = 0|y = 0)p(y = 0) = (1 - p) \times \frac{1}{2} = \frac{(1-p)}{2}$.

Finally,

$$C = \max I(X;Y) = \frac{(1-p)}{2}\log\frac{\frac{(1-p)}{2}}{\frac{1}{2}\times\frac{1}{2}} + \frac{p}{2}\log\frac{\frac{p}{2}}{\frac{1}{2}\times\frac{1}{2}} + \frac{p}{2}\log\frac{\frac{p}{2}}{\frac{1}{2}\times\frac{1}{2}} + \frac{(1-p)}{2}\log\frac{\frac{(1-p)}{2}}{\frac{1}{2}\times\frac{1}{2}}$$
$$= (1-p)\log 2(1-p) + p\log 2p$$
$$= 1 - p + p + (1-p)\log(1-p) + p\log p$$
$$= 1 - H(p).$$

• Binary Erasure Channel

The analog of the binary symmetric channel in which some bits are lost (rather than corrupted) is the *binary erasure channel*. In this channel, a fraction $\alpha$ of the bits are erased. The receiver knows which bits have been erased. The binary erasure channel has two inputs and three outputs (Figure 7.6).

We calculate the capacity of the binary erasure channel as follows:

$$C = \max_{p(x)} I(X;Y)$$
$$= \max_{p(x)} \big(H(Y) - H(Y|X)\big)$$

$$= \max_{p(x)} H(Y) - H(\alpha).$$

The first guess for the maximum of $H(Y)$ would be $\log 3$, but we cannot achieve this by any choice of input distribution $p(x)$. Since

$$p(y=0) = p(y=0|x=0)p(x=0)$$
$$p(y=e) = p(y=e|x=0)p(x=0) + p(y=e|x=1)p(x=1)$$
$$p(y=1) = p(y=1|x=1)p(x=1)$$

and letting $\Pr(X=1) = \pi$, we have

$$H(Y) = H\big((1-\pi)(1-\alpha), \alpha, \pi(1-\alpha)\big) = H(\alpha) + (1-\alpha)H(\pi).$$

Hence

$$C = \max_{p(x)} H(Y) - H(\alpha)$$
$$= \max_{\pi}(1-\alpha)H(\pi) + H(\alpha) - H(\alpha)$$
$$= \max_{\pi}(1-\alpha)H(\pi)$$
$$= 1-\alpha.$$

where capacity is achieved by $\Pr(X=1) = \frac{1}{2}$. (Proof is same as above)

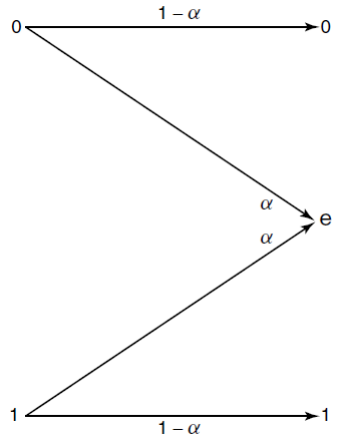



Figure 7. 6 Binary erasure channel.

The expression for the capacity has some intuitive meaning: Since a proportion $\alpha$ of the bits are lost in the channel, we can recover (at most) a proportion $1-\alpha$ of the bits. Hence the capacity is at most $1-\alpha$. It is not immediately obvious that it is possible to achieve this rate. This will follow from Shannon's second theorem.

In many practical channels, the sender receives some feedback from the receiver. If feedback is available for the binary erasure channel, it is very clear what to do: If a bit is lost, retransmit it until it gets through. Since the bits get through with probability $1-\alpha$, the effective rate of transmission is $1-\alpha$. In this way we are easily able to achieve a capacity of $1-\alpha$ with feedback.

Later in the chapter we prove that the rate $1-\alpha$ is the best that can be achieved both with and without feedback. This is one of the consequences of the surprising fact that feedback **does not** increase the capacity of discrete memoryless channels.

