Mathematical Foundations of Data Sciences

Gabriel Peyré
CNRS & DMA
École Normale Supérieure
gabriel.peyre@ens.fr
https://mathematical-tours.github.io
www.numerical-tours.com

November 18, 2020

Chapter 1

Shannon Theory

Shannon theory of information, published in 1948/1949, is made of three parts:

1. Sampling: it studies condition under which sampling a continuous function to obtain a discrete vector is invertible. The discrete real values representing the signal are then typically quantized to a finite precision to obtain a set of symbols in a finite alphabet.

2. Source coding: it studies optimal ways to represent (code) such a set of symbols as a binary sequence. It leverages the statistical distributions to obtain the most possible compact code.

3. Channel coding (not studied here): it studies how to add some redundancy to the coded sequence in order to gain robustness to errors or attacks during transmission (flip of certain bits with some probability). It is often named "error correcting codes theory".

The main reference for this chapter is [1].

1.1 Analog vs. Discrete Signals

To develop numerical tools and analyze their performances, the mathematical modelling is usually done over a continuous setting (so-called "analog signals"). Such continuous setting also aims at representing the signal in the physical world, which are inputs to sensors hardwares such as microphone, digital cameras or medical imaging devices. An analog signal is a 1-D function $f_0 \in \mathrm{L}^2([0,1])$ where $[0,1]$ denotes the domain of acquisition, which might for instance be time. An analog image is a 2D function $f_0 \in \mathrm{L}^2([0,1]^2)$ where the unit square $[0,1]^2$ is the image domain.

Although these notes are focussed on the processing of sounds and natural images, most of the methods extend to multi-dimensional datasets, which are higher dimensional mappings

$$f_0 : [0,1]^d \to [0,1]^s$$

where d is the dimensionality of the input space ($d = 1$ for sound and $d = 2$ for images) whereas s is the dimensionality of the feature space. For instance, gray scale images corresponds to $(d = 2, s = 1)$, videos to $(d = 3, s = 1)$, color images to $(d = 2, s = 3)$ where one has three channels (R, G, B). One can even consider multi-spectral images where $(d = 2, s \gg 3)$ that is made of a large number of channels for different light wavelengths. Figures 1.1 and 1.2 show examples of such data.

1.1.1 Acquisition and Sampling

Signal acquisition is a low dimensional projection of the continuous signal performed by some hardware device. This is for instance the case for a microphone that acquires 1D samples or a digital camera that

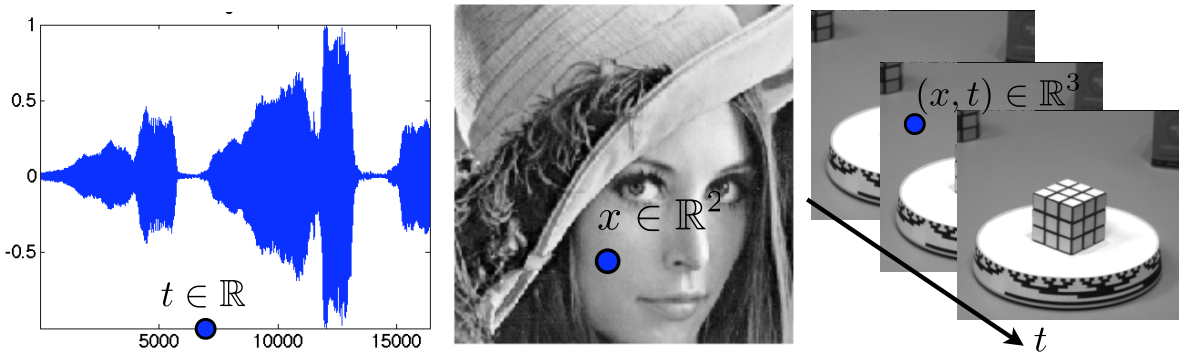

Figure 1.1: Examples of sounds ($d = 1$), image ($d = 2$) and videos ($d = 3$).

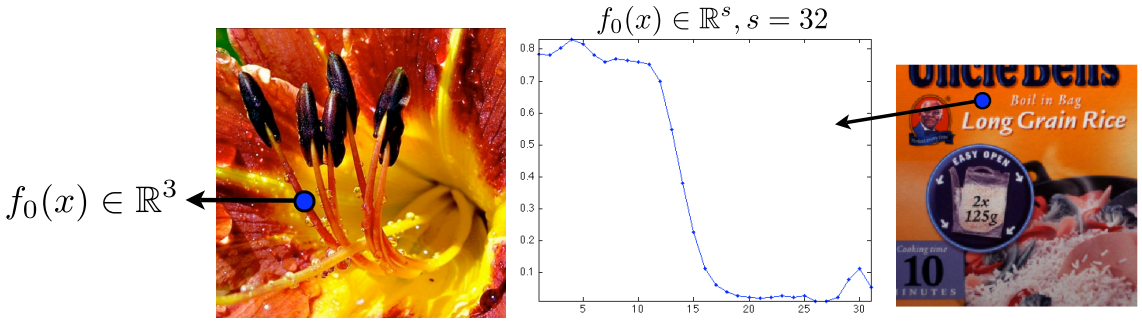

Figure 1.2: Example of color image $s = 3$ and multispectral image ($s = 32$).

acquires 2D pixel samples. The sampling operation thus corresponds to mapping from the set of continuous functions to a discrete finite dimensional vector with N entries.

$$f_0 \in \mathrm{L}^2([0,1]^d) \mapsto f \in \mathbb{C}^N$$

Figure 1.3 shows examples of discretized signals.

