Information theory

How can we quantify information? Let us take a specific example: Alice (A) is in a state of ignorance about a certain variable X which is known to Bob (B). She anticipates that the answer $X \in \mathcal{X}$ can be one of $n = |\mathcal{X}|$ possible ones. One way to quantify the information content of X is to count the number of binary questions (yes/no) that A needs to pose to B in order to know the answer X. Indeed, A's uncertainty will be dispelled after she hears the answers because she will know what X is. Therefore, the number N_Q of binary questions needed to dispel A's ignorance is an operative definition of the information content of X, and it is measured in *bits*[213].

This chapter heavily draws from Cover Chapter 2, and Chapter 4 of

William Bialek. *Biophysics: searching for principles*. Princeton University Press, 2012

[213] A bit is a variable that takes two values, 0 (for no) or 1 (for yes).

Take for example the case $\mathcal{X} = \{a, b, c, d\}$. Then A may ask a first question

Q_1: is $X \in \{a, b\}$ or not?

and depending on the answer, A may ask

Q_2: if $X \in \{a, b\}$ is $X = a$ or not? Else, if $X \notin \{a, b\}$ is $X = c$ or not?

The answers to these two questions reveal the correct outcome X. Hence the information is $N_Q = 2$ bits. Yet there are many other ways in which A could ask questions, and hence N_Q could vary accordingly.

For example A can modify her questions as follows:

Q'_1: is $X = a$ or not?

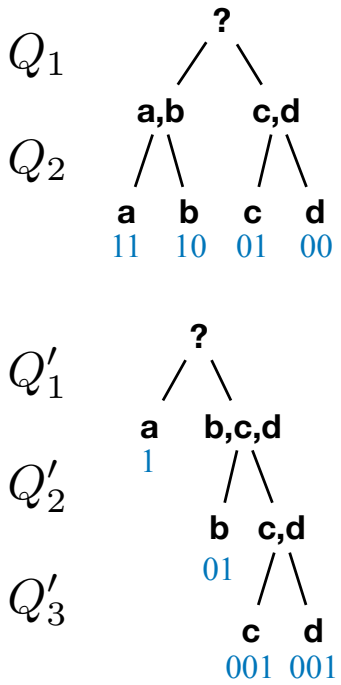
only if $X \neq a$ A will need to pose a further question. Then she may ask:

Q'_2: is $X = b$ or not?

Only if the result is no, she will need to ask

Q'_3: is $X = c$ or not?

in which case the number of binary questions can be $N_Q(a) = 1$, $N_Q(b) = 2$ or $N_Q(c) = N_Q(d) = 3$, depending on the value of X. Indeed, $N_Q(X)$ is a random variable, because it is a function of X.



Formally, the state of uncertainty of A is encoded in the probability distribution $P\{X = x\} = p_x$, of X. We're seeking for a measure

of information content of X that can quantify the uncertainty of A *before* the questions are posed and the answers are heard. Therefore, it makes sense to define a measure of information content as the expected number $\mathbb{E}\left[N_Q\right]$ of binary questions that are needed to elicit the value of X.

The expected value

$$\mathbb{E}\left[N_Q\right] = \sum_{x\in\mathcal{X}} p_x N_Q(x)$$

depends on the distribution p_x, that we assume is known to A, and on the way in which the answers are posed. For example, if A didn't know p_x, there is nothing that would distinguish the different outcomes, e.g. $X = a$ from $X = b$, so there is nothing that suggests that p_a should be smaller or larger than p_b. Hence, she would have to assume that $p_x = 1/4$ for all x. This distribution indeed encodes a state of maximal ignorance, as we shall see. Then asking questions (Q_1, Q_2) yields $\mathbb{E}\left[N_Q\right] = 2$ whereas formulating questions (Q_1', Q_2', Q_3') leads to a larger value of $\mathbb{E}\left[N_{Q'}\right] = 9/4$. If instead $p_a = 1/2$, $p_b = 1/4$ and $p_c = p_d = 1/8$, then again $\mathbb{E}\left[N_Q\right] = 2$, but

$$\mathbb{E}\left[N_{Q'}\right] = p_a \cdot 1 + p_b \cdot 2 + p_c \cdot 3 + p_d \cdot 3 = \frac{7}{4}. \tag{270}$$

The optimal way of answering questions is different in the two cases. The minimal expected number of binary questions that A needs to pose to elicit X is a measure of her irreducible ignorance about X. Hence, we provisionally define

> The information content $H[X]$ of a random variable X is the *minimal* expected number of binary questions needed to elicit the value of X,
>
> $$H[X] = \min_{Q} \mathbb{E}\left[N_Q\right] \tag{271}$$
>
> where the expected value is taken with respect to the distribution $P\{X = x\} = p_x$ that defines the state of knowledge on X, and the minimum is taken over all possible ways of posing yes/no questions.

Note that the information content

$$H: \quad X \to \mathbf{R}$$

is a *functional* that associates a real number $H[X]$ to a function

$$X: \ \Omega \to \mathbf{R}.$$

This is why we use square brackets in $H[\cdot]$.

The way in which Alice poses question associates to each values of X a strings of binary variables that we can take to be 1 for yes and 0 for no. Such a transformation between values of X and strings of bits

is called a *code*. Imagine that Alice asks Bob the same question many times (e.g. what's the weather today?) and that they communicate through a binary channel, i.e. a device that allow Bob and Alice to send either a 0 or a 1 to the other end at any time. Alice and Bob might be interested in finding the code which makes them exchange the shortest possible strings of bits. This problem is the same as the problem of finding the best way to ask questions.

Indeed, each protocol Q for asking questions corresponds to a scheme to encode the possible answers X. For example, the protocol Q above would correspond to the code

$$a \to 00;\ b \to 01;\ c \to 10;\ d \to 11\,.$$

The bit strings associated to a given value of X is called its *codeword*[214]. This code will require 2 bits for each answer transmitted from B to A. Protocol Q' corresponds to a different association of values of X to codewords, i.e.

$$a \to 0;\ b \to 10;\ c \to 110;\ d \to 111$$

Notice that each codeword has length $\ell_Q(X) = N_Q(X)$ which is equal to the number of binary questions needed to elicit X under protocol Q.

Therefore the problem of finding the code that is *expected* to use the least number of bits (i.e. that minimises $\mathbb{E}\left[\ell_Q\right]$) is exactly the same as the problem of finding the best way to pose questions. The fact that these two apparently different problems – A posing questions to B optimally and B transmitting answers to A efficiently – are the same, is interesting.

Note also that the optimal way Q^* of posing questions, and hence $H[X]$, depends only on the probabilities p_x, and not on what X is[215]. In particular, if an answer x is more likely than x', then it is natural that[216] $\ell_{Q^*}(x) \leq \ell_{Q^*}(x')$. For example, the knowledge of p_x in the example above, carries some information on the answer, which can be quantified in the difference between $\mathbb{E}\left[N_Q\right]$ in the two cases, and is $1/4$ of a bit in that case.

