1

INTRODUCTION TO INFORMATION THEORY

{ch:intro_info}

This chapter introduces some of the basic concepts of information theory, as well as the definitions and notations of probabilities that will be used throughout the book. The notion of entropy, which is fundamental to the whole topic of this book, is introduced here. We also present the main questions of information theory, data compression and error correction, and state Shannon's theorems.

1.1 Random variables

The main object of this book will be the behavior of large sets of **discrete random variables**. A discrete random variable X is completely defined[1] by the set of values it can take, $\mathcal{X}$, which we assume to be a finite set, and its **probability distribution** $\{p_X(x)\}_{x\in\mathcal{X}}$. The value $p_X(x)$ is the probability that the random variable X takes the value x. The probability distribution $p_X : \mathcal{X} \to [0,1]$ must satisfy the normalization condition

$$\sum_{x\in\mathcal{X}} p_X(x) = 1 \ . \tag{1.1}$$ {proba_norm}

We shall denote by $\mathbb{P}(A)$ the probability of an **event** $A \subseteq \mathcal{X}$, so that $p_X(x) = \mathbb{P}(X = x)$. To lighten notations, when there is no ambiguity, we use $p(x)$ to denote $p_X(x)$.

If $f(X)$ is a real valued function of the random variable X, the **expectation value** of $f(X)$, which we shall also call the average of f, is denoted by:

$$\mathbb{E}\, f = \sum_{x\in\mathcal{X}} p_X(x) f(x) \ . \tag{1.2}$$

While our main focus will be on random variables taking values in finite spaces, we shall sometimes make use of **continuous random variables** taking values in $\mathbb{R}^d$ or in some smooth finite-dimensional manifold. The probability measure for an 'infinitesimal element' dx will be denoted by $dp_X(x)$. Each time p_X admits a density (with respect to the Lebesgue measure), we shall use the notation $p_X(x)$ for the value of this density at the point x. The total probability $\mathbb{P}(X \in \mathcal{A})$ that the variable X takes value in some (Borel) set $\mathcal{A} \subseteq \mathcal{X}$ is given by the integral:

[1] In probabilistic jargon (which we shall avoid hereafter), we take the probability space $(\mathcal{X}, \mathsf{P}(\mathcal{X}), p_X)$ where $\mathsf{P}(\mathcal{X})$ is the σ-field of the parts of $\mathcal{X}$ and $p_X = \sum_{x\in X} p_X(x)\, \delta_x$.

$$\mathbb{P}(X \in \mathcal{A}) = \int_{x\in\mathcal{A}} dp_X(x) = \int \mathbb{I}(x \in \mathcal{A})\, dp_X(x)\ , \tag{1.3}$$

where the second form uses the **indicator function** $\mathbb{I}(s)$ of a logical statement s,which is defined to be equal to 1 if the statement s is true, and equal to 0 if the statement is false.

The expectation value of a real valued function $f(x)$ is given by the integral on $\mathcal{X}$:

$$\mathbb{E}\, f(X) = \int f(x)\ dp_X(x)\ . \tag{1.4}$$

Sometimes we may write $\mathbb{E}_X f(X)$ for specifying the variable to be integrated over. We shall often use the shorthand **pdf** for the **probability density function** $p_X(x)$.

Example 1.1 A fair dice with M faces has $\mathcal{X} = \{1, 2, ..., M\}$ and $p(i) = 1/M$ for all $i \in \{1, ..., M\}$. The average of x is $\mathbb{E}\, X = (1 + ... + M)/M = (M + 1)/2$.

Example 1.2 Gaussian variable: a continuous variable $X \in \mathbb{R}$ has a Gaussian distribution of mean m and variance σ^2 if its probability density is

$$p(x) = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{[x-m]^2}{2\sigma^2}\right)\ . \tag{1.5}$$

One has $\mathbb{E}X = m$ and $\mathbb{E}(X - m)^2 = \sigma^2$.

The notations of this chapter mainly deal with discrete variables. Most of the expressions can be transposed to the case of continuous variables by replacing sums $\sum_x$ by integrals and interpreting $p(x)$ as a probability density.

Exercise 1.1 Jensen's inequality. Let X be a random variable taking value in a set $\mathcal{X} \subseteq \mathbb{R}$ and f a convex function (i.e. a function such that $\forall x, y$ and $\forall \alpha \in [0, 1]$: $f(\alpha x + (1 - \alpha y)) \leq \alpha f(x) + (1 - \alpha) f(y)$). Then

{eq:Jensen}

$$\mathbb{E} f(X) \geq f(\mathbb{E}X)\ . \tag{1.6}$$

Supposing for simplicity that $\mathcal{X}$ is a finite set with $|\mathcal{X}| = n$, prove this equality by recursion on n.

{se:entropy}

1.2 Entropy

The **entropy** H_X of a discrete random variable X with probability distribution $p(x)$ is defined as

{S_def}

$$H_X \equiv -\sum_{x\in\mathcal{X}} p(x) \log_2 p(x) = \mathbb{E} \log_2 \left[\frac{1}{p(X)}\right]\ , \tag{1.7}$$

where we define by continuity $0 \log_2 0 = 0$. We shall also use the notation $H(p)$ whenever we want to stress the dependence of the entropy upon the probability distribution of X.

In this Chapter we use the logarithm to the base 2, which is well adapted to digital communication, and the entropy is then expressed in **bits**. In other contexts one rather uses the natural logarithm (to base $e \approx 2.7182818$). It is sometimes said that, in this case, entropy is measured in **nats**. In fact, the two definitions differ by a global multiplicative constant, which amounts to a change of units. When there is no ambiguity we use H instead of H_X.

Intuitively, the entropy gives a measure of the uncertainty of the random variable. It is sometimes called the missing information: the larger the entropy, the less a priori information one has on the value of the random variable. This measure is roughly speaking the logarithm of the number of typical values that the variable can take, as the following examples show.

Example 1.3 A fair coin has two values with equal probability. Its entropy is 1 bit.

Example 1.4 Imagine throwing M fair coins: the number of all possible outcomes is 2^M. The entropy equals M bits.

Example 1.5 A fair dice with M faces has entropy $\log_2 M$.

