Universidad de Murcia
Facultad de Matemáticas

Trabajo fin de grado

La transformada de Fourier discreta y el formato JPEG

António Ricardo Cabuzo

Supervisado por
Gustavo Garrigós

Enero 2018

Declaración de originalidad

António Ricardo Cabuzo, estudiante del Grado en Matemáticas de la Universidad de Murcia, declara que el presente trabajo de fin de grado, tutorizado por el profesor Gustavo Garrigós, es original y que todas las fuentes utilizadas para su realización han sido debidamente citadas en el mismo.

Firmado ..... Fecha .....

Índice general

Resumen

El objetivo de este trabajo es introducir la teoría matemática relacionada con la *Transformada de Fourier Discreta*, así como algunas de sus aplicaciones más relevantes. Entre ellas pretendemos dar una descripción detallada del formato JPEG que habitualmente se utiliza en la compresión de imágenes.

A principios del siglo XIX, J. B. Fourier postuló que toda función $f(x)$ definida en un intervalo, digamos $(-\pi, \pi)$, puede escribirse como una suma infinita de funciones trigonométricas

$$f(x) = \sum_{n=0}^{\infty} a_n \cos(nx) + b_n \operatorname{sen}(nx), \quad -\pi < x < \pi,$$

para ciertos coeficientes a_n, b_n que dependen de f. En la terminología moderna, usando notación compleja, se tiene

$$f(x) = \frac{1}{2\pi} \sum_{n\in\mathbb{Z}} \hat{f}(n) e^{inx}, \quad x \in (-\pi, \pi), \tag{0.0.1}$$

donde los coeficientes $\hat{f}(n)$ vienen dados explícitamente por

$$\hat{f}(n) = \int_{-\pi}^{\pi} f(t) e^{-int}\, dt, \quad n \in \mathbb{Z}. \tag{0.0.2}$$

De forma similar, toda función $f(x)$ definida en la recta real $x \in \mathbb{R}$ se debe poder representar como una "integral trigonométrica"

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{f}(\omega)\, e^{i\omega x}\, d\omega, \quad x \in \mathbb{R}, \tag{0.0.3}$$

a partir de una función adecuada de "coeficientes" $\hat{f}(\omega)$, dada por

$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t}\, dt, \quad \omega \in \mathbb{R}. \tag{0.0.4}$$

A esta última expresión $\hat{f}(\omega)$ se le denomina *transformada de Fourier* de f.

La transformada de Fourier ha resultado ser una herramienta extremadamente útil para analizar y extraer propiedades de una función f, tanto desde el punto de vista del Análisis Matemático, como de muchas aplicaciones de la Física y la Ingeniería. En el terreno aplicado es habitual que la función $f(x)$ modelice a una "señal analógica" que contiene información sobre un determinado fenómeno físico: intensidad de corriente eléctrica, densidad electrónica de una molécula, posición de una cuerda o membrana vibrante, intensidad luminosa de una imagen, etc... En

esos casos, $\hat{f}(\omega)$ nos da una representación alternativa de la señal f que a menudo contiene información relevante sobre las frecuencias de ésta.

A partir de 1950 las señales analógicas, tanto auditivas como visuales, han sido sustituidas por señales digitales sobre las que es más fácil realizar manipulaciones numéricas, y además los cálculos pueden realizarse de manera muy rápida. Este ha sido el caso en aplicaciones a telefonía digital, cámaras digitales de fotos, de televisión, etc... Tanto para una señal de audio como para una imagen, la forma más extendida de obtener una señal digital es mediante un *muestreo*. Es decir, en lugar de trabajar con una función continua $f(x)$, trabajamos con la "señal discreta" $\{f(nT)\}_{1\leq n\leq N}$, para un cierto paso T, y un número prefijado de datos N.

Se hace pues necesaria una noción de transformada de Fourier de un conjunto discreto de datos $\mathbf{x} = (x_1, \ldots, x_N) \in \mathbb{C}^N$, que definimos a continuación

$$\widehat{\mathbf{x}}[k] = \sum_{n=1}^{N} x_n e^{-\frac{2\pi i k n}{N}}, \quad k = 1, \ldots, N. \tag{0.0.5}$$

A menudo denotaremos al vector $(\widehat{\mathbf{x}}[1], \ldots, \widehat{\mathbf{x}}[N])$ por $DFT(\mathbf{x})$, que denominaremos *Transformada de Fourier Discreta* (TFD) de $\mathbf{x}$.

El objetivo de este trabajo es describir las propiedades principales de la TFD, y desarrollar específicamente aquéllas que tienen relación con el algoritmo JPEG de tratamiento de imágenes digitales. Para ello estructuramos los diferentes capítulos de este trabajo como sigue.

En el capítulo 1 introducimos las propiedades principales de la transformada de Fourier continua, así como las herramientas matemáticas que usaremos en este trabajo. Entre ellos el concepto de convolución y el de aproximación de la identidad. Ambos juegan un papel importante a la hora de probar los dos principales resultados de este tema: el Teorema de Inversión de la Transformada de Fourier, y el Teorema de Plancherel. Concluimos con el Teorema de Muestreo de Shannon, un resultado clásico que justifica el paso de señales analógicas a señales digitales.

El capítulo 2 lo dedicamos al estudio de señales discretas. En particular los conceptos de convolución circular y Transformada de Fourier Discreta, así como los análogos de los resultados del capítulo 1 en el caso finito dimensional. El teorema más relevante es el algoritmo FFT, para el cálculo de la Transformada de Fourier Rápida, así como su aplicación a las convoluciones rápidas.