THE MINIMAL number of binary questions needed to elicit X, or equivalently the expected length of the optimal code for X, is given by the *Shannon entropy*

$$H[X] = \mathbb{E}\left[\log_2 1/p_X\right] = -\sum_{x\in\mathcal{X}} p_x \log_2 p_x \tag{272}$$

of the random variable X, that we shall simply call *entropy*, henceforth. The entropy depends on the distribution p_x, and we will equivalently denote it as $\mathcal{H}[p]$, when referring to it as a functional of the probability distribution p_x.

[214] In coding theory jargon, X are called *words*.

[215] X could be football teams in the Premier League or species of bird on some island. As long as the probabilities p_x are the same, the information content is the same.

[216] Think of the first binary question you would ask to know which team won the last Premier League championship.

It is easy to check that this is the correct answer in the examples above, where codewords have length exactly equal to $\log_2 1/p_x$, but one can argue that Eq. (272) works for all *discrete* random variables X, provided that we consider *messages* $\mathbf{X}_n = (X_1, \dots, X_n)$ where each of the n *characters* $X_i \in \chi$, are drawn i.i.d. from the distribution p_x. Then, in the limit $n \to \infty$, almost surely, we need at most $H[X]$ bits per character. This result, that goes under the name of *Shannon theorem*, is a direct consequence of the Asymptotic Equipartition Property. The idea of the proof is simple. Remember that the Asymptotic Equipartition Property ensures us that, for any $\epsilon > 0$, a message $\mathbf{X}_n$ belongs to the ϵ-typical set

$$A_n^{(\epsilon)} = \left\{ \mathbf{X}_n : \left| \frac{1}{n} \log P(\mathbf{X}_n) + H[X] \right| < \epsilon \right\}$$

almost surely, as $n \to \infty$. Imagine that Alice and Bob assign to all messages $\mathbf{X}_n \in A_n^{(\epsilon)}$ a different integer $Q(\mathbf{X}_n)$ from one to $|A_n^{(\epsilon)}|$, and to messages $\mathbf{X}_n \notin A_n^{(\epsilon)}$ integers $Q(\mathbf{X}_n)$ larger than $|A_n^{(\epsilon)}|$. Then each message will require a codeword of length $\ell_Q(\mathbf{X}) = \log_2 Q(\mathbf{X}_n)$, which is given by the binary representation of $Q(\mathbf{X}_n)$. Then, almost surely, Alice and Bob will need less than

$$\frac{1}{n} \max_{\mathbf{X}_n \in A_n^{(\epsilon)}} \log_2 Q(\mathbf{X}_n) = \frac{1}{n} \log_2 |A_n^{(\epsilon)}|$$

bits per character, as $n \to \infty$. In this limit, the Asymptotic Equipartition Property also implies that, almost surely,

$$\lim_{n \to \infty} \frac{1}{n} \log_2 |A_n^{(\epsilon)}| = H[X].$$

because $|A_n^{(\epsilon)}| \backsim e^{nH[X]}$. Therefore, at most $H[X]$ bits per character X need to be used to transmit the message, almost surely.

There are other ways to derive this result. For example, the same result can be obtained observing that the optimal number of bits needed to code X, should be a function $f(p_X)$ of p_X. Then the expected number of bits needed has to be of the form

$$H[X] = \mathbb{E}\left[f(p_X)\right] = \sum_{x \in \chi} p_x f(p_x)\,.$$

If $X = (Y, Z)$ where $Y \in \chi_Y$ and $Z \in \chi_Z$ are independent random variables, then $H[X] = H[Y] + H[Z]$, because knowing Y does not give any clue on what Z could be. Hence

$$\sum_{Y \in \chi_Y, Z \in \chi_Z} p_y p_z f(p_y p_z) = \sum_{Y \in \chi_Y, Z \in \chi_Z} p_y p_z \left[f(p_y) + f(p_z)\right]$$

for any p_y and p_z. Therefore $f(p_y p_z) = f(p_y) + f(p_z)$, which means that $f(p) = a \log p$. If in addition we want to measure information in

Exercise: The Rényi entropy is defined as

$$H_a[X] = \frac{1}{1-a} \log \sum_{x \in \chi} p_x^a$$

with $a > 0$. Show that $H_a[X]$ is a generalisation of the Shannon entropy, which is recovered in the limit $a \to 1$. Show that if X and Y are independent

$$H_a[X, Y] = H_a[X] + H_a[Y].$$

Show that, if the conditional Rényi entropy is defined as

$$H_a[X|Y] = \frac{1}{1-a} \mathbb{E}\left[\log \sum_{x \in \chi} p^a(x|Y) \right]$$

then the chain rule

$$H_a[X, Y] = H_a[Y] + H_a[X|Y]$$

holds only for $a \to 1$.

Exercise: Tsallis entropy is defined as

$$H_q[X] = \frac{1}{1-q} \left(1 - \mathbb{E}\left[p_X^{q-1}\right]\right).$$

Show that *i)* $H_q[X]$ reduces to the Shannon entropy for $q \to 1$, and that *ii)* H_q is not additive for $q \neq 1$, i.e. if X and Y are independent random variables, then

$$H_q[X, Y] = H_q[X] + H_q[Y] + (1-q) H_q[X] H_q[Y]\,.$$

bits, then $f(1/2) = 1$, i.e. $f(p) = -\log_2 p$. The entropy quantifies how much B's reply can be surprising for A. Indeed if both A and B knows that $p_x = 1$ if $x = a$ and $p_x = 0$ for all $x \neq a$, then B's reply cannot be surprising. Actually A doesn't even need to ask because both of them know that $X = a$. So no bit needs to be exchanged and, accordingly $H[X] = 0$. As we said, $H[X]$ quantifies the uncertainty of Alice about Bob's answer *before* she hears the answer. After she hears the answer, she knows that one answer occurs with probability one and the others with probability zero, i.e. $H = 0$. Then H measures how much Alice has decreased her degree of uncertainty.

Conversely, the entropy is maximal when X is maximally uncertain: $p_x = 1/|\mathcal{X}|$. Accordingly

$$0 \leq H[X] \leq \log |\chi|.$$

The entropy can be generalised to any number of random variables $X_1, \ldots, X_n$ in a straightforward fashion, i.e.

$$H[X_1, \ldots, X_n] = -\mathbb{E}\left[\log_2 P\{X_1, \ldots, X_n\}\right].$$

Likewise, we can define the conditional entropy

$$H[X|Y] = -\mathbb{E}\left[\log_2 P\{X|Y\}\right] = -\sum_{y\in\mathcal{Y}} p(y) \sum_{x\in\mathcal{X}} p(x|y) \log_2 p(x|y)$$

as the entropy of the conditional distribution $p(x|y)$, averaged over y. The law of conditional probability imply that[217]

$$H(X|Y) = H(X,Y) - H(Y). \tag{273}$$

[217] **Exercise:** check this.