Example 1.6 Bernouilli process. A random variable X can take values $0, 1$ with probabilities $p(0) = q$, $p(1) = 1 - q$. Its entropy is

$$H_X = -q \log_2 q - (1-q) \log_2 (1-q) \ , \tag{1.8}$$ {S_bern}

it is plotted as a function of q in fig.1.1. This entropy vanishes when $q = 0$ or $q = 1$ because the outcome is certain, it is maximal at $q = 1/2$ when the uncertainty on the outcome is maximal.

Since Bernoulli variables are ubiquitous, it is convenient to introduce the function $\mathcal{H}(q) \equiv -q \log q - (1-q) \log(1-q)$, for their entropy.

Exercise 1.2 An unfair dice with four faces and $p(1) = 1/2$, $p(2) = 1/4$, $p(3) = p(4) = 1/8$ has entropy $H = 7/4$, smaller than the one of the corresponding fair dice.

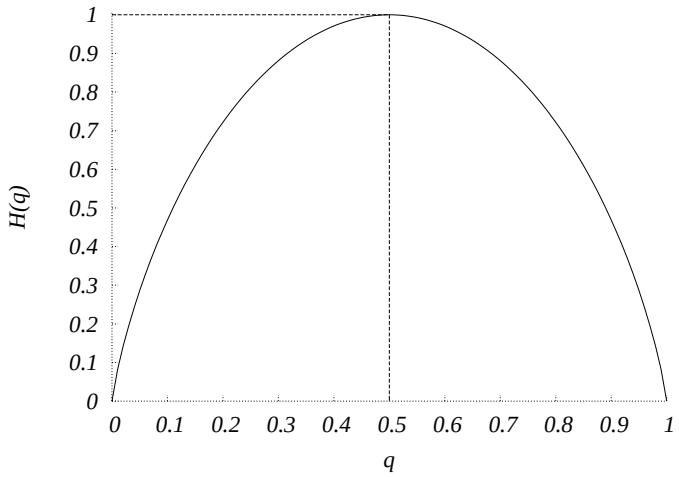

FIG. 1.1. The entropy $\mathcal{H}(q)$ of a binary variable with $p(X = 0) = q$, $p(X = 1) = 1 - q$, plotted versus q {fig_bernouilli}

Exercise 1.3 DNA is built from a sequence of bases which are of four types, A,T,G,C. In natural DNA of primates, the four bases have nearly the same frequency, and the entropy per base, if one makes the simplifying assumptions of independence of the various bases, is $H = -\log_2(1/4) = 2$. In some genus of bacteria, one can have big differences in concentrations: $p(G) = p(C) = 0.38$, $p(A) = p(T) = 0.12$, giving a smaller entropy $H \approx 1.79$.

Exercise 1.4 In some intuitive way, the entropy of a random variable is related to the 'risk' or 'surprise' which are associated to it. In this example we discuss a simple possibility for making these notions more precise.

Consider a gambler who bets on a sequence of bernouilli random variables $X_t \in \{0, 1\}$, $t \in \{0, 1, 2, \dots\}$ with mean $\mathbb{E}X_t = p$. Imagine he knows the distribution of the X_t's and, at time t he bets a fraction $w(1) = p$ of his money on 1 and a fraction $w(0) = (1-p)$ on 0. He looses whatever is put on the wrong number, while he doubles whatever has been put on the right one. Define the average doubling rate of his wealth at time t as

$$W_t = \frac{1}{t}\,\mathbb{E}\,\log_2\left\{\prod_{t'=1}^{t} 2w(X_{t'})\right\}\,. \tag{1.9}$$

It is easy to prove that the expected doubling rate $\mathbb{E}W_t$ is related to the entropy of X_t: $\mathbb{E}W_t = 1 - \mathcal{H}(p)$. In other words, it is easier to make money out of predictable events.

Another notion that is directly related to entropy is the **Kullback-Leibler**

(KL) divergence between two probability distributions $p(x)$ and $q(x)$ over the same finite space $\mathcal{X}$. This is defined as:

$$D(q||p) \equiv \sum_{x\in\mathcal{X}} q(x) \log \frac{q(x)}{p(x)} \tag{1.10}$$

where we adopt the conventions $0 \log 0 = 0$, $0 \log(0/0) = 0$. It is easy to show that: (i) $D(q||p)$ is convex in $q(x)$; (ii) $D(q||p) \geq 0$; (iii) $D(q||p) > 0$ unless $q(x) \equiv p(x)$. The last two properties derive from the concavity of the logarithm (i.e. the fact that the function $-\log x$ is convex) and Jensen's inequality (1.6): if $\mathbb{E}$ denotes expectation with respect to the distribution $q(x)$, then $-D(q||p) = \mathbb{E} \log[p(x)/q(x)] \leq \log \mathbb{E}[p(x)/q(x)] = 0$. The KL divergence $D(q||p)$ thus looks like a distance between the probability distributions q and p, although it is not symmetric.

The importance of the entropy, and its use as a measure of information, derives from the following properties:

1. $H_X \geq 0$.
2. $H_X = 0$ if and only if the random variable X is certain, which means that X takes one value with probability one.
3. Among all probability distributions on a set $\mathcal{X}$ with M elements, H is maximum when all events x are equiprobable, with $p(x) = 1/M$. The entropy is then $H_X = \log_2 M$.
   Notice in fact that, if $\mathcal{X}$ has M elements, then the KL divergence $D(p||\overline{p})$ between $p(x)$ and the uniform distribution $\overline{p}(x) = 1/M$ is $D(p||\overline{p}) = \log_2 M - \mathcal{H}(p)$. The thesis follows from the properties of the KL divergence mentioned above.
4. If X and Y are two **independent** random variables, meaning that $p_{X,Y}(x,y) = p_X(x)p_Y(y)$, the total entropy of the pair X, Y is equal to $H_X + H_Y$:

$$\begin{aligned} H_{X,Y} &= -\sum_{x,y} p(x,y) \log_2 p_{X,Y}(x,y) = \\ &= -\sum_{x,y} p_X(x)p_Y(y) \left(\log_2 p_X(x) + \log_2 p_Y(y)\right) = H_X + H_Y \end{aligned} \tag{1.11}$$