1.1.2 Linear Translation Invariant Sampler

A translation invariant sampler performs the acquisition as an inner product between the continuous signal and a constant impulse response h translated at the sample location

$$f_n = \int_{-S/2}^{S/2} f_0(x) h(n/N - x) \mathrm{d}x = f_0 \star h(n/N). \tag{1.1}$$

The precise shape of $h(x)$ depends on the sampling device, and is usually a smooth low pass function that is maximal around $x = 0$. The size S of the sampler determines the precision of the sampling device, and is usually of the order of $1/N$ to avoid blurring (if S is too large) or aliasing (if S is too small).

Section **??** details how to reverse the sampling operation in the case where the function is smooth.

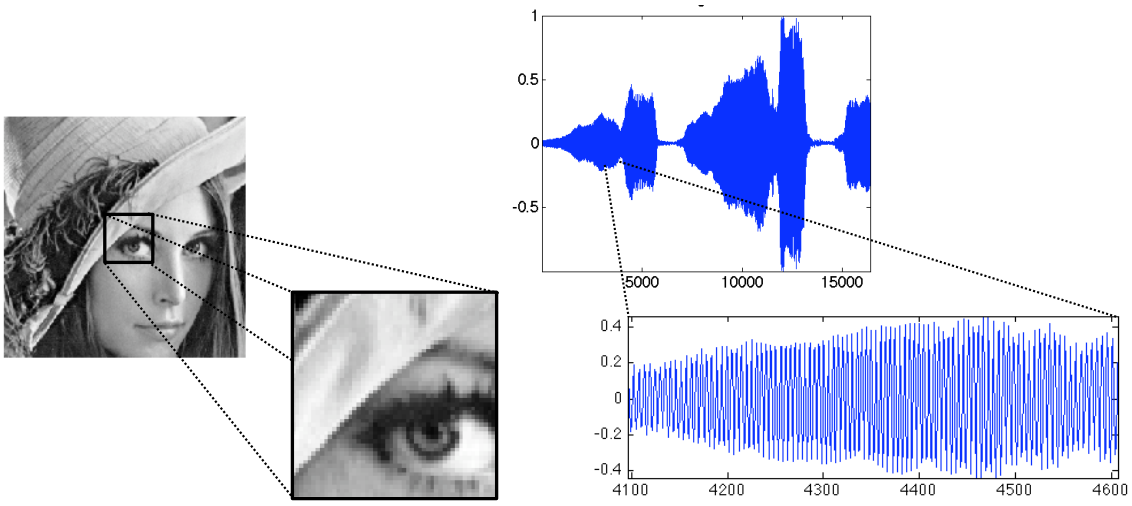

Figure 1.3: Image and sound discretization.

1.2 Shannon Sampling Theorem

Reminders about Fourier transform. For $f \in L^1(\mathbb{R})$, its Fourier transform is defined as

$$\forall\, \omega \in \mathbb{R}, \quad \hat{f}(\omega) \stackrel{\text{def.}}{=} \int_{\mathbb{R}} f(x) e^{-\mathrm{i}x\omega} \mathrm{d}x. \tag{1.2}$$

One has $\|\hat{f}\|^2 = (2\pi)^{-1}\|f\|^2$, so that $f \mapsto \hat{f}$ can be extended by continuity to $L^2(\mathbb{R})$, which corresponds to computing $\hat{f}$ as a limit when $T \to +\infty$ of $\int_{-T}^{T} f(x) e^{-\mathrm{i}x\omega}\mathrm{d}x$. When $\hat{f} \in L^1(\mathbb{R})$, one can invert the Fourier transform so that

$$f(x) = \frac{1}{2\pi} \int_{\mathbb{R}} \hat{f}(\omega) e^{\mathrm{i}x\omega} \mathrm{d}\omega, \tag{1.3}$$

which shows in particular that f is continuous with vanishing limits at $\pm\infty$.

The Fourier transform $\mathcal{F} : f \mapsto \hat{f}$ exchanges regularity and decay. For instance, if $f \in C^p(\mathbb{R})$ with an integrable Fourier transform, then $\mathcal{F}(f^{(p)})(\omega) = (\mathrm{i}\omega)^p \hat{f}(\omega)$ so that $|\hat{f}(\omega)| = O(1/|\omega|^p)$. Conversely,

$$\int_{\mathbb{R}} (1+|\omega|)^p |\hat{f}(\omega)| \mathrm{d}\omega < +\infty \quad \Longrightarrow \quad f \in C^p(\mathbb{R}). \tag{1.4}$$

For instance, if $\hat{f}(\omega) = O(1/|\omega|^{p+2})$, one obtains that $f \in C^p(\mathbb{R})$.

Reminders about Fourier series. We denote $\mathbb{T} = \mathbb{R}/2\pi\mathbb{Z}$ the torus. A function $f \in L^2(\mathbb{T})$ is 2π-periodic, and can be viewed as a function $f \in L^2([0, 2\pi])$ (beware that this means that the boundary points are glued together), and its Fourier coefficients are

$$\forall\, n \in \mathbb{Z}, \quad \hat{f}_n \stackrel{\text{def.}}{=} \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-\mathrm{i}xn} \mathrm{d}x.$$

This formula is equivalent to the computation of an inner-product $\hat{f}_n = \langle f,\, e_n \rangle$ for the inner-product $\langle f,\, g \rangle \stackrel{\text{def.}}{=} \frac{1}{2\pi} \int_{\mathbb{T}} f(x)\bar{g}(x)\mathrm{d}x$. For this inner product, $(e_n)_n$ is orthonormal and is actually an Hilbert basis, meaning that one reconstructs with the following converging series

$$f = \sum_{n \in \mathbb{Z}} \langle f,\, e_n \rangle e_n \tag{1.5}$$

which means $\|f - \sum_{n=-N}^{N} \langle f, e_n \rangle e_n\|_{L^2(\mathbb{T})} \to 0$ for $N \to +\infty$. The pointwise convergence of (1.5) at some $x \in \mathbb{T}$ is ensured if for instance f is differentiable. The series is normally convergent (and hence uniform) if for instance f if of class C^2 on $\mathbb{T}$ sin ce in this case, $\hat{f}_n = O(1/n^2)$. If there is a step discontinuities, then there is Gibbs oscillations preventing uniform convergence, but the series still converges to the half of the left and right limit.