In words, the conditional entropy is the reduction of the uncertainty about X and Y when Y is known, and it quantifies the the residual uncertainty on X (when Y is known). In particular, for a sequence of random variables $X_1, \ldots, X_n$, we have that

$$H[X_1, \ldots, X_n] = \sum_{m=2}^{n} H[X_m|X_{m-1}, \ldots, X_1] + H[X_1].$$

If the sequence is a Markov chain, then $H[X_m|X_{m-1}, \ldots, X_1] = H[X_m|X_{m-1}]$, because X_m given X_{m-1} is independent of X_k, for all $k < m-1$. If the transition probability $p_{i,j} = P\{X_n = j|X_{n-1} = i\}$ does not depend on n, and if the Markov chain is irreducible, then

$$H[X_2|X_1] = -\mathbb{E}\left[\log p_{X_1,X_2}\right]$$

is called the *entropy rate*, because $H[X_1, \ldots, X_n]/n \to H[X_2|X_1]$ as $n \to \infty$[218].

[218] The expected value on X_1 is taken on the invariant measure.

Entropy for continuous variables

The generalisation of the concept of entropy to continuous variables is problematic. Indeed, imagine that Alice asks to Bob about the price X of his car (and X is a real number). Even if she knows the *a priori* pdf $p(x)$ of X, she needs to ask an infinite number of binary questions in order to know X exactly. This does not match with the straightforward generalisation of Eq. (272)

See Chapter 8 of COVER.

$$h[X] = E[\log_2 1/p(X)] = -\int dx p(x) \log_2 p(x) \tag{274}$$

which is finite, barring pathological cases[219]. On the other hand, Eq. (274) seems problematic, since you may get negative numbers[220]! So what is the meaning of $h[X]$?

[219] **Exercise:** Compute $h[X]$ in Eq. (274) for $p(x) = 1/[x(\log x)^2]$ for $x \geq e$ and $p(x) = 0$ for $x < e$.

[220] **Exercise:** Check that $h[X] = -3$ for a uniform random variable $X \in [0, 1/8]$.

Coming back to Alice and Bob, even if the car retailer were really charging any price $X \in \mathbb{R}$, Alice may be happy to know X to a pre-assigned precision Δ. So imagine that Alice "quantizes" the random variable X into the random variable X^Δ that takes values x_i which are defined as[221]

$$p(x_i)\Delta = \int_{i\Delta}^{(i+1)\Delta} dx p(x)\,, \tag{275}$$

for all integer $i = 0, \pm 1, \pm 2, \ldots$. With this definition, the distribution of X^Δ is defined as $P\{X^\Delta = x_i\} = p(x_i)\Delta$, which is the probability that $X \in [i\Delta, (i+1)\Delta)$. She can now give a precise estimate of the information content of Bob's answer, which is the entropy $H[X^\Delta]$ of X^Δ. For $\Delta \ll 1$, this can be expressed as

[221] Because of the mean value theorem for integrals, $x_i \in [i\Delta, (i+1)\Delta]$ is inside the interval of integration.

$$\begin{aligned} H[X^\Delta] &= -\sum_i p(x_i)\Delta \log_2 [p(x_i)\Delta] \\ &= -\sum_i \int_{i\Delta}^{(i+1)\Delta} dx p(x) \log_2 p(x_i) - \log_2 \Delta \simeq h[X] - \log_2 \Delta \end{aligned} \tag{276}$$

where the approximation gets more an more precise as $\Delta \to 0$. Here $h[X]$ is defined in Eq. (274), and it is called *differential entropy*. Its meaning is that $h[X] - \log_2 \Delta$ is the expected number of bits needed to specify X to a precision Δ. The fact that $h[X]$ may not be positive is not a problem. For example, a uniform random variable $X \in [0, a]$ has $h[X] = \log_2 a$ which is negative if $a < 1$. If $a = 1/8$ and you want to determine X up to the n^{th} binary digit (i.e. $\Delta = 2^{-n}$), you will need $n - 3$ bits, because the first three bits will be zero anyhow.

One property of the entropy that we used, is that $H[X]$ does not actually depends on what values X takes. It only depends on the value of the probabilities $p_x = P\{X = x\}$. In particular, if we do a bijective transformation $X \to Y = f(X)$ – i.e. such that to every possible value of X there corresponds one and only one value of Y – then $H[X] = H[Y]$.

This is not true for the differential entropy, because even when $f(x)$ is monotonous – and hence to every X there correspond one and only

one $Y = f(X)$ – the pdf transforms as $p_Y(y) = p_X(x)/|f'(x)|_{x=f^{-1}(y)}$. Therefore

$$h[Y] = h[X] + \mathbb{E}\left[\log_2 |f'(X)|\right] . \tag{277}$$

Hence, the differential entropy is not reparametrization invariant[222]. A simple application of this is that, if a is a constant, then $h[X + a] = h[X]$ and $h[aX] = h[X] + \log_2 |a|$.

[222] **Exercise:** compute the differential entropy for a Gaussian with mean μ and variance σ^2, for an exponential distribution $p(x) = ae^{-ax}$, $a, x > 0$, and for a multi-dimensional Gaussian with mean $\vec{\mu}$ and covariance $\mathrm{Cov}[X_i, X_j] = A_{i,j}$.

Relative entropy

Imagine now that A has a wrong estimate q_x of the probability p_x of B's answers x. How much this impacts on the efficiency of the questions she's going to ask?

Given q, A is going to effectively encode B's answers in such a way that answer x will require $\log_2 1/q_x$ bits, so the number of questions she will ask, on average, is

$$\mathbb{E}\left[\log_2 \frac{1}{q}\right] = \sum_{x\in\mathcal{X}} p_x \log_2 \frac{1}{q}_x$$

the difference between this and the most efficient representation, which requires $H[p]$ bits, is

$$D_{KL}(p||q) = \sum_{x\in\mathcal{X}} p_x \log_2 \frac{p_x}{q_x}$$

which is known as the *Kullback-Leibler divergence* or *relative entropy*. It tells us how costly is the error in the estimate of probabilities, in bits. In this sense, D_{KL} is a measure of how "far" Alice is from the true distribution. This is why D_{KL} is often considered as a distance, though it is not symmetric[223] and it does not satisfy the triangle inequality[224].

Though it is not evident $D_{KL}(p||q) \geq 0$ and it vanishes only for $q = p$. The way to prove it, is to use the convexity of the logarithm $\log_2 x \leq (x-1)/\log 2$ in the definition of D_{KL}, i.e.

$$\begin{aligned} D_{KL}(p||q) &= -\sum_{x\in\mathcal{X}} p_x \log_2 \frac{q_x}{p_x} && (278)\\ &\geq -\frac{1}{\log 2}\sum_{x\in\mathcal{X}} p_x \left[\frac{q_x}{p_x} - 1\right] = 0 && (279)\end{aligned}$$

because of normalisation of p_x and q_x.