5. For any pair of random variables, one has in general $H_{X,Y} \leq H_X + H_Y$, and this result is immediately generalizable to n variables. (The proof can be obtained by using the positivity of the KL divergence $D(p_1||p_2)$, where $p_1 = p_{X,Y}$ and $p_2 = p_X p_Y$). ⋆
6. Additivity for composite events. Take a finite set of events $\mathcal{X}$, and decompose it into $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2$, where $\mathcal{X}_1 \cap \mathcal{X}_2 = \emptyset$. Call $q_1 = \sum_{x\in\mathcal{X}_1} p(x)$ the probability of $\mathcal{X}_1$, and q_2 the probability of $\mathcal{X}_2$. For each $x \in \mathcal{X}_1$, define as usual the conditional probability of x, given that $x \in \mathcal{X}_1$, by $r_1(x) = p(x)/q_1$ and define similarly $r_2(x)$ as the conditional probability

of x, given that $x \in \mathcal{X}_2$. Then the total entropy can be written as the sum of two contributions $H_X = -\sum_{x\in\mathcal{X}} p(x)\log_2 p(x) = H(q) + H(r)$, where:

$$H(q) = -q_1 \log_2 q_1 - q_2 \log_2 q_2 \tag{1.12}$$

$$H(r) = -q_1 \sum_{x\in\mathcal{X}_1} r_1(x)\log_2 r_1(x) - q_2 \sum_{x\in\mathcal{X}_1} r_2(x)\log_2 r_2(x) \tag{1.13}$$

⋆ The proof is obvious by just substituting the laws r_1 and r_2 by their expanded definitions. This property is interpreted as the fact that the average information associated to the choice of an event x is additive, being the sum of the relative information $H(q)$ associated to a choice of subset, and the information $H(r)$ associated to the choice of the event inside the subsets (weighted by the probability of the subsetss). It is the main property of the entropy, which justifies its use as a measure of information. In fact, this is a simple example of the so called chain rule for conditional entropy, which will be further illustrated in Sec. 1.4.

Conversely, these properties together with some hypotheses of continuity and monotonicity can be used to define axiomatically the entropy.

{sec:RandomVarSequences}

1.3 Sequences of random variables and entropy rate

In many situations of interest one deals with a random process which generates **sequences of random variables** $\{X_t\}_{t\in\mathbb{N}}$, each of them taking values in the same finite space $\mathcal{X}$. We denote by $P_N(x_1,\dots,x_N)$ the joint probability distribution of the first N variables. If $A \subset \{1,\dots,N\}$ is a subset of indices, we shall denote by $\overline{A}$ its complement $\overline{A} = \{1,\dots,N\} \setminus A$ and use the notations $\underline{x}_A = \{x_i, i \in A\}$ and $\underline{x}_{\overline{A}} = \{x_i, i \in \overline{A}\}$. The **marginal distribution** of the variables in A is obtained by summing P_N on the variables in $\overline{A}$:

$$P_A(\underline{x}_A) = \sum_{\underline{x}_{\overline{A}}} P_N(x_1,\dots,x_N)\ . \tag{1.14}$$

Example 1.7 The simplest case is when the X_t's are **independent**. This means that $P_N(x_1,\dots,x_N) = p_1(x_1)p_2(x_2)\dots p_N(x_N)$. If all the distributions p_i are identical, equal to p, the variables are **independent identically distributed**, which will be abbreviated as **iid**. The joint distribution is

$$P_N(x_1,\dots,x_N) = \prod_{t=1}^{N} p(x_i)\,. \tag{1.15}$$

Example 1.8 The sequence $\{X_t\}_{t\in\mathbb{N}}$ is said to be a **Markov chain** if

$$P_N(x_1,\dots,x_N) = p_1(x_1)\prod_{t=1}^{N-1} w(x_t\to x_{t+1})\,. \tag{1.16}$$

Here $\{p_1(x)\}_{x\in\mathcal{X}}$ is called the **initial state**, and $\{w(x\to y)\}_{x,y\in\mathcal{X}}$ are the **transition probabilities** of the chain. The transition probabilities must be non-negative and normalized:

$$\sum_{y\in\mathcal{X}} w(x\to y) = 1\,, \qquad \text{for any } y\in\mathcal{X}. \tag{1.17}$$

When we have a sequence of random variables generated by a certain process, it is intuitively clear that the entropy grows with the number N of variables. This intuition suggests to define the **entropy rate** of a sequence $\{X_t\}_{t\in\mathbb{N}}$ as

$$h_X = \lim_{N\to\infty} H_{\underline{X}_N}/N\,, \tag{1.18}$$

if the limit exists. The following examples should convince the reader that the above definition is meaningful.

Example 1.9 If the X_t's are i.i.d. random variables with distribution $\{p(x)\}_{x\in\mathcal{X}}$, the additivity of entropy implies

$$h_X = H(p) = -\sum_{x\in\mathcal{X}} p(x)\log p(x)\,. \tag{1.19}$$

Example 1.10 Let $\{X_t\}_{t\in\mathbb{N}}$ be a Markov chain with initial state $\{p_1(x)\}_{x\in\mathcal{X}}$ and transition probabilities $\{w(x\to y)\}_{x,y\in\mathcal{X}}$. Call $\{p_t(x)\}_{x\in\mathcal{X}}$ the marginal distribution of X_t and assume the following limit to exist independently of the initial condition:

$$p^*(x) = \lim_{t\to\infty} p_t(x)\,. \tag{1.20}$$

As we shall see in chapter 4, this turns indeed to be true under quite mild hypotheses on the transition probabilities $\{w(x\to y)\}_{x,y\in\mathcal{X}}$. Then it is easy to show that

$$h_X = -\sum_{x,y\in\mathcal{X}} p^*(x)\,w(x\to y)\log w(x\to y)\,. \tag{1.21}$$

If you imagine for instance that a text in English is generated by picking letters randomly in the alphabet $\mathcal{X}$, with empirically determined transition probabilities $w(x\to y)$, then Eq. (1.21) gives a first estimate of the entropy of English. But if you want to generate a text which looks like English, you need a more general process, for instance one which will generate a new letter x_{t+1} given the value of the k previous letters $x_t, x_{t-1}, ..., x_{t-k+1}$, through transition probabilities $w(x_t, x_{t-1}, ..., x_{t-k+1} \to x_{t+1})$. Computing the corresponding entropy rate is easy. For $k = 4$ one gets an entropy of 2.8 bits per letter, much smaller than the trivial upper bound $\log_2 27$ (there are 26 letters, plus the space symbols), but many words so generated are still not correct English words. Some better estimates of the entropy of English, through guessing experiments, give a number around 1.3.