Poisson formula. The poisson formula connects the Fourier transform and the Fourier series to sampling and periodization operators. For some function $h(\omega)$ defined on $\mathbb{R}$ (typically the goal is to apply this to $h = \hat{f}$), its periodization reads

$$h_P(\omega) \stackrel{\text{def.}}{=} \sum_n h(\omega - 2\pi n). \tag{1.6}$$

This formula makes sense if $h \in L^1(\mathbb{R})$, and in this case $\|h_P\|_{L^1(\mathbb{T})} \leqslant \|h\|_{L^1(\mathbb{R})}$ (and there is equality for positive functions). The Poisson formula, stated in Proposition 1 below, corresponds to proving that the following diagram

$$\begin{array}{ccccc} & f(x) & \stackrel{\mathcal{F}}{\longrightarrow} & \hat{f}(\omega) & \\ \text{sampling} & \downarrow & & \downarrow & \text{periodization} \\ & (f(n))_n & \stackrel{\text{Fourier serie}}{\longrightarrow} & \sum_n f(n) e^{-\mathrm{i}\omega n} & \end{array}$$

is actually commutative.

Proposition 1 (Poisson formula)**.** *Assume that $\hat{f}$ has compact support and that $|f(x)| \leqslant C(1+|x|)^{-3}$ for some C. Then one has*

$$\forall\, \omega \in \mathbb{R}, \quad \sum_n f(n) e^{-\mathrm{i}\omega n} = \hat{f}_P(\omega). \tag{1.7}$$

Proof. Since $\hat{f}$ is compactly supported, $\hat{f}_P$ is well defined (it involves only a finite sum) and since f has fast decay, using (1.4), $(\hat{f})_P$ is C^1. It is thus the sum of its Fourier series

$$(\hat{f})_P(\omega) = \sum_k c_k e^{\mathrm{i}k\omega}, \tag{1.8}$$

where

$$c_k = \frac{1}{2\pi} \int_0^{2\pi} (\hat{f})_P(\omega) e^{-\mathrm{i}k\omega} \mathrm{d}\omega = \frac{1}{2\pi} \int_0^{2\pi} \sum_n \hat{f}(\omega - 2\pi n) e^{-\mathrm{i}k\omega} \mathrm{d}\omega.$$

One has

$$\int_0^{2\pi} \sum_n |\hat{f}(\omega - 2\pi n) e^{-\mathrm{i}k\omega}| \mathrm{d}\omega = \int_{\mathbb{R}} |\hat{f}|$$

which is bounded because $\hat{f} \in L^1(\mathbb{R})$ (it has a compact support and is C^1), so one can exchange the sum and integral

$$c_k = \sum_n \frac{1}{2\pi} \int_0^{2\pi} \hat{f}(\omega - 2\pi n) e^{-\mathrm{i}k\omega} \mathrm{d}\omega = \frac{1}{2\pi} \int_{\mathbb{R}} \hat{f}(\omega) e^{-\mathrm{i}k\omega} \mathrm{d}\omega = f(-k)$$

where we used the inverse Fourier transform formula (1.3), which is legit because $\hat{f} \in L^1(\mathbb{R})$. □

Shannon theorem. Shannon sampling theorem state a sufficient condition ensuring that the sampling operator $f \mapsto (f(ns))_n$ is invertible for some sampling step size $s > 0$. It require that $\mathrm{supp}(\hat{f}) \subset [-\pi/s, \pi/s]$, which, thanks to formula (1.3), implies that $\hat{f}$ is C^∞ (in fact it is even analytic). This theorem was first proved by Wittaker in 1915. It was re-proved and put in perspective in electrical engineering by Nyquist in 1928. It became famous after the paper of Shannon in 1949, which put forward its importance in numerical communications. Figure 1.4 give some insight on how the proof works (left) and more importantly, on what happens when the compact support hypothesis fails (in which case aliasing occurs, see also Figure 1.7).

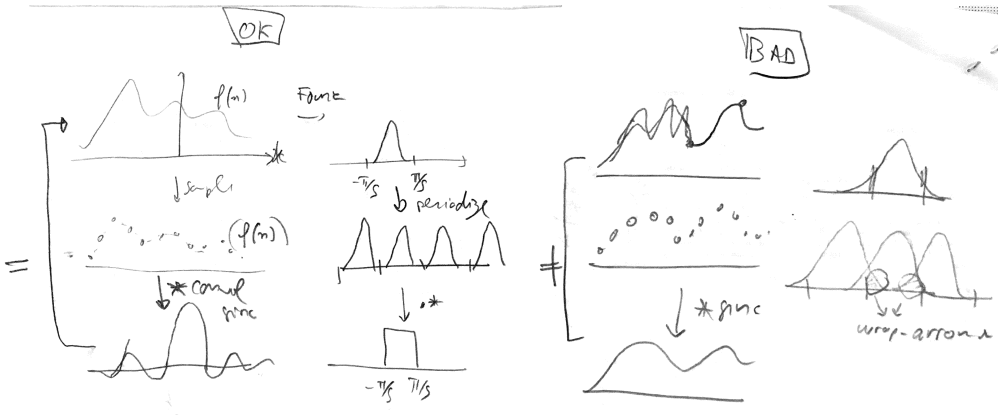

Figure 1.4: Schematic view for the proof of Theorem 1.