[223] **Exercise:** A coin can either be fair with $P\{\text{head}\} = P\{\text{tail}\} = 1/2$, or biased, with $P\{\text{head}\} = p$ and $P\{\text{tail}\} = 1 - p$. Show that it is worse to assume that the coin is biased when it is not, than to assume that it is fair when it is biased.

[224] See Theorem 11.6.1 in Cover for an example where $D_{KL}(p||q)$ satisfies the opposite of the triangle inequality.

The Kullback-Leibler divergence (or relative entropy) generalises to continuous variables as

$$D_{\mathrm{KL}}(p||q) = \int dx p(x) \log \frac{p(x)}{q(x)} \tag{280}$$

Contrary to the differential entropy, the relative entropy is reparametrization invariant. If p and q represent two possible distributions for the

random variable X, their divergence remains the same if one changes parametrization[225] $Y = f(X)$. As for discrete variables, it is easy to see that $D_{\mathrm{KL}}(p||q) \geq 0$ with equality holding only if $p = q$ (apart from sets of measure zero).

[225] **Exercise:** Why?

Mutual information

Imagine you have two random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ with joint distribution $p(x,y)$ and marginals $p(x)$ and $p(y)$[226]. One way to quantify their mutual dependence is to compute how much information is lost by assuming that they are independent. This is given by

[226] The abuse of the symbol $p(\cdot)$ follows the notation of COVER. It should be understood that $p(x)$ and $p(y)$ are different functions of their arguments.

$$\begin{aligned} I(X,Y) &= D_{KL}\left[p(x,y)||p(x)p(y)\right] && (281)\\ &= \sum_{x\in\mathcal{X},y\in\mathcal{Y}} p(x,y)\log_2\frac{p(x,y)}{p(x)p(y)} && (282)\\ &= H(X)+H(Y)-H(X,Y) && (283)\end{aligned}$$

and it is called the *mutual information* between X and Y. The last equality, which follows from simple algebra, with the positivity of D_{KL} implies that $H(X,Y) \leq H(X) + H(Y)$. In other words, *the state of maximal ignorance about two random variables X and Y corresponds to the case where they are independent.*

In the same way, one can define the mutual information $I[X,Y]$ between continuous variables as

$$I[X,Y] = D_{\mathrm{KL}}[p(x,y)||p(x)p(y)] = h[X]+h[Y]-h[X,Y] \qquad (284)$$

where

$$p(x) = \int dy p(x,y), \quad p(y) = \int dx p(x,y),$$

are the marginal distributions. This implies that $I[X,Y] \geq 0$ with equality if and only if X and Y are independent. So the mutual information provides a universal measure of statistical dependence. It is universal also because, the mutual information is invariant under any transformation $(X,Y) \to (U,V)$ of the random variables, where $U = f(X)$ and $V = g(Y)$ with $f(x)$ and $g(y)$ monotonous functions. These transformations changes the "shape" of the distributions of the two variables, but leaves their statistical dependence invariant. This invariance becomes manifest if we apply the transformation $f(x) = P\{X \leq x\}$ and $g(y) = P\{Y \leq y\}$ which transforms X and Y into two uniform random variables U and V. The mutual information can then be expressed as

$$I(X,Y) = \int_0^1 du \int_0^1 dv c(u,v)\log_2 c(u,v), \qquad (285)$$

where[227]

[227] **Exercise:** Prove Eqs. (285) and (286).

$$c(u,v) = \frac{\partial^2}{\partial u \partial v} C(u,v).$$

and the function $C(u,v)$ is the joint cumulative distribution of U and V, defined as

$$P\{X \leq x, Y \leq y\} = C\left(P\{X \leq x\}, P\{Y \leq y\}\right). \tag{286}$$

The function $C(u,v)$ is called the *copula function* of the two random variables X and Y[228].

In order to illustrate the meaning of I consider the following problem. We are interested in estimating a random variable X of which at present we know the distribution $p(x)$, and the corresponding entropy $H[X]$ which quantifies our state of uncertainty about X. You can think of X as a parameter of a theory of a given system[229]. Now we have the possibility to perform an experiment, i.e. to measure a random variable Y, of which we know, before doing the experiment, its distribution. We also know the joint distribution $p(x,y)$ of the two variables. How much information can I expect the experiment will convey on X? The reduction in the uncertainty is given by

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

as can be shown by a direct calculation. So the mutual information tells us how much we learn, on average, about X if we know Y. Note that the mutual information is symmetric

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

In other words, the amount of information that we can gain about a theory by performing an experiment, is exactly equal to the uncertainty that the theory provides on the outcome of the experiment[230].

Another important point is that knowledge of Y reduces *a priori* the uncertainty on X, since $H(X|Y) \leq H(X)$, but *a posteriori* this might not be the case! Take, for example, two random variables $X, Y \in \{1,2\}$, with a joint distribution:

$$p(x,y) = \begin{cases} 0 & \text{if } x = y = 1 \\ 3/4 & \text{if } x = 2 \text{ and } y = 1 \\ 1/8 & \text{if } x = 1 \text{ and } y = 2 \text{ or if } x = y = 2 \end{cases} \tag{287}$$

Then $H(X) \simeq 0.544$ and $H(X|Y) = 0.25$ bits, i.e. $I(X,Y) \simeq 0.294$ bits. However if the outcome $Y = 2$ occurs, the uncertainty on X actually increases, because $H(X|Y=2) = 1$ bit. It is instructive to check the opposite. Does the uncertainty on Y decreases no matter what value X turns out to take or not? This should give you a sense of what are the conditions under which the uncertainty may increase after a measurement[231].

[228] Eqs. (285) and (286). suggest an easy way to check whether two variables are dependent or not, based on a sample $(X_1,Y_1),\ldots,(X_n,Y_n)$ of n joint observations. Let $U(x)$ and $V(y)$ be the fraction of points for which $X_i \leq x$ and $Y_j \leq y$, respectively. Plot the points $(U(X_i), V(Y_i))$ in the (u,v) plane. If X and Y are independent, the n points should be uniformly distributed in the unit square $[0,1]^2$. Statistical dependence is spotted by the clustering of points in some region. This plot reveals not only whether X and Y are dependent or not, but also how they depend on each other. For example a monotonous dependence (e.g. if X increases Y tends to increase or decrease) corresponds to points clustering on one of the diagonals of the square. This is the kind of dependence which is usually quantified by covariance measures. Yet there are many other possibilities of how X and Y can depend on each other, some of which may not be detectable by covariance.

[229] $I(X,Y)$ is the reduction of Alice's uncertainty on X if, instead of asking Bob about X, she asks Carl about a different variable Y.