7.2 SYMMETRIC CHANNELS

The capacity of the binary symmetric channel is $C = 1 - H(p)$ bits per transmission, and the capacity of the binary erasure channel is $C = 1-\alpha$ bits per transmission. Now consider the channel with **transition matrix**:

$$p(y|x) = \begin{bmatrix} 0.3 & 0.2 & 0.5 \\ 0.5 & 0.3 & 0.2 \\ 0.2 & 0.5 & 0.3 \end{bmatrix}.$$

Here the entry in the $x$th row and the $y$th column denotes the conditional probability $p(y|x)$ that $y$ is received when $x$ is sent. In this channel, all the rows of the probability transition matrix are permutations of each other and so are the columns. Such a channel is said to be ***symmetric***. Another example of a symmetric

channel is one of the form

$$Y = X + Z \pmod{c},$$

where $Z$ has some distribution on the integers $\{0,1,2,\cdots,c-1\}$, $X$ has the same alphabet as $Z$, and $Z$ is independent of $X$.

In both these cases, we can easily find an explicit expression for the capacity of the channel. Letting $\mathbf{r}$ be a row of the transition matrix, we have

$$\begin{aligned} I(X;Y) &= H(Y) - H(Y|X) \\ &= H(Y) - H(\mathbf{r}) \\ &\leq \log|\mathcal{Y}| - H(\mathbf{r}) \end{aligned}$$

with equality if the output distribution is uniform. But $p(x) = 1/|\mathcal{X}|$ achieves a uniform distribution on $Y$, as seen from

$$p(y) = \sum_{x\in\mathcal{X}} p(y|x)p(x) = \frac{1}{|\mathcal{X}|}\sum p(y|x) = c\frac{1}{|\mathcal{X}|} = \frac{1}{|\mathcal{Y}|},$$

where $c$ is the sum of the entries in one column of the probability transition matrix.

Thus, the channel above has the capacity

$$C = \max_{p(x)} I(X;Y) = \log 3 - H(0.5,0.3,0.2),$$

and $C$ is achieved by a uniform distribution on the input.

The transition matrix of the symmetric channel defined above is **doubly stochastic**. In the computation of the capacity, we used the facts that the rows were permutations of one another and that all the column sums were equal.

Considering these properties, we can define a generalization of the concept of a symmetric channel as follows:

**Definition** A channel is said to be ***symmetric*** if the rows of the channel transition matrix $p(y|x)$ are permutations of each other and the columns are permutations of each other. A channel is said to be *weakly symmetric* if every row of the transition matrix $p(\cdot|x)$ is a permutation of every other row and all the column sums $\sum_x p(y|x)$ are equal.

For example, the channel with transition matrix

$$p(y|x) = \begin{pmatrix} \frac{1}{3} & \frac{1}{6} & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & \frac{1}{6} \end{pmatrix}$$

is weakly symmetric but not symmetric.

**Theorem 7.2.1** For a weakly symmetric channel,

$$C = \log|\mathcal{Y}| - H(row\ of\ transition\ matrix),$$

and this is achieved by a uniform distribution on the input alphabet.

7.3 PROPERTIES OF CHANNEL CAPACITY

1. $C \geq 0$ since $I(X;Y) \geq 0$.
2. $C \leq \log|\mathcal{X}|$ since $C = \max I(X;Y) \leq \max H(X) = \log|\mathcal{X}|$.
3. $C \leq \log|\mathcal{Y}|$ for the same reason.
4. $I(X;Y)$ is a continuous function of $p(x)$.
5. $I(X;Y)$ is a concave function of $p(x)$.

In general, there is no closed-form solution for the capacity. But for many simple channels it is possible to calculate the capacity using properties such as symmetry. Some of the examples considered earlier are of this form.

7.4 PREVIEW OF THE CHANNEL CODING THEOREM

So far, we have defined the information capacity of a discrete memoryless channel. In the next section we prove Shannon's second theorem, which gives an operational meaning to the definition of capacity as the number of bits we can transmit reliably over the channel. But first we will try to give an intuitive idea as to why we can transmit $C$ bits of information over a channel. The basic idea is that for large block lengths,

every channel looks like the noisy typewriter channel (Figure 7.4) and the channel has a subset of inputs that produce essentially disjoint sequences at the output.

For each (typical) input $n$-sequence, there are approximately $2^{nH(Y|X)}$ possible $Y$ sequences, all of them equally likely (Figure 7.7). We wish to ensure that no two $X$ sequences produce the same $Y$ output sequence. Otherwise, we will not be able to decide which $X$ sequence was sent.