Theorem 1. *If* $|f(x)| \leqslant C(1+|x|)^{-3}$ *for some* C *and* $\mathrm{supp}(\hat{f}) \subset [-\pi/s, \pi/s]$*, then one has*

$$\forall\, x \in \mathbb{R}, \quad f(x) = \sum_n f(ns)\,\mathrm{sinc}(x/s - n) \quad \textit{where} \quad \mathrm{sinc}(u) = \frac{\sin(\pi u)}{\pi u} \tag{1.9}$$

with uniform convergence.

Proof. The change of variable $g \overset{\text{def.}}{=} f(s\cdot)$ results in $\hat{g} = 1/s\hat{f}(\cdot/s)$, indeed, denoting $z = sx$

$$\hat{g}(\omega) = \int f(sx)e^{-\mathrm{i}\omega x}\mathrm{d}x = \frac{1}{s}\int f(z)e^{-\mathrm{i}(\omega/s)z}\mathrm{d}z = \hat{f}(\omega/s)/s,$$

so that we can restrict our attention to $s = 1$. The compact support hypothesis implies $\hat{f}(\omega) = 1_{[-\pi,\pi]}(\omega)\hat{f}_P(\omega)$. Combining the inversion formula (1.3) with Poisson formula (1.8)

$$f(x) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \hat{f}_P(\omega)e^{\mathrm{i}\omega x}\mathrm{d}\omega = \frac{1}{2\pi}\int_{-\pi}^{\pi}\sum_n f(n)e^{\mathrm{i}\omega(x-n)}\mathrm{d}\omega.$$

Since f has fast decay, $\int_{-\pi}^{\pi}\sum_n |f(n)e^{\mathrm{i}\omega(x-n)}|\mathrm{d}\omega = \sum_n |f(n)| < +\infty$, so that one can exchange summation and integration and obtain

$$f(x) = \sum_n f(n)\frac{1}{2\pi}\int_{-\pi}^{\pi} e^{\mathrm{i}\omega(x-n)}\mathrm{d}\omega = \sum_n f(n)\,\mathrm{sinc}(x-n).$$

□

One issue with this reconstruction formula is that it uses a slowly decaying and very oscillating sinc kernels. In practice, one rarely uses such a kernel for interpolation, and one prefers smoother and more localized kernel. If $\mathrm{supp}(\hat{f}) \subset [-\pi/s', \pi/s']$ with $s' > s$ (i.e. have a more compact spectrum), one can re-do the proof of the theorem, and one gains some degree of freedom to design the reconstruction kernel, which now can be chosen smoother in Fourier and hence have exponential decay in time.

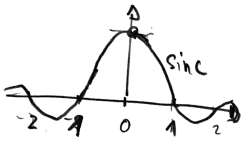

Figure 1.5: sinc kernel

Spline interpolation are defined by considering $\varphi_0 = 1_{[-1/2,1/2]}$ and $\varphi_k = \varphi_{k-1} \star \varphi_0$ which is a piecewise polynomial of degree k and has bounded derivative of order k (and is of class C^{k-1}) with compact support on $[-(k+1)/2, (k+1)/2]$.

The reconstruction formula reads $f \approx \tilde{f} \stackrel{\text{def.}}{=} \sum_n a_n \varphi(\cdot - n)$ where $(a_n)_n$ is computed from the $(f(n))_n$ by solving a linear system (associated to the interpolation property $\tilde{f}(n) = f(n)$). It is only in the cases $k \in \{0, 1\}$ (piecewise constant and affine interpolations) that one has $a_n = f(n)$. In practice, one typically use the cubic spline interpolation, which corresponds to $k = 3$.

`Associated code: test_sampling.m`

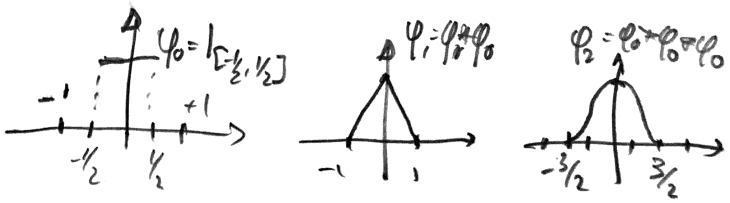

Figure 1.6: Cardinal splines as bases functions for interpolation.

This theorem also explains what happens if $\hat{f}$ is not supported in $[-\pi/s, \pi/s]$. This leads to aliasing, and high frequency outside this interval leads to low frequency artifacts often referred to as "aliasing". If the input signal is not bandlimitted, it is thus very important to pre-filter it (smooth it) before sampling to avoid this phenomena (of course this kills the high frequencies, which are lost), see Figure 1.7.

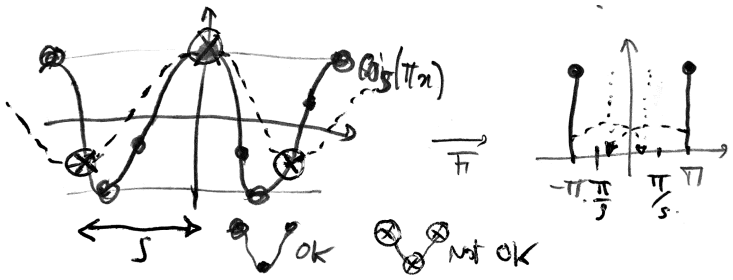

Figure 1.7: Aliasing in the simple case of a sine wave (beware however that this function does not have compact support).