[230] **Exercise:** Problem 131 of Bialek's book, *Biophysics*. Let there be $n+1$ boxes labeled $\omega = 0,1,\ldots,n$, with n even. One of the boxes contains a prize, the others are empty. The probability that the prize is in box ω is p_0 for $\omega = 0$ and $(1-p_0)/n$ for all $\omega > 0$. We have two available strategies:
1) open the box $\omega = 0$
2) open the last $n/2$ boxes ($\omega > n/2$)
Which one is the most convenient? Which one conveys more information on where the prize actually is?
Draw a plot of the threshold p_0^* for which strategies 1 and 2 are equivalent, according to the two criteria.

This is a toy model for a situation where a phenomenon can be explained by alternative theories, one of which is the prevailing one, whereas the others are very unlikely but are many. The two options correspond to two possible experiments, one that tries to refute or confirm the prevailing theory, the other that can exclude half of the unlikely ones. Check that even if $p_0 = 0.99$ it might be more informative to exclude unlikely theories if $n > 270$.

[231] **Exercise:** Generalise this example to the case where $P\{X=2, Y=1\} = a$ and $P\{X=1, Y=2\} = b$ and $P\{X=2, Y=2\} = 1-a-b$. What is the values of a and b for which no measurement of one of the variables can increase the uncertainty on the other? Are there values of a, b such that measuring any of the two variables will increase the uncertainty on the other?

The data processing inequality

Information is degraded at every passage, as we know from everyday life. Imagine that Alice communicates a message X to Bob, and Bob refers the message to Carl. The message Y that Bob receives may be corrupted by noise, so $Y \neq X$, likewise Carl receives a message Z that may be different from Y. Formally we represent the situation by saying that X, Y and Z are three random variables that form a *Markov chain*, denoted as[232]

$$X \to Y \to Z$$

which means that

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

As a consequence, conditional to Y, X and Z are independent, because $p(x, z|y) = p(x|y)p(z|y)$. Note also the the directions of the arrows can be reversed by using Bayes rule, so $X \to Y \to Z$ is equivalent to $Z \to Y \to X$.

For a Markov chain $X \to Y \to Z$, the *Data-processing inequality* states that

$$I(X, Z) \leq I(X, Y)\,. \tag{288}$$

In words, the information that Y contains on X cannot be increased, whatever transformation $Y \to Z$ one can apply. This result is important in statistics, because it suggests that any manipulation of the data can only decrease the information content of the data.

The proof of the inequality (288) is simple. The mutual information between X and $W = (Y, Z)$ can be written in two ways

$$\begin{aligned} I(X, W) &= \mathbb{E}\left[\log_2 \frac{p(X, Y, Z)}{p(X)p(Y, Z)}\right] && (289)\\ &= \mathbb{E}\left[\log_2 \frac{p(X, Z|Y)}{p(X)p(Z|Y)} \frac{p(X|Y)}{p(X|Y)}\right] && (290)\\ &= I(X, Y) + I(X, Z|Y) && (291)\\ &= I(X, Z) + I(X, Y|Z) && (292) \end{aligned}$$

where

$$I(X, Y|Z) = \mathbb{E}\left[\log_2 \frac{p(X, Y|Z)}{p(X|Z)p(Y|Z)}\right]$$

is the conditional mutual information of X and Y given Z. In Eq. (291) the term $I(X, Z|Y) = 0$ vanishes, because X and Z are independent, conditional on Y. The inequality (288) follows from the fact that $I(X, Y|Z) \geq 0$.

[232] We mention in passing that this notion generalises to *Markov fields*, that specify the dependence between n random variables with a *graphical model* of n nodes which are connected by links (or hyperlinks) if the corresponding variables are dependent.

There are other general inequalities that can be derived from basic laws. For example the mutual information between X_1 and X_2 cannot be larger than the average of the two entropies. See the book

Raymond W Yeung. *Information theory and network coding*. Springer Science & Business Media, 2008

The entropy of Markov Chains

Let us consider Markov chains, i.e. sequences $\underline{X} = (X_1, \dots, X_N)$ of random variables generated by a transition probability matrix

$$P\{X_t = s | X_{t-1} = s'\} = p_{s,s'}$$

with s, s' being elements of a finite set $\mathcal{S}$. We restrict attention to irreducible chains, for which we know that the probability to observe state $X_t = s$ converges to the invariant measure $\mu_s = \sum_{s'} p_{s,s'} \mu_{s'}$. We further assume that we know that the sequence is sampled in the stationary state, i.e. $P\{X_1 = s\} = \mu_s$. Then, the probability of the sequence is given by

$$P\{\underline{X}\} = p_{X_N, X_{N-1}} p_{X_{N-1}, N-2} \cdots p_{X_2, X_1} \mu_{X_1} \,. \tag{293}$$

Note that the time index goes from right ($t = 1$) to left ($t = N$) in this equation. Taking the logarithm and dividing by N, please check[233] that the law of large numbers implies

$$\lim_{N\to\infty} -\frac{1}{N} \log P\{\underline{X}\} = H[X_t | X_{t-1}] \equiv -\sum_{s,s'} p_{s,s'} \mu_{s'} \log p_{s,s'} \,. \tag{294}$$

[233] **Exercise:** do it.

Note that the entropy of the sequence is smaller than N times $H[X_t] = -\sum_s \mu_s \log \mu_s$, because knowledge of X_{t-1} provides information on X_t, in contrast with the i.i.d. case. From the point of view of the Asymptotic Equipartition property, sequences of N random variables explore a smaller space than that of N i.i.d. random variables drawn from μ_s.

Irreversibility and the arrow of time

Imagine that we do not know whether the sequence $\underline{X}$ has been given to us in the right order – with time going from 1 to N – or in the reverse one – with time going from N to 1. Can we figure this out? In order to do this, let us refine our notation and call $P\{\underline{X}\} = P_{\rightarrow}\{\underline{X}\}$, as defined in Eq. (293), to distinguish it from from the backward probability[234]

$$P_{\leftarrow}\{\underline{X}\} = p_{X_1, X_2} \cdots p_{X_{N-2}, N-1} p_{X_{N-1}, X_N} \mu_{X_N} \,. \tag{295}$$

[234] **Exercise:** Show that the naïve generalisation of Eq. (294)

$$\log P_{\leftarrow}\{\underline{X}\} \simeq -H[X_{t-1} | X_t]$$

is wrong. Show also that $H[X_{t-1}|X_t] = H[X_t|X_{t-1}]$ in the stationary state. In loose words, given the present, the past is as uncertain as the future in a Markov chain.