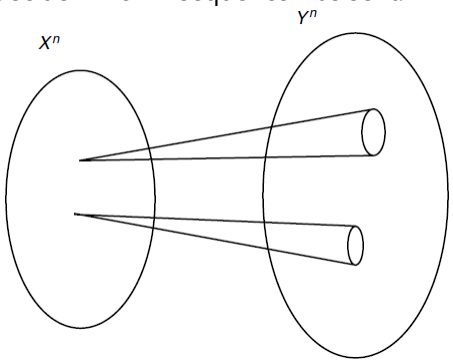



Figure 7. 7 Channels after $n$ uses.

The total number of possible (typical) $Y$ sequences is $\approx 2^{nH(Y)}$. This set has to be divided into sets of size $2^{nH(Y|X)}$ corresponding to the different input $X$ sequences. The total number of disjoint sets is less than or equal to $2^{n(H(Y)-H(Y|X))} = 2^{nI(X;Y)}$. Hence, we can send at most $\approx 2^{nI(X;Y)}$ distinguishable sequences of length $n$.

Although the above derivation outlines an upper bound on the capacity, a stronger version of the above argument will be used in the next section to prove that this rate $I$ is achievable with an arbitrarily low probability of error.

7.5 DEFINITIONS

We analyze a communication system as shown in Figure 7.8.

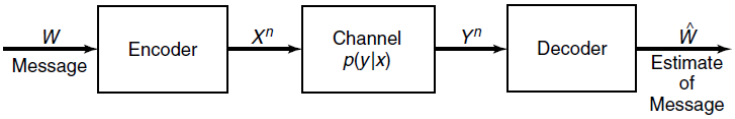



Figure 7. 8 Communication channel.

A message $W$, drawn from the index set $\{1,2,\cdots,M\}$, results in the signal $X^n(W)$, which is received by the receiver as a random sequence $Y^n \sim p(y^n|x^n)$. The receiver **then guesses** the index $W$ by an appropriate decoding rule $\widehat{W} = g(Y^n)$. The receiver makes an error if $\widehat{W}$ is not the same as the index $W$ that was transmitted. We now define these ideas formally.

**Definition** A *discrete channel*, denoted by $(\mathcal{X}, p(y|x), \mathcal{Y})$, consists of two finite sets $\mathcal{X}$ and $\mathcal{Y}$ and a collection of probability mass functions $p(y|x)$, one for each $x \in \mathcal{X}$, such that for every $x$ and $y$, $p(y|x) \geq 0$, and for every $x$, $\sum_y p(y|x) = 1$, with the interpretation that $X$ is the input and $Y$ is the output of the channel.

**Definition** The $n$th extension of the *discrete memoryless channel* (DMC) is the channel $(\mathcal{X}^n, p(y^n|x^n), \mathcal{Y}^n)$, where

$$p(y_k|x^k, y^{k-1}) = p(y_k|x_k), \quad k = 1,2,\cdots,n.$$

**Remark** If the channel is used *without feedback* [i.e., if the input symbols do not depend on the past output symbols, namely, $p(x_k|x^{k-1}, y^{k-1}) = p(x_k|x^{k-1})$], the channel transition function for the $n$th extension of the discrete memoryless channel reduces to

$$p(y^n|x^n) = \prod_{i=1}^{n} p(y_i|x_i).$$

When we refer to the discrete memoryless channel, we mean the discrete memoryless channel without feedback unless we state explicitly otherwise.

**Definition** An $(M, n)$ code for the channel $(\mathcal{X}, p(y|x), \mathcal{Y})$ consists of the following:
1. An index set $\{1,2,\cdots,M\}$.
2. An encoding function $X^n: \{1,2,\cdots,M\} \to \mathcal{X}^n$, yielding codewords $x^n(1)$, $x^n(2)$, $\cdots$, $x^n(M)$. The set of codewords is called the ***codebook***.
3. A decoding function

$$g: \mathcal{Y}^n \to \{1,2,\cdots,M\},$$

which is a deterministic rule that assigns a guess to each possible received vector.

**Definition** (*Conditional probability of error*) Let

$$\lambda_i = \Pr\big(g(Y^n) \neq i | X^n = x^n(i)\big) = \sum_{y^n} p\big(y^n|x^n(i)\big) I(g(y^n) \neq i)$$

be the conditional probability of error given that index $i$ was sent, where $I(\cdot)$ is the indicator function.