Quantization. Once the signal have been sampled to obtain a discrete vector, in order to store it and transmit it, it is necessary to quantize the value to some finite precision. Section **??** presents transform coding, which is an efficient family of compression schemes which operate the quantization over some transformed domain (which correspond to applying a linear transform, usually orthogonal, to the sampled values). This is useful to enhance the performance of the source coding scheme. It is however common to operate directly the quantization over the sampled value.

Considering for instance a step size $s = 1/N$, one samples $(u_n \stackrel{\text{def.}}{=} f(n/N))_{n=1}^N \in \mathbb{R}^N$ to obtain a finite dimensional data vector of length N. Note that dealing with finite data corresponds to restricting the function f to some compact domain (here $[0, 1]$) and is contradictory with Shannon sampling theorem, since a function f cannot have a compact support in both space and frequency (so perfect reconstruction never holds when using finite storage).

Choosing a quantization step T, quantization $v_n = Q_T(u_n) \in \mathbb{Z}$ rounds to the nearest multiple of T, i.e.

$$v = Q_T(u) \quad \Leftrightarrow \quad v - \frac{1}{2} \leqslant u/T < v + \frac{1}{2},$$

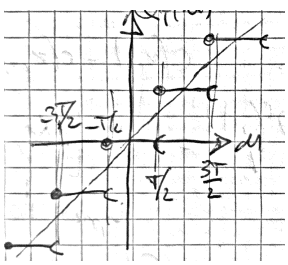

Figure 1.8:

see Fig. **??**. De-quantization is needed to restore a signal, and the best reconstruction (in average or in worse case) is defined by setting $D_T(v) \stackrel{\text{def.}}{=} Tv$. Quantizing and then de-quantizing introduce an error bounded by $T/2$, since $|D_T(Q_T(u)) - u| \leqslant T/2$. Up to machine precision, quantization is the only source of error (often called "lossy compression") in Shannon's standard pipeline.

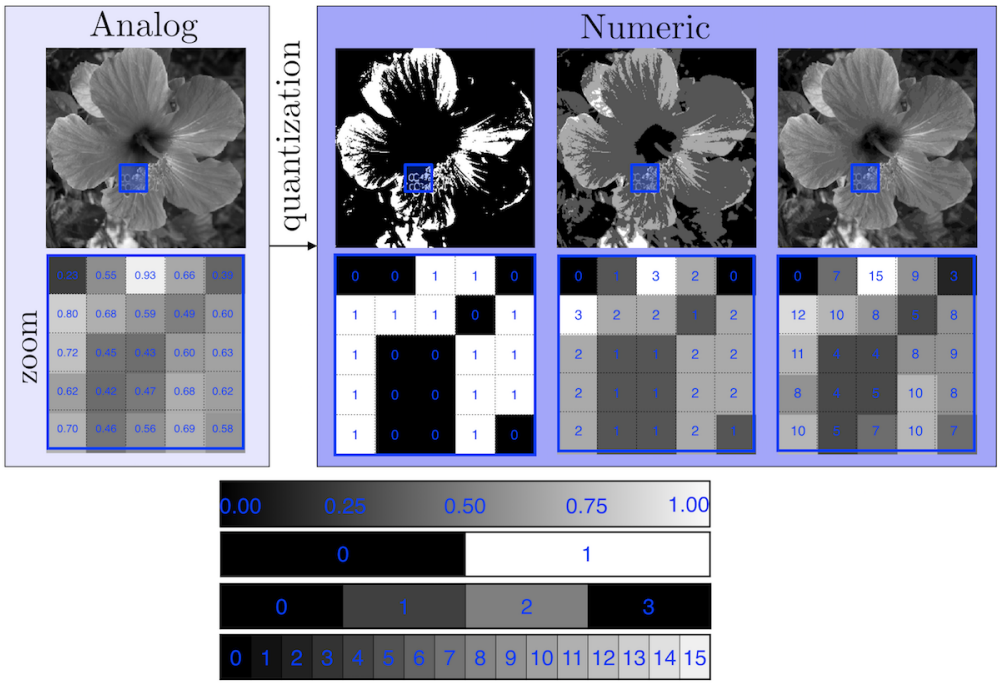

Figure 1.9: Quantizing an image using a decaying $T = 1/K$ where $K \in \{2, 3, 4, 16\}$ is the number of graylevels and the original image is normalized so that $0 \leqslant f_0 < 1$.

1.3 Shannon Source Coding Theorem

Uniform coding. We consider an alphabet $(s_1, \ldots, s_K)$ of K symbols. For instance, if one samples and quantize a bounded signal $0 \leqslant f_0 < 1$ using a step size $1/K$, then one can consider $s_k = k$ to be integer symbols. For text, these symbols include the letter plus extra punctuation symbols and blank. It is of course possible to code a sequence of such symbols using a uniform code (e.g. using the base 2 expansion) with $\lceil \log_2(K) \rceil$ bit per symbols. For instance if $K = 4$ and the symbols are $\{0, 1, 2, 3\}$, then the code words are $(c_0 = 00, c_1 = 01, c_2 = 10, c_3 = 11)$.

This uniform coding strategy is however extremely inefficient if the symbols are not uniformly distributed (i.e. if some symbols are more frequent than other, which is likely to be the case). We aim at designing better codes.