{se:CorrelatedVariables}

1.4 Correlated variables and mutual entropy

Given two random variables X and Y, taking values in $\mathcal{X}$ and $\mathcal{Y}$, we denote their joint probability distribution as $p_{X,Y}(x,y)$, which is abbreviated as $p(x,y)$, and the conditional probability distribution for the variable y given x as $p_{Y|X}(y|x)$, abbreviated as $p(y|x)$. The reader should be familiar with Bayes' classical theorem:

$$p(y|x) = p(x,y)/p(x)\ . \tag{1.22}$$

When the random variables X and Y are independent, $p(y|x)$ is x-independent. When the variables are dependent, it is interesting to have a measure on their degree of dependence: how much information does one obtain on the value of y if one knows x? The notions of conditional entropy and mutual entropy will be useful in this respect.

Let us define the **conditional entropy** $H_{Y|X}$ as the entropy of the law $p(y|x)$, averaged over x:

{Scond_def}

$$H_{Y|X} \equiv -\sum_{x\in\mathcal{X}} p(x) \sum_{y\in\mathcal{Y}} p(y|x)\log_2 p(y|x)\,. \tag{1.23}$$

The total entropy $H_{X,Y} \equiv -\sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log_2 p(x,y)$ of the pair of variables x, y can be written as the entropy of x plus the conditional entropy of y given x:

$$H_{X,Y} = H_X + H_{Y|X}\,. \tag{1.24}$$

In the simple case where the two variables are independent, $H_{Y|X} = H_Y$, and $H_{X,Y} = H_X + H_Y$. One way to measure the correlation of the two variables is the **mutual entropy** $I_{X,Y}$ which is defined as:

$$I_{X,Y} \equiv \sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log_2 \frac{p(x,y)}{p(x)p(y)}\ . \tag{1.25}$$ {Smut_def}

It is related to the conditional entropies by:

$$I_{X,Y} = H_Y - H_{Y|X} = H_X - H_{X|Y}\ , \tag{1.26}$$

which shows that $I_{X,Y}$ measures the reduction in the uncertainty of x due to the knowledge of y, and is symmetric in x, y.

Proposition 1.11 *$I_{X,Y} \geq 0$. Moreover $I_{X,Y} = 0$ if and only if X and Y are independent variables.*

Proof: Write $-I_{X,Y} = \mathbb{E}_{x,y}\log_2 \frac{p(x)p(y)}{p(x,y)}$. Consider the random variable $u = (x,y)$ with probability distribution $p(x,y)$. As the logarithm is a concave function (i.e. -log is a convex function), one and applies Jensen's inequality (1.6). This gives the result $I_{X,Y} \geq 0$ □

Exercise 1.5 A large group of friends plays the following game (telephone without cables). The guy number zero chooses a number $X_0 \in \{0,1\}$ with equal probability and communicates it to the first one without letting the others hear, and so on. The first guy communicates the number to the second one, without letting anyone else hear. Call X_n the number communicated from the n-th to the $(n+1)$-th guy. Assume that, at each step a guy gets confused and communicates the wrong number with probability p. How much information does the n-th person have about the choice of the first one?

We can quantify this information through $I_{X_0,X_n} \equiv I_n$. A simple calculation shows that $I_n = 1 - \mathcal{H}(p_n)$ with p_n given by $1 - 2p_n = (1-2p)^n$. In particular, as $n \to \infty$

$$I_n = \frac{(1-2p)^{2n}}{2\log 2}\left[1 + O((1-2p)^{2n})\right]\ . \tag{1.27}$$

The 'knowledge' about the original choice decreases exponentially along the chain.

The mutual entropy gets degraded when data is transmitted or processed. This is quantified by:

Proposition 1.12 ***Data processing inequality***.

Consider a Markov chain $X \to Y \to Z$ (so that the joint probability of the three varaibles can be written as $p_1(x)w_2(x \to y)w_3(y \to z)$). Then: $I_{X,Z} \leq I_{X,Y}$. In particular, if we apply this result to the case where Z is a function of Y, $Z = f(Y)$, we find that applying f degrades the information: $I_{X,f(Y)} \leq I_{X,Y}$.

Proof: Let us introduce, in general, the mutual entropy of two varaibles conditioned to a third one: $I_{X,Y|Z} = H_{X|Z} - H_{X,(YZ)}$. The mutual information between a variable X and a pair of varaibles (YZ) can be decomposed in a sort of **chain rule**: $I_{X,(YZ)} = I_{X,Z} + I_{X,Y|Z} = I_{X,Y} + I_{X,Z|Y}$. If we have a Markov chain $X \to Y \to Z$, X and Z are independent when one conditions on the value of Y, therefore $I_{X,Z|Y} = 0$. The result follows from the fact that $I_{X,Y|Z} \geq 0$. □

1.5 Data compression

Imagine an information source which generates a sequence of symbols $\underline{X} = \{X_1, \ldots, X_N\}$ taking values in a finite alphabet $\mathcal{X}$. Let us assume a probabilistic model for the source: this means that the X_i's are taken to be random variables. We want to store the information contained in a given realization $\underline{x} = \{x_1 \ldots x_N\}$ of the source in the most compact way.

This is the basic problem of **source coding**. Apart from being an issue of utmost practical interest, it is a very instructive subject. It allows in fact to formalize in a concrete fashion the intuitions of 'information' and 'uncertainty' which are associated to the definition of entropy. Since entropy will play a crucial role throughout the book, we present here a little *detour* into source coding.

1.5.1 *Codewords*

We first need to formalize what is meant by "storing the information". We define[2] therefore a **source code** for the random variable $\underline{X}$ to be a mapping w which associates to any possible information sequence in $\mathcal{X}^N$ a string in a reference alphabet which we shall assume to be $\{0, 1\}$:

$$\begin{aligned} w : \mathcal{X}^N &\to \{0,1\}^* \\ \underline{x} &\mapsto w(\underline{x})\,. \end{aligned} \tag{1.28}$$

Here we used the convention of denoting by $\{0, 1\}^*$ the set of binary strings of arbitrary length. Any binary string which is in the image of w is called a **codeword**.