**Definition** The maximal probability of error $\lambda^{(n)}$ for an $(M, n)$ code is defined as

$$\lambda^{(n)} = \max_{i \in \{1,2,\cdots,M\}} \lambda_i.$$

**Definition** The (*arithmetic*) average probability of error $P_e^{(n)}$ for an $(M, n)$ code is defined as

$$P_e^{(n)} = \frac{1}{M} \sum_{i=1}^{M} \lambda_i.$$

Note that if the index $W$ is chosen according to a uniform distribution over the set $\{1,2,\cdots,M\}$, and $X^n = x^n(W)$, then

$$P_e^{(n)} \triangleq \Pr\big(W \neq g(Y^n)\big),$$

(i.e., $P_e^{(n)}$ is the probability of error). Also, obviously,

$$P_e^{(n)} \leq \lambda^{(n)}.$$

One would expect the maximal probability of error to behave quite differently from the average probability. But in the next section we prove that a small average probability of error implies a small maximal probability of error at essentially the same rate.

**Definition** The *rate* $R$ of an $(M, n)$ code is

$$R = \frac{\log M}{n} \quad \text{bits per transmission.}$$

**Definition** A rate $R$ is said to be *achievable* if there exists a sequence of $(\lceil 2^{nR} \rceil, n)$ codes such that the maximal probability of error $\lambda^{(n)}$ tends to 0 as $n \to \infty$.

**Definition** The *capacity* of a channel is the supremum of all achievable rates.

Thus, rates less than capacity yield arbitrarily small probability of error for sufficiently large block lengths.

7.6 JOINTLY TYPICAL SEQUENCES

Roughly speaking, we decode a channel output $Y^n$ as the $i$th index if the codeword $X^n(i)$ is "jointly typical" with the received signal $Y^n$. We now define the important idea of joint typicality and find the probability of joint typicality when $X^n(i)$ is the true cause of $Y^n$ and when it is not.

**Definition** The set $A_\epsilon^{(n)}$ of *jointly typical* sequences $\{(x^n, y^n)\}$ with respect to the distribution $p(x, y)$ is the set of $n$-sequences with empirical entropies $\epsilon$-close to the true entropies:

$$A_\epsilon^{(n)} = \Big\{(x^n, y^n) \in \mathcal{X}^n \times \mathcal{Y}^n : \Big|-\frac{1}{n}\log p(x^n) - H(X)\Big| < \epsilon, \Big|-\frac{1}{n}\log p(y^n) - H(Y)\Big| < \epsilon, \Big|-\frac{1}{n}\log p(x^n, y^n) - H(X,Y)\Big| < \epsilon\Big\},$$

where

$$p(x^n, y^n) = \prod_{i=1}^{n} p(x_i, y_i).$$

**Theorem 7.6.1** (*Joint AEP*) Let $(X^n, Y^n)$ be sequences of length $n$ drawn i.i.d. according to $p(x^n, y^n) = \prod_{i=1}^n p(x_i, y_i)$. Then:
1. $\Pr\big((X^n, Y^n) \in A_\epsilon^{(n)}\big) \to 1$ as $n \to \infty$.

2. $\left|A_\epsilon^{(n)}\right| \leq 2^{n(H(X,Y)+\epsilon)}$.
3. If $\left(\tilde{X}^n, \tilde{Y}^n\right) \sim p(x^n)p(y^n)$ [i.e., $\tilde{X}^n$ and $\tilde{Y}^n$ are independent with the same marginals as $p(x^n, y^n)$], then

$$\Pr\left(\left(\tilde{X}^n, \tilde{Y}^n\right) \in A_\epsilon^{(n)}\right) \leq 2^{-n(I(X;Y)-3\epsilon)}.$$

Also, for sufficiently large $n$,

$$\Pr\left(\left(\tilde{X}^n, \tilde{Y}^n\right) \in A_\epsilon^{(n)}\right) \geq (1-\epsilon)2^{-n(I(X;Y)+3\epsilon)}.$$

The jointly typical set is illustrated in Figure 7.9. There are about $2^{nH(X)}$ typical $X$ sequences and about $2^{nH(Y)}$ typical $Y$ sequences. However, since there are only $2^{nH(X,Y)}$ jointly typical sequences, not all pairs of typical $X^n$ and typical $Y^n$ are also jointly typical. The probability that any randomly chosen pair is jointly typical is about $2^{-nI(X;Y)}$. Hence, we can consider about $2^{nI(X;Y)}$ such pairs before we are likely to come across a jointly typical pair. This suggests that there are about $2^{nI(X;Y)}$ distinguishable signals $X^n$.