The probability of the sequence $\underline{X}$ can also be expressed in terms of the reverse Markov chain with transition matrix $q_{s,s'} = p_{s',s} \mu_s / \mu_{s'}$, as

$$Q_{\leftarrow}\{\underline{X}\} = q_{X_1, X_2} \cdots q_{X_{N-2}, N-1} q_{X_{N-1}, X_N} \mu_{X_N} = P_{\rightarrow}\{\underline{X}\} \tag{296}$$

$$Q_{\rightarrow}\{\underline{X}\} = q_{X_N, X_{N-1}} q_{X_{N-1}, N-2} \cdots q_{X_2, X_1} \mu_{X_1} = P_{\leftarrow}\{\underline{X}\} \tag{297}$$

where the proof of the last equalities relies on repeated use of the identities $q_{X_{t-1}, X_t} \mu_{X_t} = p_{X_t, X_{t-1}} \mu_{X_{t-1}}$. For large N, the probability of

the sequence in the reverse process is given by

$$\frac{1}{N}\log P_{\leftarrow}\{\underline{X}\} = \frac{1}{N}\sum_{t=2}^{N}\log p_{X_{t-1},X_t} + \frac{1}{N}\log \mu_{X_t} \quad (298)$$

$$= \sum_{s,s'}\frac{k_{s,s'}}{N}\log p_{s',s} + \frac{1}{N}\log \mu_{X_t} \quad (299)$$

where $k_{s,s'}$ is the number of transitions from s' to s in the sequence $\underline{X}$. As $N \to \infty$, the fraction $k_{s,s'}/N$ of transitions $s' \to s$ converges to the probability $p_{s,s'}\mu_{s'}$. Therefore

$$\lim_{N\to\infty}\frac{1}{N}\log P_{\leftarrow}\{\underline{X}\} = \sum_{s,s'} p_{s,s'}\mu_{s'}\log p_{s',s} = \sum_{s,s'} p_{s,s'}\mu_{s'}\log q_{s,s'} \quad (300)$$

where the proof of the last equality is left as an exercise. Therefore, for large N

$$P_{\leftarrow}\{\underline{X}\} \simeq P_{\rightarrow}\{\underline{X}\}e^{-N\Sigma} \quad (301)$$

where

$$\Sigma \equiv D_{KL}(P_{\rightarrow}||P_{\leftarrow}) = \sum_{s,s'} p_{s,s'}\mu_{s'}\log\frac{p_{s',s}}{p_{s,s'}} = \sum_{s,s'} p_{s,s'}\mu_{s'}\log\frac{p_{s,s'}}{q_{s,s'}} \quad (302)$$

is called the *entropy production*. As long as $q_{s,s'} \neq p_{s,s'}$, the probability of the forward process is exponentially (in N) more likely than the backward one, because $D_{KL}(P_{\rightarrow}||P_{\leftarrow}) > 0$. Hence given the transition matrix $p_{s,s'}$, we can detect the arrow of time because the two transition probabilities $p_{s,s'}$ and $q_{s,s'}$ are different and they define two distinguishable stochastic processes. Furthermore, notice that the Kullback-Leibler divergence is symmetric in this case, i.e. $D_{KL}(P_{\leftarrow}||P_{\rightarrow}) = D_{KL}(P_{\rightarrow}||P_{\leftarrow})$. This reflects the mirror symmetry of the directions of the time arrow: the forward arrow of time under the reverse process is as unlikely as the backward arrow under the forward Markov chain.

If, instead, the Markov chain is reversible, i.e. $q_{s,s'} = p_{s,s'}$, then there is no way in which the arrow of time can be detected.

The entropy production is a measure of how much the forward process is more likely than the reversed one, which is expressed in Eq. (302) as the difference between the logarithms of the forward and the backward transition probabilities. Indeed irreversibility is related to the existence of a *probability current*, whereby these two terms do not cancel each other and then net probability flow is non-zero[235].

[235] **Exercise:** Show that a Markov chain with two states is always reversible. Irreversibility requires at least three states and a probability current that either runs clockwise or counter-clockwise through the states.

Data compression and coding theory

Data compression deals with the problem of optimally representing messages. We refer to Chapter 5 of COVER for a detailed discussion. This is a short summary of the main ideas. The relation between information theory and coding was already hinted at in the introduction. As discussed there, the typical setting is the one where Alice and Bob need to communicate using a binary channel. Then Alice will *encode* her messages to Bob in a string of bits, transmit this string over the channel, and Bob will read it and *decode* it to get the original message. A message $\underline{X} = (X_1, \dots, X_n)$ is a sequence of symbols $X_i \in \chi$ drawn from an alphabet χ. The simplest example is a text (e.g. a book) which is a sequence of ASCII characters (letters, numbers, spaces, punctuation, etc). But you can likewise think of images, e.g. digital pictures of paintings, as sequences of RGB values for each pixel. Ultimately, each message is stored in digital devices in the form of sequences of zeros and ones, so there is a function $C(\underline{X})$ that associates to each message $\underline{X}$ a string $C(\underline{X})$ of bits. Coding theory deals with the problem of finding ways of representing the data as efficiently as possible, i.e. with the minimal number of bits. Each bit can be thought of as the answer to a yes/no question, so efficient coding, i.e. the problem of optimally[236] representing information, coincides with the problem of eliciting information in an optimal manner, that we already discussed.

Coding theory enters into play, for example, when you use a data compression algorithm (e.g. gzip) on your computer that transforms a text file written in ASCII code into a file that occupies less space on the hard disk of your computer. Compression is possible because messages contain regularities. For example, if the character "q" is always followed by "u" in a text, a code that translates "q" and "u" by different sequences of bits (called codewords) is less efficient than one that codes the pair "qu" directly. Indeed, what the compression program does when you invoke it, is to scan the file you want to compress in search of regularities, i.e. of patterns that occur very frequently. Formally we shall consider messages as being generated as random draws from a probability distribution. Then the knowledge of the probability distribution is what makes optimal compression possible. This is why probability theory, coding theory and information theory are so intimately connected[237].

The main result in coding theory, due to Shannon, makes this connection explicit in the simple case of messages generated as i.i.d. draws from a a distribution $p(x)$ with $x \in \chi$. We already discussed Shannon's theorem when we introduced information theory. Let us briefly recall it. Shannon theorem is a consequence (or restatement) of the Asymptotic Equipartition Property. The latter says that, almost surely a message

[236] In the sense of most parsimoniously.

[237] A theatre play, such as Othello, is an example of a message, because it is a sequence of letters. It is definitely true that any understanding of the production of Shakespeare has to do with a better understanding of the regularities that one can find in his works. Yet, thinking of his works as being generated as a random draw from a probability distribution seems somewhat extreme, and it is at best an approximation. The simplest such approximation is to think of each letter as being drawn independently at random from a probability distribution. The fact that letters from 'a' to 'z' do not occur with the same probability allows a certain degree of compression of Othello. Furthermore, one realises that certain words (e.g. 'the' or 'and') occur much more frequently than others (e.g. 'Iago') and some (e.g. 'yqat') never occur. This leads to better approximations of the generative process, which affords further compression. Furthermore, the occurrence of words depends on the occurrence of other words in the same act or even in other acts. The more regularities one detects the better one can compress Othello. Note that some of these features are generic of English texts some are generic of Shakespeare's production and some are specific of Othello.