Prefix coding. A code $c_k = c(s_k)$ associate to each symbol s_k a code word $c_k \in \{0, 1\}^{\mathbb{N}}$ with a varying length $|c_k| \in \mathbb{N}^*$. A prefix code $c_k = c(s_k)$ is such that no word c_k is the beginning of another word c'_k. This is equivalent to be able to embed the $(c_k)_k$ as leaves of a binary tree T, with the code being output of a traversal from root to leaves (with a convention that going to a left (resp. right) child output a 0 (resp. a 1). We denote $c = \text{Leaves}(T)$ such prefix property.

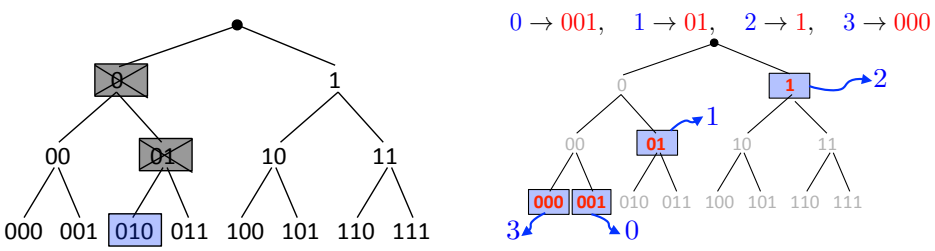

Figure 1.10: Left: complete tree of all codes of length 3; right: example of prefix code.

This tree-based representation is useful to decode a binary stream by simply performing tree traversal. One follows the tree, from top to bottom, and outputs a symbol each time a leaf is reached (and then re-start at the top).

Probabilistic modeling. We aim at designing the most possible compact code c_k. We assume at our disposal some probability distribution over this alphabet, which is just an histogram $p = (p_1, \dots, p_K) \in \mathbb{R}_+^K$ in the simplex, i.e. $\sum_k p_k = 1$. In practice, this probability is usually the empirical probability of appearance of the symbols x_k in the data to be coded.

The entropy of such an histogram is

$$H(p) \stackrel{\text{def.}}{=} -\sum_k p_k \log_2(p_k)$$

with the convention $0 \log_2(0) = 0$.

Denoting $h(u) = -u \log_2(u)$, $h'(u) \propto -\log(u) - 1$, $h''(u) \propto -1/u < 0$ so that H is strictly concave. The definition of the entropy extends to continuous density $f(x)$ for x on some measure space with reference measure $\mathrm{d}x$ (e.g. Lebesgue on $\mathbb{R}^d$) by setting $H(f) \stackrel{\text{def.}}{=} -\int f(x) \log(f(x)) \mathrm{d}x$.

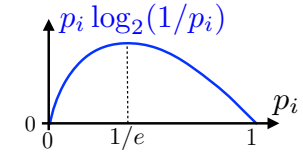

Figure 1.11:

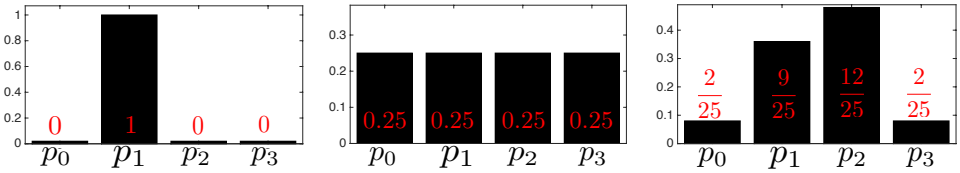

Figure 1.12: Three examples of probability distributions with corresponding entropies.

Lemma 1. *One has*

$$0 = H(\delta_i) \leqslant H(p) \leqslant H(1/K) = \log_2(K)$$

where δ_i is the Dirac histogram distribution at i.

Proof. First one notes that $-u \log_2(u) \geqslant 0$ for $u \in [0, 1]$ (see figure above), so that $H \geqslant 0$. Then we show the following inequality, for any histogram q

$$H(p) \leqslant -\sum_i p_i \log(q_i) \quad \Leftrightarrow \quad \sum_i p_i \log(q_i/p_i) \geqslant 0.$$

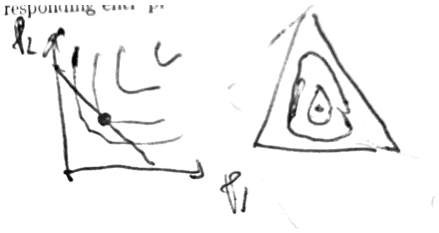

Figure 1.13: Linked extrema for the entropy.

This follows from $\log(u) \leqslant u-1$, since

$$\sum_i p_i \log(q_i/p_i) \leqslant \sum_i p_i(q_i/p_i - 1) = \sum_i q_i - \sum_i p_i = 0.$$

Applying this inequality to $q_i = 1/K$ gives

$$H(p) \leqslant -\sum_i p_i \log(1/K) = \log(1/K).$$

□

Shannon theorem. Assuming that $(p_k)_k$ is the empirical probability of appearance of the symbols x_k in the data to be coded, the average symbol length associated to some code c is

$$L(c) \stackrel{\text{def.}}{=} \sum_k p_k |c_k|.$$

The goal is to design the best possible c so that $L(c)$ is as small as possible. Shannon theorem of entropic coding, proved below, give both lower and upper bound for this question.

Theorem 2. *(i) If $c = \text{Leaves}(T)$ for some tree T, then*

$$L(c) \geqslant H(p).$$

(ii) Conversely, there exists a code c with $c = \text{Leaves}(T)$ such that

$$L(c) \leqslant H(p) + 1.$$

Before proving this theorem, we prove Kraft inequality, which describes the set of prefix codes using an inequality.