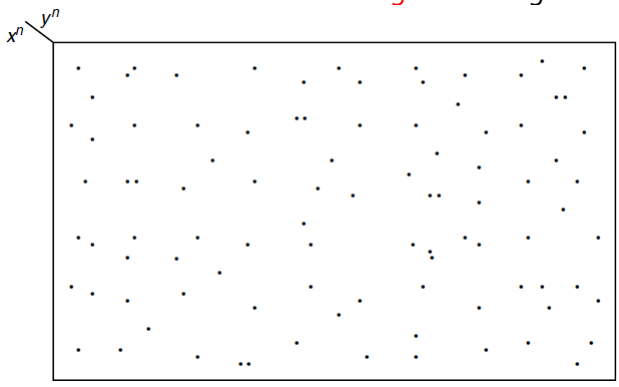



Figure 7. 9 Jointly typical sequences.

Another way to look at this is in terms of the set of jointly typical sequences for a fixed output sequence $Y^n$, presumably the output sequence resulting from the true input signal $X^n$. For this sequence $Y^n$, there are about $2^{nH(X|Y)}$ conditionally typical input signals. The probability that some randomly chosen (other) input signal $X^n$ is jointly typical with $Y^n$ is about $2^{nH(X|Y)}/2^{nH(X)} = 2^{-nI(X;Y)}$. This again suggests that we can choose about $2^{nI(X;Y)}$ codewords $X^n(W)$ before one of these codewords will get confused with the codeword that caused the output $Y^n$.

7.7 CHANNEL CODING THEOREM

We now prove what is perhaps the basic theorem of information theory, the achievability of channel capacity, first stated and essentially proved by Shannon in his original 1948 paper. The result is rather counterintuitive; if the channel introduces errors, how can one correct them all? Any correction process is also subject to error, ad infinitum.

Shannon used a number of new ideas to prove that information can be sent reliably over a channel at **all rates up to** the channel capacity. These ideas include:

- Allowing an arbitrarily small but nonzero probability of error
- Using the channel many times in succession, so that the law of large numbers comes into effect
- Calculating the average of the probability of error over a random choice of codebooks, which symmetrizes the probability, and which can then be used to show the existence of at least one good code

**Theorem 7.7.1** (*Channel coding theorem*) For a discrete memoryless channel, all rates below capacity $C$ are achievable. Specifically, for every rate $R < C$, there exists a sequence of $(2^{nR}, n)$ codes with maximum probability of error $\lambda^{(n)} \to 0$.

Conversely, any sequence of $(2^{nR}, n)$ codes with $\lambda^{(n)} \to 0$ must have $R < C$.

Although the theorem shows that there exist good codes with arbitrarily small probability of error for long block lengths, it does not provide a way of constructing the best codes.

7.8 ZERO-ERROR CODES

7.9 FANO'S INEQUALITY AND THE CONVERSE TO THE CODING THEOREM

7.10 EQUALITY IN THE CONVERSE TO THE CHANNEL CODING THEOREM

7.11 HAMMING CODES

The channel coding theorem promises the existence of block codes that will allow us to transmit information at rates below capacity with an arbitrarily small probability of error if the **block length** is **large enough**. Ever since the appearance of Shannon's original paper [471], people have **searched** for such codes. In addition to achieving low probabilities of error, useful codes should be "simple," so that they can be encoded and decoded efficiently.

The object of coding is to introduce redundancy so that even if some of the information is lost or corrupted, it will still be possible to recover the message at the receiver. The most obvious coding scheme is to repeat information. For example, to send a $1$, we send $11111$, and to send a $0$, we send $00000$. This scheme uses five symbols to send $1$ bit, and therefore has a *rate* of $\frac{1}{5}$ bit per symbol. If this code is used on a binary symmetric channel, the optimum decoding scheme is to take the **majority vote** of each block of five received bits. If three or more bits are $1$, we decode the block as a $1$; otherwise, we decode it as $0$. An error occurs if and only if more than three of the bits are changed. By using **longer** repetition codes, we can achieve an arbitrarily **low** probability of error. But the rate of the code also goes to zero with block length, so even though the code is "simple," it is really **not** a very useful code.