Often the sequence of symbols $X_1 \ldots X_N$ is a part of a longer stream. The compression of this stream is realized in three steps. First the stream is broken into blocks of length N. Then each block is encoded separately using w. Finally the codewords are glued to form a new (hopefully more compact) stream. If the original stream consisted in the blocks $\underline{x}^{(1)}, \underline{x}^{(2)}, \ldots, \underline{x}^{(r)}$, the output of the

[2]The expert will notice that here we are restricting our attention to "fixed-to-variable" codes.

encoding process will be the concatenation of $w(\underline{x}^{(1)}), \dots, w(\underline{x}^{(r)})$. In general there is more than one way of parsing this concatenation into codewords, which may cause troubles to any one willing to recover the compressed data. We shall therefore require the code w to be such that any concatenation of codewords can be parsed unambiguously. The mappings w satisfying this property are called **uniquely decodable codes**.

Unique decodability is surely satisfied if, for any pair $\underline{x}, \underline{x}' \in \mathcal{X}^N$, $w(\underline{x})$ is not a prefix of $w(\underline{x}')$. If this stronger condition is verified, the code is said to be **instantaneous** (see Fig. 1.2). Hereafter we shall focus on instantaneous codes, since they are both practical and (slightly) simpler to analyze.

Now that we precised how to store information, namely using a source code, it is useful to introduce some figure of merit for source codes. If $l_w(x)$ is the length of the string $w(x)$, the **average length** of the code is:

$$L(w) = \sum_{\underline{x} \in \mathcal{X}^N} p(\underline{x})\, l_w(\underline{x}) \ . \tag{1.29}$$ {avlength}

Example 1.13 Take $N = 1$ and consider a random variable X which takes values in $\mathcal{X} = \{1, 2, \dots, 8\}$ with probabilities $p(1) = 1/2$, $p(2) = 1/4$, $p(3) = 1/8$, $p(4) = 1/16$, $p(5) = 1/32$, $p(6) = 1/64$, $p(7) = 1/128$, $p(8) = 1/128$. Consider the two codes w_1 and w_2 defined by the table below

x	$p(x)$	$w_1(x)$	$w_2(x)$
1	1/2	000	0
2	1/4	001	10
3	1/8	010	110
4	1/16	011	1110
5	1/32	100	11110
6	1/64	101	111110
7	1/128	110	1111110
8	1/128	111	11111110

(1.30)

These two codes are instantaneous. For instance looking at the code w_2, the encoded string 10001101110010 can be parsed in only one way since each symbol 0 ends a codeword. It thus corresponds to the sequence $x_1 = 2, x_2 = 1, x_3 = 1, x_4 = 3, x_5 = 4, x_6 = 1, x_7 = 2$. The average length of code w_1 is $L(w_1) = 3$, the average length of code w_2 is $L(w_2) = 247/128$. Notice that w_2 achieves a shorter average length because it assigns the shortest codeword (namely 0) to the most probable symbol (i.e. 1).

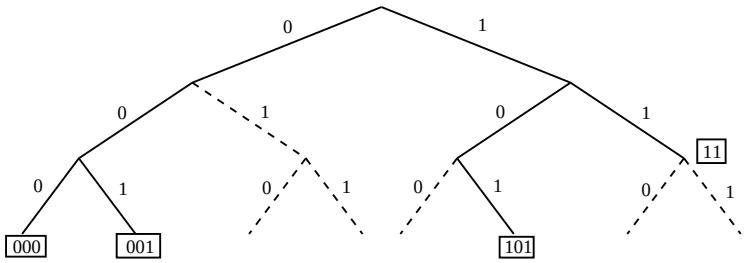

FIG. 1.2. An instantaneous source code: each codeword is assigned to a node in a binary tree in such a way that no one among them is the ancestor of another one. Here the four codewords are framed. {fig_kraft}

Example 1.14 A useful graphical representation of source code is obtained by drawing a binary tree and associating each codeword to the corresponding node in the tree. In Fig. 1.2 we represent in this way a source code with $|\mathcal{X}^N| = 4$. It is quite easy to recognize that the code is indeed instantaneous. The codewords, which are framed, are such that no codeword is the ancestor of any other codeword in the tree. Given a sequence of codewords, parsing is immediate. For instance the sequence `00111000101001` can be parsed only in `001, 11, 000, 101, 001`

1.5.2 *Optimal compression and entropy*

Suppose to have a 'complete probabilistic characterization' of the source you want to compress. What is the 'best code' w for this source? What is the shortest achievable average length?

This problem was solved (up to minor refinements) by Shannon in his celebrated 1948 paper, by connecting the best achievable average length to the entropy of the source. Following Shannon we assume to know the probability distribution of the source $p(\underline{x})$ (this is what 'complete probabilistic characterization' means). Moreover we interpret 'best' as 'having the shortest average length'.

{theorem:ShannonSource}

Theorem 1.15 *Let L_N^* the shortest average length achievable by an instantaneous code for $\underline{X} = \{X_1, \dots, X_N\}$, and $H_{\underline{X}}$ the entropy of the same variable. Then*

1. *For any $N \geq 1$:*

{Shcomp1}
$$H_{\underline{X}} \leq L_N^* \leq H_{\underline{X}} + 1\,. \tag{1.31}$$

2. *If the source has a finite entropy rate $h = \lim_{N\to\infty} H_{\underline{X}}/N$, then*

{Shcomp2}
$$\lim_{N\to\infty} \frac{1}{N} L_N^* = h\,. \tag{1.32}$$

Proof: The basic idea in the proof of Eq. (1.31) is that, if the codewords were too short, the code wouldn't be instantaneous. 'Kraft's inequality' makes

this simple remark more precise. For any instantaneous code w, the lengths $l_w(\underline{x})$ satisfy:

$$\sum_{\underline{x}\in\mathcal{X}^N} 2^{-l_w(\underline{x})} \leq 1\,. \tag{1.33}$$ {kraft}

This fact is easily proved by representing the set of codewords as a set of leaves on a binary tree (see fig.1.2). Let L_M be the length of the longest codeword. Consider the set of all the 2^{L_M} possible vertices in the binary tree which are at the generation L_M, let us call them the 'descendants'. If the information $\underline{x}$ is associated with a codeword at generation l (i.e. $l_w(\underline{x}) = l$), there can be no other codewords in the branch of the tree rooted on this codeword, because the code is instantaneous. We 'erase' the corresponding 2^{L_M-l} descendants which cannot be codewords. The subsets of erased descendants associated with each codeword are not overlapping. Therefore the total number of erased descendants, $\sum_{\underline{x}} 2^{L_M-l_w(\underline{x})}$, must be smaller or equal to the total number of descendants, 2^{L_M}. This establishes Kraft's inequality.