Lemma 2 (Kraft inequality)**.** *(i) For a code c, if there exists a tree T such that $c = \text{Leaves}(T)$ then*

$$\sum_k 2^{-|c_k|} \leqslant 1. \tag{1.10}$$

(ii) Conversely, if $(\ell_k)_k$ are such that

$$\sum_k 2^{-\ell_k} \leqslant 1 \tag{1.11}$$

then there exists a code $c = \text{Leaves}(T)$ such that $|c_k| = \ell_k$.

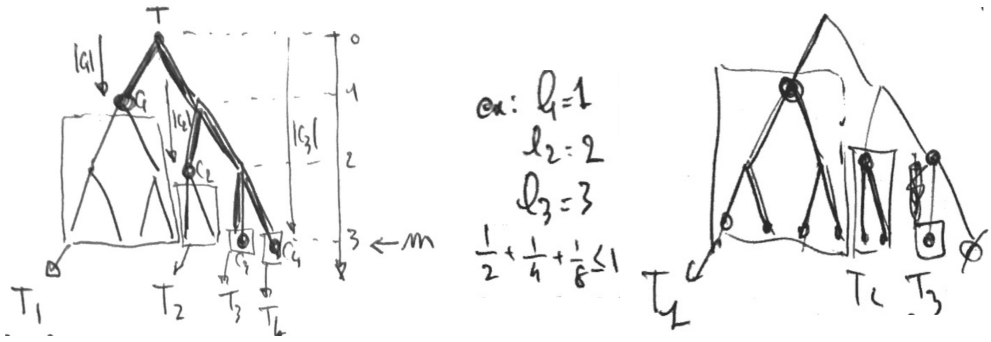

Figure 1.14: Left: full binary tree obtained by completing the tree associated to the code ($c_1 = 0, c_2 = 10, c_3 = 110, c_4 = 111$). Right: packing sub-trees associated to code length to form the left part of the full tree.

Proof. $\Rightarrow$ We suppose $c = \text{Leaves}(T)$. We denote $m = \max_k |c_k|$ and consider the full binary tree. Bellow each c_k, one has a sub-tree of height $m - |c_k|$, see Figure 1.14, left. This sub-tree has $2^{m-|c_k|}$ leaves. Since all these sub-trees do not overlap, the total number of leaf do not exceed the total number of leaves 2^m of the full binary tree, hence

$$\sum_k 2^{m-|c_k|} \leqslant 2^m,$$

hence (1.10).

$\Leftarrow$ Conversely, we assume (1.10) holds. Without loss of generality, we assume that $|c_1| \geqslant \ldots \geqslant |c_K|$. We start by putting a sub-tree of height $2^{m-|c_1|}$. Since the second tree is smaller, one can put it immediately aside, and continue this way, see Figure 1.14, right. Since $\sum_k 2^{m-|c_k|} \leqslant 2^m$, this ensure that we can stack side-by-side all these sub-tree, and this defines a proper sub-tree of the full binary tree. □

Shannon theorem. First, we consider the following optimization problem

$$\min_{\ell=(\ell_k)_k} \left\{ f(\ell) \stackrel{\text{def.}}{=} \sum_k \ell_k p_k \;;\; g(\ell) \stackrel{\text{def.}}{=} \sum_k 2^{-\ell_k} \leqslant 1 \right\}. \tag{1.12}$$

We fist show that at an optimal $\ell^\star$, the constraint is saturated, i.g. $g(\ell^\star) = 1$. Indeed, if $g(\ell^\star) = 2^{-u} < 1$, with $u > 0$, we define $\ell'_k \stackrel{\text{def.}}{=} \ell^\star_k - u$, which satisfies $g(\ell') = 1$ and also $f(\ell') = \sum_k (\ell_k - u)p_k < f(\ell^\star)$, which is a contradiction. So we can restrict in (1.12) the constraint to $g(\ell) = 1$ and apply the linked extrema theorem, which shows that necessarily, there exists $\lambda \in \mathbb{R}$ with $\nabla f(\ell^\star) = \nabla g(\ell^\star)$, i.e. $(p_k)_k = -\lambda \ln(2)(2^{-\ell^\star_k})_k$. Since

$$1 = \sum_k p_k = -\lambda \ln(2) \sum_k 2^{-\ell_k} = -\lambda \ln(2)$$

we deduce that $\ell^\star_k = -\log(p_k)$.

(i) If $c = \text{Leave}(T)$, the by Kraft inequality (1.10), necessarily $\ell_k = |c_k|$ satisfy the constraints of (1.12), and thus $H(p) = f(\ell^\star) \leqslant f(\ell) = L(\ell)$.

(ii) We define $\ell_k \stackrel{\text{def.}}{=} \lceil -\log_2(p_k) \rceil \in \mathbb{N}^*$. Then $\sum_k 2^{-\ell_k} \leqslant \sum_k 2^{\log_2(p_k)} = 1$, so that these lengths satisfy (1.11). Thanks to Proposition 2 (ii), there thus exists a prefix code c with $|c_k| = \lceil -\log_2(p_k) \rceil$. Furthermore

$$L(c) = \sum_k p_k \lceil -\log_2(p_k) \rceil \leqslant \sum_k p_k (-\log_2(p_k) + 1) = H(p) + 1.$$

□

Note that this proof is constructing, i.e. it gives an algorithm that construct an almost optimal c, and this code is often called the Shannon-Fano code. It is usually a good code, although it is not necessarily the optimal code with the smallest $L(c)$. Such an optimal code can easily be computed in almost linear time (only sorting of the probability is needed, so it is $K(\log(K))$) by Huffman's dynamic programming algorithm (invented in 1952). The proof of correctness of this algorithm is however a bit tedious. Figure 1.15 shows an example of application of this algorithm.