$\underline{X} = (X_1, \ldots, X_n)$ composed of characters drawn independently from the same distribution $p(x)$ belongs to the set A_n of typical sequences, which contains $|A_n| \sim 2^{nH[X]}$ elements. If we label all messages $\underline{X} \in A_n$ with an integer $C(\underline{X})$ we can take the binary representation of $C(\underline{X})$ as the code[238]. Then, almost surely for $n \to \infty$, $C(\underline{X}) \leq 2^{nH[X]}$ which means that at most $H[X]$ bits per character are needed to transmit a message.

[238] A good way of labelling messages is by their rank in probability, from the most probable to the least probable. You can check that in this way at most $H[X]$ bits per character are needed to transmit a message.

This strategy, however, is not very practical because the calculation of $C(\underline{X})$ requires ranking all messages which are exponentially many in n. This is practically unfeasible. Also if a message is composed of two parts $\underline{X} = (\underline{X}_1, \underline{X}_2)$ the code of $\underline{X}$ is not easily related to those of its parts. For messages where X_i are drawn as i.i.d. variables from the same distribution, it may be more practical to consider codes such that

$$C(\underline{X}) = (c(X_1), \ldots, c(X_n))$$

that are sequences of *codewords* $c(x)$ each of which corresponds to a character $x \in \chi$. So the key question is, how should the function $c(x)$ be chosen?

We already encountered examples of codes in the introduction, for the case where $\chi = \{a, b, c, d\}$ has four elements, reported on the right[239]. This should allow one to translate each sequence of bits, such as

$$0010110101001\ldots$$

into a sequence of characters in χ. A minimal requirement for codes is that they be *uniquely decodable*. This means that any sequence of bits that is produced by translating a sequence of characters should be decodable in a unique manner. This does not happen if there are two or more sequences of characters $\underline{X}$ that correspond to the same sequence of bits. The three codes c_1, c_2 and c_3 satisfy this property. For example, c_1 would translate that sequence into $cbabbc\ldots$ whereas c_2 will give $dbaccd\ldots$. In both cases, the translated sequence can be computed as we scan the sequence of bits from left to right. Codes that have this property are called *instantaneous* codes, because they allow to instantaneously translate bit-strings into messages. The key property that makes a code an instantaneous code is that no codeword is the prefix of another codeword, i.e. no codeword coincides with the leftmost part of another codeword.

[239] Four examples of codes:

χ	$c_1(x)$	$c_2(x)$	$c_3(x)$	$c_4(x)$
a	1	11	0	00
b	01	10	010	01
c	001	01	01	10
d	000	00	10	000

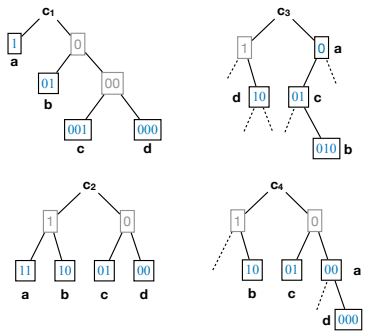


Figure 34: Representation of the codes c_1, c_2, c_3 and c_4 as trees.

This is not true for c_3 for which $c_3(a)$ is a prefix of $c_3(b)$ and $c_3(c)$, for example. In this case it is not possible to figure out what the translation of the leftmost bits is unless one considers also the bits that come after. For example, according to c_3, the first 0 in the sequence above could correspond to a or to the beginning of the codewords for b or c. However the latter two options should be discarded because

the second bit is a 0, which is not compatible with either a b or a c. If the first character is an a the second can be a b or a c. Yet it cannot be a b because otherwise the bits that follow $11\ldots$ do not correspond to a decodable sequence (c_3 has no codewords that starts with 11). So the first characters should be $accddd\ldots$ but the next characters depend on what the following characters are. Finally code c_4 is not uniquely decodable. For example the bit string 000000 could either be aaa of dd.

We shall focus on instantaneous codes only. Each of these codes admits a representation as a tree, as shown in Fig. 34. In this tree, the codewords correspond to the leaves (the terminal nodes) and the length $\ell(x) = |c(x)|$ of each codeword (i.e. the number of bits) corresponds to the distance of the corresponding node from the root (which is the top most node). For each instantaneous code $c(x)$, the lengths $\ell(x)$ satisfy Kraft's inequality

$$\sum_{x\in\chi} 2^{-\ell(x)} \leq 1. \tag{303}$$

This is very easily proven[240].

With some more effort one can show (see Cover) that for any set of lengths $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_{|\chi|}\}$ that satisfy Kraft's inequality Eq. (303), i.e. such that $\sum_i 2^{-\ell_i} \leq 1$, there is at least one instantaneous code $c(x)$ such that the lengths $|c(x)|$ match exactly the ℓ_i's .

Among all instantaneous codes, we want to find those that make the expected length of the bit-string it produces, when characters are drawn from a distribution $P\{X = x\} = p_x$, as short as possible. The two results above imply that it is enough to find a set $\mathcal{L}$ of lengths that satisfy Kraft's inequality and we're guaranteed that an instantaneous code with those lengths exists. So it is enough to solve the problem

$$\min_{\mathcal{L}} \mathbb{E}\left[\ell(X)\right] \tag{304}$$

over all sets $\mathcal{L} = \{\ell(x),\ \forall x \in \chi\}$ of lengths that satisfy Kraft's inequality. Introducing this constraint with a Lagrange multiplier, leads to the problem[241]

$$\min_{\mathcal{L},\lambda} \left[\sum_{x\in\chi} p_x \ell(x) - \lambda\left(\sum_{x\in\chi} 2^{-\ell(x)} - 1\right)\right]. \tag{305}$$

What makes this problem complicated is that $\ell(x)$ must be an integer variable. If we neglect this problem and minimise over real values of $\ell(x)$, then we're going to obtain a lower bound. The latter problem is simple and is solved by setting to zero the first order derivative of the objective function in Eq. (305). This yields $\ell(x) = -\log_2 p_x$ and

$$\min_{\mathcal{L}} \mathbb{E}\left[\ell(X)\right] \geq H[X] = -\sum_{x\in\chi} p_x \log_2 p_x. \tag{306}$$

If you take the smallest integer $\ell(x)$ which is larger than $-\log_2 p_x$, then you can get better estimate of the minimal expected length. The smallest

[240] Proof: let $\bar{\ell} = \max_{x\in\chi} \ell(x)$. Then continue the tree to all nodes at distance $\bar{\ell}$ from the root. For each word x, this results in $2^{\bar{\ell}-\ell(x)}$ nodes at distance $\bar{\ell}$ down the codeword corresponding to x. The number of these nodes is $\sum_{x\in\chi} 2^{\bar{\ell}-\ell(x)}$. This number has to be smaller than the total number of nodes at distance $\bar{\ell}$ from the root, which is $2^{\bar{\ell}}$. This leads to Eq. (303).