Conversely, for any set of lengths $\{l(\underline{x})\}_{\underline{x}\in\mathcal{X}^N}$ which satisfies the inequality (1.33) there exist at least a code, whose codewords have the lengths $\{l(\underline{x})\}_{\underline{x}\in\mathcal{X}^N}$. A possible construction is obtained as follows. Consider the smallest length $l(\underline{x})$ and take the first allowed binary sequence of length $l(\underline{x})$ to be the codeword for $\underline{x}$. Repeat this operation with the next shortest length, and so on until you have exhausted all the codewords. It is easy to show that this procedure is successful if Eq. (1.33) is satisfied.

The problem is therefore reduced to finding the set of codeword lengths $l(\underline{x}) = l^*(\underline{x})$ which minimize the average length $L = \sum_{\underline{x}} p(\underline{x})l(\underline{x})$ subject to Kraft's inequality (1.33). Supposing first that $l(\underline{x})$ are real numbers, this is easily done with Lagrange multipliers, and leads to $l(\underline{x}) = -\log_2 p(\underline{x})$. This set of optimal lengths, which in general cannot be realized because some of the $l(\underline{x})$ are not integers, gives an average length equal to the entropy $H_{\underline{X}}$. This gives the lower bound in (1.31). In order to build a real code with integer lengths, we use

$$l^*(\underline{x}) = \lceil -\log_2 p(\underline{x}) \rceil\,. \tag{1.34}$$

Such a code satisfies Kraft's inequality, and its average length is less or equal than $H_{\underline{X}} + 1$, proving the upper bound in (1.31).

The second part of the theorem is a straightforward consequence of the first one. □

The code we have constructed in the proof is often called a **Shannon code**. For long strings ($N \gg 1$), it gets close to optimal. However it has no reason to be optimal in general. For instance if only one $p(x)$ is very small, it will code it on a very long codeword, while shorter codewords are available. It is interesting to know that, for a given source $\{X_1, \ldots, X_N\}$, there exists an explicit construction of the optimal code, called Huffman's code.

At first sight, it may appear that Theorem 1.15, together with the construction of Shannon codes, completely solves the source coding problem. But this is far from true, as the following arguments show.

From a computational point of view, the encoding procedure described above is unpractical. One can build the code once for all, and store it somewhere, but this requires $O(|\mathcal{X}|^N)$ memory. On the other hand, one could reconstruct the code each time a string requires to be encoded, but this takes $O(|\mathcal{X}|^N)$ time. One can use the same code and be a bit smarter in the encoding procedure, but this does not improve things dramatically.

From a practical point of view, the construction of a Shannon code requires an accurate knowledge of the probabilistic law of the source. Suppose now you want to compress the complete works of Shakespeare. It is exceedingly difficult to construct a good model for the source 'Shakespeare'. Even worse: when you will finally have such a model, it will be of little use to compress Dante or Racine.

Happily, source coding has made tremendous progresses in both directions in the last half century.

{sec:DataTransmission}

1.6 Data transmission

In the previous pages we considered the problem of encoding some information in a string of symbols (we used bits, but any finite alphabet is equally good). Suppose now we want to communicate this string. When the string is transmitted, it may be corrupted by some noise, which depends on the physical device used in the transmission. One can reduce this problem by adding redundancy to the string. The redundancy is to be used to correct (some) transmission errors, in the same way as redundancy in the English language can be used to correct some of the typos in this book. This is the field of channel coding. A central result in information theory, again due to Shannon's pioneering work in 1948, relates the level of redundancy to the maximal level of noise that can be tolerated for error-free transmission. The entropy again plays a key role in this result. This is not surprising in view of the symmetry between the two problems. In data compression, one wants to reduce the redundancy of the data, and the entropy gives a measure of the ultimate possible reduction. In data transmission, one wants to add some well tailored redundancy to the data.

1.6.1 *Communication channels*

The typical flowchart of a communication system is shown in Fig. 1.3. It applies to situations as diverse as communication between the earth and a satellite, the cellular phones, or storage within the hard disk of your computer. Alice wants to send a message m to Bob. Let us assume that m is a M bit sequence. This message is first encoded into a longer one, a N bit message denoted by $\underline{x}$ with $N > M$, where the added bits will provide the redundancy used to correct for transmission errors. The encoder is a map from $\{0,1\}^M$ to $\{0,1\}^N$. The encoded message is sent through the communication channel. The output of the channel is a message $\underline{y}$. In a noiseless channel, one would simply have $\underline{y} = \underline{x}$. In a realistic channel, $\underline{y}$ is in general a string of symbols different from $\underline{x}$. Notice that $\underline{y}$ is not even necessarily a string of bits. The **channel** will be described by the transition probability $Q(\underline{y}|\underline{x})$. This is the probability that the received signal is

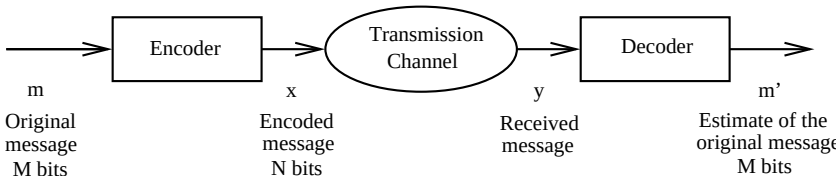

FIG. 1.3. Typical flowchart of a communication device. {fig_channel}

$\underline{y}$, conditional to the transmitted signal being $\underline{x}$. Different physical channels will be described by different $Q(\underline{y}|\underline{x})$ functions. The decoder takes the message $\underline{y}$ and deduces from it an estimate m' of the sent message.