**Instead of** simply repeating the bits, we can combine the bits in some intelligent fashion so that **each extra** bit **checks** whether there is an error in some subset of the information bits. A simple example of this is a **parity check code**. Starting with a block of $n-1$ information bits, we choose the $n$th bit so that the parity of the entire block is $0$ (the number of $1$'s in the block is even). Then if there is an odd number of errors during the transmission, the receiver will notice that the parity has changed and detect the error. This is the simplest example of an **error-detecting code**. The code does not detect an even number of errors and does not give any information about how to **correct** the errors that occur.

| Character | Sender | Parity Bit | Receiver | Parity | |
|---|---|---|---|---|---|
| "E" | 1000101 | 1 | 10001011 | Even | |
| "A" | 1000001 | 0 | 10000010 | Even | |
| "C" | 1000011 | 1 | 11100111 | Even | |
| "q" | 1110001 | 0 | 11100000 | Odd | **Error!** |

We can extend the idea of parity checks to allow for **more than one** parity check bit and to allow the parity checks to depend on **various subsets** of the information bits. The Hamming code that we describe below is **an example** of a parity check code. We describe it using some simple ideas from linear algebra.

To **illustrate** the principles of Hamming codes, we consider a binary code of block length $7$. All operations will be done modulo 2. Consider the set of all nonzero binary vectors of length $3$. Arrange them in columns to form a matrix:

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

Consider the **set** of vectors of length $7$ in the **null space** of $H$ (the vectors which when multiplied by $H$ give $000$). From the theory of linear spaces, since $H$ has rank $3$, we expect the null space of $H$ to have dimension $4$. These $2^4$ codewords are

$$\begin{array}{cccc} 0000000 & 0100101 & 1000011 & 1100110 \\ 0001111 & 0101010 & 1001100 & 1101001 \\ 0010110 & 0110011 & 1010101 & 1110000 \\ 0011001 & 0111100 & 1011010 & 1111111 \end{array}$$

Since the set of codewords is the null space of a matrix, it is *linear* in the sense that the sum of any two codewords is **also** a codeword. The set of codewords therefore forms a **linear subspace** of dimension $4$ in the vector space of dimension $7$.

**Looking** at the codewords, we notice that other than the all-$0$ codeword, the **minimum number of 1's** in any codeword is $3$. This is called the ***minimum weight*** of the code. We can see that the minimum weight of a code has to be **at least** $3$ **since** all the columns of $H$ are different, **so** no two columns can add to $000$. The **fact** that the minimum distance is exactly $3$ can be seen from **the fact** that the sum of any two columns must be one of the columns of the matrix.

Since the code is linear, the difference between any two codewords is **also** a codeword, and hence any two codewords differ in at least three places. The **minimum** number of places in which **two** codewords **differ** is called the ***minimum distance*** of the code. The minimum distance of the code is a **measure** of how far apart the codewords are and will determine how distinguishable the codewords will be at the output of the channel. The minimum distance is **equal** to the minimum weight for a linear code. We aim to develop codes that have a large minimum distance.

For the code described **above**, the minimum distance is $3$. Hence if a codeword $\mathbf{c}$ is corrupted in **only one place**, it will **differ** from any other codeword in **at least** two places and therefore be **closer** to $\mathbf{c}$ than to any other codeword. **But** can we discover which is the closest codeword without searching over all the codewords?

The answer is **yes**. We can use the structure of the matrix $H$ for **decoding**. The matrix $H$, called the ***parity check matrix***, has the property that for every codeword $\mathbf{c}$, $H\mathbf{c} = 0$. Let $\mathbf{e}_i$ be a vector with a $1$ in the $i$th position and $0$'s elsewhere. If the codeword is corrupted at position $i$, the received vector $\mathbf{r} = \mathbf{c} + \mathbf{e}_i$. If we multiply this vector by the matrix $H$, we obtain

$$H\mathbf{r} = H(\mathbf{c} + \mathbf{e}_i) = H\mathbf{c} + H\mathbf{e}_i = H\mathbf{e}_i,$$

which is the vector **corresponding to** the $i$th column of $H$. **Hence** looking at $H\mathbf{r}$, we can **find** which position of the vector was corrupted. **Reversing** this bit will **give** us a codeword. This yields a simple procedure for correcting one error in the received sequence. We have **constructed** a codebook with $16$ codewords of block length $7$, which can correct up to one error. **This code** is called a ***Hamming code***.