Exercise: Indeed there is more than one code that corresponds to the same lengths. Count the number of codes which have the same lengths as the codes c_1 and c_2.

[241] Note the sign of the λ term. The most efficient codes are those which have shorter codewords, so those for which the left hand side of Eq. (303) is as large as possible, i.e. for which Eq. (303) is satisfied as an equality.

Exercise: check that c_1 and c_2 satisfy Kraft's inequality as an equality whereas c_3 does not satisfy it. What about c_4? Can you find an instantaneous code for which Kraft's inequality is not satisfied as an equality?

integer larger than $-\log_2 p_x$ is smaller than $-\log_2 p_x + 1$. Therefore the expected length must be smaller than $H[X] + 1$. Taken together these results show that for any X there is an instantaneous code that allows to represent X with an expected number of bits that is bounded by

$$H[X] \leq \min_{\mathcal{L}} \mathbb{E}\left[\ell(X)\right] \leq H[X] + 1. \tag{307}$$

This result can be improved by invoking *block coding*. This means that, in sending a message $\underline{X} = (X_1, \ldots, X_n)$ with $n \gg 1$, instead of using codes that translate each X_i separately, we can look for the instantaneous codes that translate a pair X_i, X_{i+1} of successive variables, or a subsequence $\underline{X}_i^{(m)} = (X_{i+1}, \ldots, X_{i+m})$ of m successive characters. The same argument that we have applied above implies that

$$H[\underline{X}_i^{(m)}] \leq \min_{\mathcal{L}} \mathbb{E}\left[\ell(\underline{X}_i^{(m)})\right] \leq H[\underline{X}_i^{(m)}] + 1.$$

However $H[\underline{X}_i^{(m)}] = mH[X]$ which means that block coding can achieve a compression that satisfies

$$H[X] \leq \frac{1}{m} \min_{\mathcal{L}} \mathbb{E}\left[\ell(\underline{X}_i^{(m)})\right] \leq H[X] + \frac{1}{m}. \tag{308}$$

This result, for $m \to \infty$, coincides with Shannon's bound that ensures that at most $H[X]$ bits per character need to be exchanged by Alice and Bob in order to communicate messages generated from the distribution p_x.

This derivation also tells us how optimal codes should look like. Indeed the equation $\ell(x) = -\log_2 p_x$ tells us that short codewords should be assigned to most probable characters. The Huffman coding algorithm, for example, is based on the idea of iteratively assigning bits to the least probable values of x, by grouping them together[242]. We refer to COVER for a detailed discussion of this and other algorithms.

Data compression is only the simplest of the problems discussed in coding theory. A different class of problems have to do with the fact that most daily life communication channels are affected by noise. The string of bits in output is not equal to the one in input, because some bits may be turned from 0 to 1 or viceversa. Communication over noisy channels requires *error correcting codes*, i.e. codes with a built in redundancy that can help recover the original message, even if that was corrupted by noise. This is a fascinating subject which we will not discuss, however. Yet again, the solution has to do with understanding what the typical messages that need to be transmitted are and how typically they would be corrupted by noise. This allows to get precise bounds, again in terms of entropies, on the amount of redundancy that needs to be embedded in messages, in order to achieve an error free communication.

[242] **Huffman codes:** Huffman coding algorithm reconstruct the tree from the bottom, starting from a partition of the set χ of words into singleton sets $\{x\}$ with an associated probability p_x. At every step, the algorithm generates a new partition from the old one by merging the two sets $\mathcal{S}$ and $\mathcal{S}'$ with the smallest probability, assigning to the new set $\mathcal{S} \cup \mathcal{S}'$ the sum of the probabilities $p_{\mathcal{S}\cup\mathcal{S}'} = p_{\mathcal{S}} + p_{\mathcal{S}'}$. At the same time, the algorithm assigns bits 0 and 1 to the edges joining the nodes corresponding to $\mathcal{S}$ and $\mathcal{S}'$ to $\mathcal{S} \cup \mathcal{S}'$. The algorithm ends when the partition formed by the single set χ is reached, i.e. when all words are merged in the same set. The codeword of x is given by the sequence of bits associated to all the merging of sets $\mathcal{S}$ that contain x, starting from the root χ, down to the set $\{x\}$.

If you understood the main gist of the arguments discussed above, then you may consider pondering on the following questions:[243]

1. What do you expect the sequence of bits of an optimally compressed sequence $X_1, \ldots, X_n$ should look like? What is the probability that a (randomly chosen) bit is equal to one? What is the difference of this sequence from a sequence of random i.i.d. bits?

2. In all our discussion we have assumed a binary alphabet for the codes. Yet the same results can be derived for codes in an alphabet with three different characters (e.g. 0, 1 and 2), or the 26 characters of the English alphabet. How would this change the results, e.g. Eq. (303) and Eq. (308)?

3. Languages (e.g. English, French, Chinese, etc) might be though of as the codes that we use to communicate. A text is a representation of something (an object, a concept, an idea, etc) that is coded as a sequence of characters. Yet, if you look at texts as coded messages, the coding looks rather inefficient. For example, you may delete a certain fraction of characters from a text but still be able to reconstruct the entire text or grasp the gist of the text. The most frequent words in a text (e.g. "the", "and", "this", etc) do not carry any meaning[244] and the least frequent words are very informative on the content of the text. There is a lot of (apparently useless) redundancy in language. Why did humans converged to such inefficient ways of communicating?

[243] **Exercise:** Let a text be generated by first drawing a subject $Z \in \mathcal{Z}$ and then a message $\underline{X} = (X_1, \ldots, X_n)$ of n characters $X_i \in \mathcal{X}$ drawn independently from a distribution $p(x|z) = P(X_i = x|Z = z)$. There are two possible strategies: *A)* use the same code irrespective of the subject, and *B)* first code the subject Z and then code the text $\underline{X}$ depending on the subject (two way code). Note that code B represents each text $\underline{X}$ optimally, at the expense of the extra cost of coding Z, whereas texts $\underline{X}$ are never coded optimally with strategy A, with an over expenditure of bits that should grow with n. Show that, irrespective of this, the two way code B is never the best one.

[244] George Zipf found that for a text like the Holy Bible, the frequency with which the r^{th} most frequent word occurs is roughly inversely proportional to r. This is true for many texts (not for phone directories) and for texts written in different languages. Note that this also implies that the number of occurrences of words used k times is inversely proportional to k. This is reminiscent of the Asymptotic Equipartition Property, that states that the number of typical sequences is inversely proportional to their probability. Is this a coincidence or does it hints to the fact that our language has evolved so that text shares some statistical properties with typical sequences?