Exercise 1.6 Consider the following example of a channel with **insertions**. When a bit `x` is fed into the channel, either `x` or `x0` are received with equal probability 1/2. Suppose that you send the string `111110`. The string `1111100` will be received with probability $2 \cdot 1/64$ (the same output can be produced by an error either on the 5^{th} or on the 6^{th} digit). Notice that the output of this channel is a bit string which is always longer or equal to the transmitted one.

A simple code for this channel is easily constructed: use the string `100` for each `0` in the original message and `1100` for each `1`. Then for instance you have the encoding

$$\texttt{01101} \mapsto \texttt{100110011001001100}\,. \tag{1.35}$$

The reader is invited to define a decoding algorithm and verify its effectiveness.

Hereafter we shall consider **memoryless** channels. In this case, for any input $\underline{x} = (x_1, ..., x_N)$, the output message is a string of N letters, $\underline{y} = (y_1, ..., y_N)$, from an alphabet $\mathcal{Y} \ni y_i$ (not necessarily binary). In memoryless channels, the noise acts independently on each bit of the input. This means that the conditional probability $Q(\underline{y}|\underline{x})$ factorizes:

$$Q(\underline{y}|\underline{x}) = \prod_{i=1}^{N} Q(y_i|x_i)\ , \tag{1.36}$$

and the transition probability $Q(y_i|x_i)$ is i independent.

Example 1.16 Binary symmetric channel (BSC). The input x_i and the output y_i are both in $\{0, 1\}$. The channel is characterized by one number, the probability p that an input bit is transmitted as the opposite bit. It is customary to represent it by the diagram of Fig. 1.4.

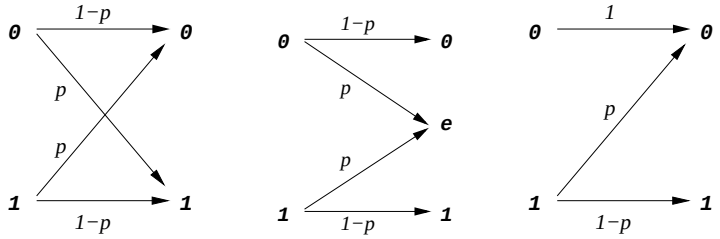

FIG. 1.4. Three communication channels. *Left*: the binary symmetric channel. An error in the transmission, in which the output bit is the opposite of the input one, occurs with probability p. *Middle*: the binary erasure channel. An error in the transmission, signaled by the output `e`, occurs with probability p. *Right*: the Z channel. An error occurs with probability p whenever a `1` is transmitted. {fig_bsc}

Example 1.17 Binary erasure channel (BEC). In this case some of the input bits are erased instead of being corrupted: x_i is still in $\{0, 1\}$, but y_i now belongs to $\{0, 1, e\}$, where `e` means erased. In the symmetric case, this channel is described by a single number, the probability p that a bit is erased, see Fig. 1.4.

Example 1.18 Z channel. In this case the output alphabet is again $\{0, 1\}$. Moreover, a `0` is always transmitted correctly, while a `1` becomes a `0` with probability p. The name of this channel come from its graphical representation, see Fig. 1.4.

A very important characteristics of a channel is the **channel capacity** C. It is defined in terms of the mutual entropy I_{XY} of the variables X (the bit which was sent) and Y (the signal which was received), through:

{capadef}
$$C = \max_{p(x)} I_{XY} = \max_{p(x)} \sum_{x\in\mathcal{X}, y\in\mathcal{Y}} p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)} \tag{1.37}$$

We recall that I measures the reduction on the uncertainty of x due to the knowledge of y. The capacity C gives a measure of how faithful a channel can be: If the output of the channel is pure noise, x and y are uncorrelated and $C = 0$. At the other extreme if $y = f(x)$ is known for sure, given x, then $C = \max_{\{p(x)\}} H(p) = 1$ bit. The interest of the capacity will become clear in section 1.6.3 with Shannon's coding theorem which shows that C characterizes the amount of information which can be transmitted faithfully in a channel.

Example 1.19 Consider a binary symmetric channel with flip probability p. Let us call q the probability that the source sends $x = \mathtt{0}$, and $1 - q$ the probability of $x = \mathtt{1}$. It is easy to show that the mutual information in Eq. (1.37) is maximized when zeros and ones are transmitted with equal probability (i.e. when $q = 1/2$).

Using the expression (1.37), we get, $C = 1 - \mathcal{H}(p)$ bits, where $\mathcal{H}(p)$ is the entropy of Bernouilli's process with parameter p (plotted in Fig. 1.1).

Example 1.20 Consider now the binary erasure channel with error probability p. The same argument as above applies. It is therefore easy to get $C = 1-p$.

Exercise 1.7 Compute the capacity of the Z channel.

1.6.2 *Error correcting codes*

{sec:ECC}

The only ingredient which we still need to specify in order to have a complete definition of the channel coding problem, is the behavior of the information source. We shall assume it to produce a sequence of uncorrelated unbiased bits. This may seem at first a very crude model for any real information source. Surprisingly, Shannon's source-channel separation theorem assures that there is indeed no loss of generality in treating this case.

The sequence of bits produced by the source is divided in blocks $m_1, m_2, m_3, \ldots$ of length M. The **encoding** is a mapping from $\{0,1\}^M \ni m$ to $\{0,1\}^N$, with $N \geq M$. Each possible M-bit message m is mapped to a **codeword** $\underline{x}(m)$ which is a point in the N-dimensional unit hypercube. The codeword length N is also called the **blocklength**. There are 2^M codewords, and the set of all possible codewords is called the **codebook**. When the message is transmitted, the codeword $\underline{x}$ is corrupted to $\underline{y} \in \mathcal{Y}^N$ with probability $Q(\underline{y}|\underline{x}) = \prod_{i=1}^N Q(y_i|x_i)$. The output alphabet $\mathcal{Y}$ depends on the channel. The **decoding** is a mapping from $\mathcal{Y}^N$ to $\{0,1\}^M$ which takes the received message $\underline{y} \in \mathcal{Y}^N$ and maps it to one of the possible original messages $m' = d(\underline{y}) \in \{0,1\}^M$.