`Associated code: coding/test_text.m`

In practice, such an entropic coding, although optimal, is not very efficient when one of the symbol has a large probability p_k. This is because then $2^{-p_k} \ll 1$ but one cannot allocate a fractional number of bit. This is why $L(c)$ can be as large as $H(p)+1$. A simple workaround is to artificially increase the size of the alphabet from K to K^r by grouping together sets of r consecutive symbols, and thus reducing the gap to $H(p)+1/r$. Constructing the code and coding however becomes very slow for large r. The standard way to achieve this without explicitly doing the grouping is by using arithmetic coding, invented in 1976, which using interval arithmetic to allocate fractional number of bits and leveraging the fact that one usually code large sequence, thus approaching to arbitrary precision Shannon bound $H(p)$ as the length of the data increases.

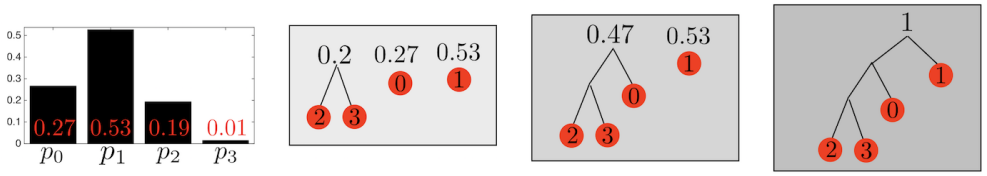

Figure 1.15: Huffman coding algorithm in action.

Note that while we give some statistical and probabilistic interpretation of entropy (measure of uncertainty) and of Shannon theorem, this theory is fully deterministic and give a bound for the actual length $NL(c)$ of coding some sequence of length N if the probability p are the empirical probability of the sequence.

If one choose a different probability q and use it to code the sequence, one necessarily obtain a worse average coding length, and this is reflected by the positivity of the so-called relative entropy (beware that it is a convex function while the entropy is concave), which is often called the Kulback-Leibler divergence

$$\mathrm{KL}(p|q) = -\sum_k p_k \log q_k - H(p) = \sum_k p_k \log \frac{p_k}{q_k} \geqslant 0.$$

This KL divergence is similar to a distance in the sense that $\mathrm{KL}(p|q)=0$ if and only if $p=q$ (note however that KL is not symmetric and does not satisfies the triangular inequality). It also has the remarkable property that it is jointly convex in (p,q). It is of paramount importance to compare probability distributions and measures, and form the basis of the fields of information theory and information geometry.

Doing better. One can wonder if it is possible to go below the entropy bound. By the virtue of Shannon theorem, it is not possible if only can only code in sequential order the symbols themselves. From a statistical perspective, it is as if the symbols where considered to be independent. If there is some redundancy in the sequence of symbol (for instance if they are discretization of a smooth function, so that to consecutive symbols are likely to be equal), it is possible to re-transform (in a bijective way) the sequence to make them "more independent". A simple illustration of this idea is given in Figure 1.16, where one computes successive difference of a 1D sequence of symbols (beware to also retain the initial symbol to be able to do the decoding.

Another, more systematic way to leverage such temporal redundancy is by performing run-length-coding, which operate by grouping together sequence of similar symbols and thus coding first a symbol and then the length of the associated group (which is coded entropically). If the sequence is generated by a Markov chain, this method can be shown to asymptotically reach the Shannon bound where now the entropy is the

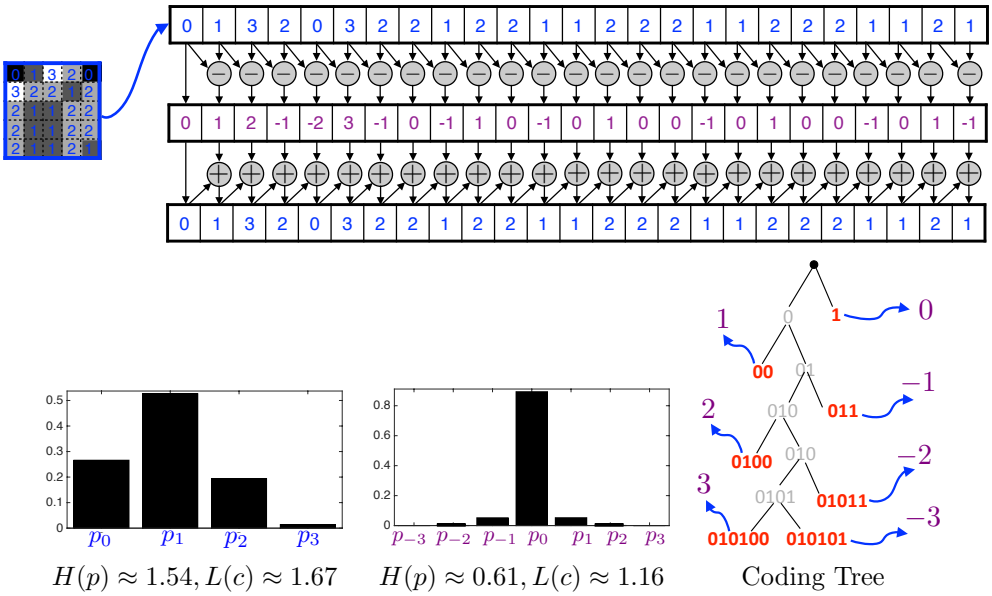

Figure 1.16: Top: retransformation by successive differences. Bottom: comparison of histograms of pixel values and differences, and a code tree for these differences.

entropy associated to the distribution of the Markov chain on infinite sequences (which can be computed as the limit of the entropy for finite sequences).

Bibliography

[1] Stephane Mallat. *A wavelet tour of signal processing: the sparse way.* Academic press, 2008.