We **have not yet** identified a simple encoding procedure; we could use any mapping from a set of $16$ messages into the codewords. But if we examine the first $4$ bits of the codewords in the table, we observe that they cycle through all $2^4$combinations of $4$ bits. Thus, we **could** use these $4$ bits to be the $4$ bits of the message we want to send; the other $3$ bits are then determined by the code. **In general**, it is possible to **modify** a linear code so that the mapping is **explicit**, so that the first $k$ bits in each codeword represent the message, and the last $n - k$ bits are parity check bits. Such a code is called a ***systematic code***. The code is often identified by its block length $n$, the number of information bits $k$ and the minimum distance $d$. For example, the above code is called a $(7,4,3)$ Hamming code ((i.e., $n = 7$, $k = 4$, and $d = 3$).

An easy way to **see how** Hamming codes **work** is by means of a **Venn diagram**. Consider the following Venn diagram with three circles and with four intersection regions as shown in Figure 7.10. To send the information sequence $1101$, we place the $4$ information bits in the four intersection regions as shown in the figure. We then place a parity bit in each of the three remaining regions so that the parity of each circle is even (i.e., there are an even number of $1$'s in each circle). Thus, the parity bits are as shown in Figure 7.11.

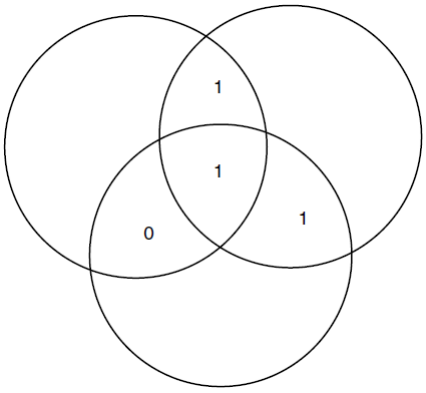



Figure 7. 10 Venn diagram with information bits.

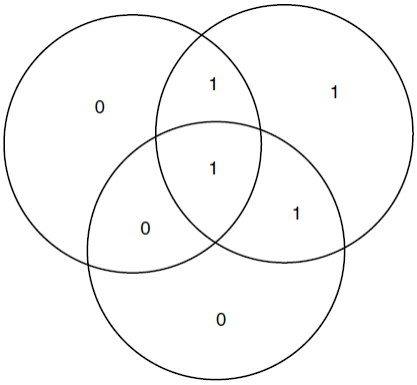



Figure 7. 11 Venn diagram with information bits and parity bits with even parity for each circle.

Now assume that one of the bits is changed; for example one of the information bits is changed from 1 to 0 as shown in Figure 7.12. Then the parity constraints are violated for two of the circles (highlighted in the figure), and it is not hard to see that given these violations, the only single bit error that could have caused it is at the intersection of the two circles (i.e., the bit that was changed). Similarly working through the other error cases, it is not hard to see that this code can detect and correct any single bit error in the received codeword.

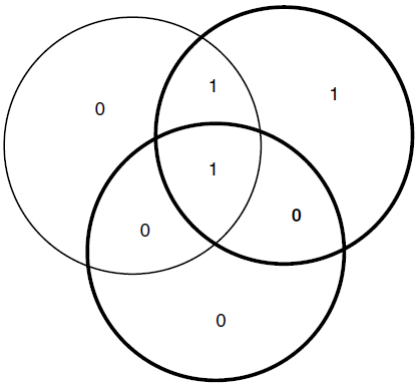



Figure 7. 12 Venn diagram with one of the information bits changed.

We can easily generalize this procedure to construct **larger matrices** $H$. In general, if we use $l$ rows in $H$, the code that we obtain will have block length $n = 2^l - 1$ (Note here, the column vectors need to be different, so row decides column), $k = 2^l - l - 1$ and minimum distance 3. All these codes are called Hamming codes and can correct one error.