An **error correcting code** is defined by the set of two functions, the encoding $\underline{x}(m)$ and the decoding $d(\underline{y})$. The ratio

$$R = \frac{M}{N} \tag{1.38}$$

of the original number of bits to the transmitted number of bits is called the **rate** of the code. The rate is a measure of the redundancy of the code. The smaller the rate, the more redundancy is added to the code, and the more errors one should be able to correct.

The **block error probability** of a code on the input message m, denoted by $\mathrm{P_B}(m)$, is given by the probability that the decoded messages differs from the one which was sent:

$$\mathrm{P_B}(m) = \sum_{\underline{y}} Q(\underline{y}|\underline{x}(m)) \;\; \mathbb{I}(d(\underline{y}) \neq m) \; . \tag{1.39}$$

Knowing thee probability for each possible transmitted message is an exceedingly detailed characterization of the code performances. One can therefore introduce a **maximal block error probability** as

$$\mathrm{P_B^{max}} \equiv \max_{m \in \{0,1\}^M} \mathrm{P_B}(m) \, . \tag{1.40}$$

This corresponds to characterizing the code by its 'worst case' performances. A more optimistic point of view consists in averaging over the input messages. Since we assumed all of them to be equiprobable, we introduce the **average block error probability** as

$$\mathrm{P_B^{av}} \equiv \frac{1}{2^M} \sum_{m \in \{0,1\}^M} \mathrm{P_B}(m) \, . \tag{1.41}$$

Since this is a very common figure of merit for error correcting codes, we shall call it block error probability and use the symbol $\mathrm{P_B}$ without further specification hereafter.

Example 1.21 Repetition code. Consider a BSC which transmits a wrong bit with probability p. A simple code consists in repeating k times each bit, with k odd. Formally we have $M = 1$, $N = k$ and

$$\underline{x}(\mathtt{0}) = \underbrace{\mathtt{000\ldots00}}_{k}, \tag{1.42}$$

$$\underline{x}(\mathtt{1}) = \underbrace{\mathtt{111\ldots11}}_{k} \tag{1.43}$$

For instance with $k = 3$, the original stream `0110001` is encoded as `00011111100000 0000111`. A possible decoder consists in parsing the received sequence in groups of k bits, and finding the message m' from a majority rule among the k bits. In our example with $k = 3$, if the received group of three bits is `111` or `110` or any permutation, the corresponding bit is assigned to `1`, otherwise it is assigned to `0`. For instance if the channel output is `000101111011000010111`, the decoding gives `0111001`.

This $k = 3$ repetition code has rate $R = M/N = 1/3$. It is a simple exercise to see that the block error probability is $\mathrm{P_B} = p^3 + 3p^2(1-p)$ independently of the information bit.

Clearly the $k = 3$ repetition code is able to correct mistakes induced from the transmission only when there is at most one mistake per group of three bits. Therefore the block error probability stays finite at any nonzero value of the noise. In order to improve the performances of these codes, k must increase. The error probability for a general k is

$$\mathrm{P_B} = \sum_{r=\lceil k/2 \rceil}^{k} \binom{k}{r} (1-p)^{k-r} p^r \ . \tag{1.44}$$

Notice that for any finite k, $p > 0$ it stays finite. In order to have $\mathrm{P_B} \to 0$ we must consider $k \to \infty$. Since the rate is $R = 1/k$, the price to pay for a vanishing block error probability is a vanishing communication rate!

Happily enough much better codes exist as we will see below.

1.6.3 *The channel coding theorem*

{sec:channeltheorem}

Consider a communication device in which the channel capacity (1.37) is C. In his seminal 1948 paper, Shannon proved the following theorem.

{theorem:Shannon_channel}

Theorem 1.22 *For every rate $R < C$, there exists a sequence of codes $\{\mathcal{C}_N\}$, of blocklength N, rate R_N, and block error probability $\mathrm{P_{B,}}_N$, such that $R_N \to R$ and $\mathrm{P_{B,}}_N \to 0$ as $N \to \infty$. Conversely, if for a sequence of codes $\{\mathcal{C}_N\}$, one has $R_N \to R$ and $\mathrm{P_{B,}}_N \to 0$ as $N \to \infty$, then $R < C$.*

In practice, for long messages (i.e. large N), reliable communication is possible if and only if the communication rate stays below capacity. We shall not give the

proof here but differ it to Chapters 6 and **???**. Here we keep to some qualitative comments and provide the intuitive idea underlying this result.

First of all, the result is rather surprising when one meets it for the first time. As we saw on the example of repetition codes above, simple minded codes typically have a finite error probability, for any non-vanishing noise strength. Shannon's theorem establishes that it is possible to achieve zero error probability, while keeping the communication rate finite.

One can get an intuitive understanding of the role of the capacity through a qualitative reasoning, which uses the fact that a random variable with entropy H 'typically' takes 2^H values. For a given codeword $\underline{x}(m) \in \{0,1\}^N$, the channel output $\underline{y}$ is a random variable with an entropy $H_{\underline{y}|\underline{x}} = NH_{y|x}$. There exist of order $2^{NH_{y|x}}$ such outputs. For a perfect decoding, one needs a decoding function $d(\underline{y})$ that maps each of them to the original message m. Globally, the typical number of possible outputs is 2^{NH_y}, therefore one can send at most $2^{N(H_y - H_{y|x})}$ codewords. In order to have zero maximal error probability, one needs to be able to send all the $2^M = 2^{NR}$ codewords. This is possible only if $R < H_y - H_{y|x} < C$.

Notes

There are many textbooks introducing to probability and to information theory. A standard probability textbook is the one of Feller (Feller, 1968). The original Shannon paper (Shannon, 1948) is universally recognized as the foundation of information theory. A very nice modern introduction to the subject is the book by Cover and Thomas (Cover and Thomas, 1991). The reader may find there a description of Huffman codes which did not treat in the present Chapter, as well as more advanced topics in source coding.

We did not show that the six properties listed in Sec. 1.2 provide in fact an alternative (axiomatic) definition of entropy. The interested reader is referred to (Csiszár and Körner, 1981). An advanced information theory book with much space devoted to coding theory is (Gallager, 1968). The recent (and very rich) book by MacKay (MacKay, 2002) discusses the relations with statistical inference and machine learning.

The information-theoretic definition of entropy has been used in many contexts. It can be taken as a founding concept in statistical mechanics. Such an approach is discussed in (Balian, 1992).