En el capítulo 3 consideramos las bases de cosenos para representar funciones en un intervalo, y sus análogos discretos. Estas bases, aunque son sólo pequeñas variantes de la base de Fourier, juegan un papel importante en las aplicaciones, pues permiten eliminar los efectos de las discontinuidades en los bordes del intervalo. Esta característica es importante en el tratamiento de imágenes, y debido a ello las bases coseno se usarán posteriormente en la descripción del algoritmo JPEG. Los resultados principales son pequeñas variaciones de los obtenidos en el capítulo 2.

En el capítulo 4 introducimos algunos conceptos algebraicos sencillos de Teoría de Códigos que se utilizan en el algoritmo JPEG. En particular, se prueba con detalle el Teorema de Entropía de Shannon, que muestra cómo una codificación adecuada de los datos puede reducir considerablemente el tamaño de un fichero. Además, construimos con detalle el Código de Huffman, y demostramos que minimiza la entropía de Shannon.

Por último, en el capítulo 5, utilizamos las herramientas de los capítulos 3 y 4 para describir con detalle los pasos que constituyen el algoritmo JPEG. Este algoritmo permite analizar y comprimir imágenes digitales, reduciendo considerablemente el tamaño de los ficheros sin apenas perder calidad visual. Ilustramos los pasos con ejemplos explícitos, y mostramos algunas imágenes antes y después del proceso de compresión.

Abstract

The aim of this work is to introduce the mathematical theory related to the *Discrete Fourier Transform*, as well as some of its most relevant applications. Among them we intend to give a detailed description of the JPEG format that is usually used in the compression of images.

At the beginning of the 19th century, J. B. Fourier postulated that every function $f(x)$ defined in a range, say $(-\pi, \pi)$, can be written as an infinite sum of trigonometric functions

$$f(x) = \sum_{n=0}^{\infty} a_n \cos(nx) + b_n \operatorname{sen}(nx), \quad -\pi < x < \pi,$$

for certain coefficients a_n, b_n that depend on f. In modern terminology, using complex notation, we have

$$f(x) = \frac{1}{2\pi} \sum_{n \in \mathbb{Z}} \hat{f}(n) e^{inx}, \quad x \in (-\pi, \pi), \tag{0.0.6}$$

where the coefficients $\hat{f}(n)$ are given explicitly by

$$\hat{f}(n) = \int_{-\pi}^{\pi} f(t) e^{-int}\, dt, \ n \in \mathbb{Z}. \tag{0.0.7}$$

Similarly, every function $f(x)$ defined on the real line $x \in \mathbb{R}$ may be represented as a "trigonometric integral"

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{f}(\omega) e^{i\omega x}\, d\omega, \quad x \in \mathbb{R}, \tag{0.0.8}$$

from an adequate function of "coefficients" $\hat{f}(\omega)$, given by

$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t}\, dt, \quad \omega \in \mathbb{R}. \tag{0.0.9}$$

This last expression $\hat{f}(\omega)$ is called the *Fourier transform* of f.

The Fourier transform has turned out to be an extremely useful tool to analyze and extract properties of a function f, both from the Mathematical Analysis point of view, as from many applications of Physics and Engineering. In the applied terrain it is usual for the function $f(x)$ to model an "analog signal" that contains information about a certain physical phenomenon: electric current intensity, electron density of a molecule, position of a string or vibrant membrane, luminous intensity of an image, etc ... In those cases, $\hat{f}(\omega)$ gives us an alternative representation of the signal f that often contains relevant information about its frequencies.

Since 1950 analog signals, both auditory and visual, have been replaced by digital signals on which it is easier to perform numerical manipulations, and also the calculations can be done very quickly. This has been the case in digital telephony applications, digital photo cameras, television, etc ... Both for an audio signal and for an image, the most widespread way to obtain a digital signal is by *sampling* That is, instead of working with a continuous function $f(x)$, we work with the "discrete signal" $\{f(nT)\}_{1\leq n\leq N}$, for a a certain step T, and a prefixed number of data N.

A Fourier transform notion of a discrete set of data is therefore necessary. If $\mathbf{x} = (x_1, \ldots, x_N) \in \mathbb{C}^N$, we define

$$\widehat{\mathbf{x}}[k] = \sum_{n=1}^{N} x_n e^{-\frac{2\pi i k n}{N}}, \quad k = 1, \ldots, N. \tag{0.0.10}$$

We will often denote the vector $(\widehat{\mathbf{x}}[1], \ldots, \widehat{\mathbf{x}}[N])$ by $DFT(\mathbf{x})$, which we will call *Discrete Fourier Transform* (TFD) of $\mathbf{x}$.

The objective of this work is to describe the main properties of DFT, and to develop specifically those related to the JPEG algorithm of digital image processing. For this we structure the different chapters of this work as follows.

In chapter 1 we introduce the main properties of the continuous Fourier transform, as well as the mathematical tools that we will use in this work. Among them the concept of convolution and approximation of the identity. Both play an important role when it comes to testing the two main results of this topic: the Fourier Transform Inversion Theorem and the Plancherel Theorem. We conclude with the Shannon Sampling Theorem, a classical result that justifies the passage of analog signals to digital signals.

Chapter 2 is dedicated to the study of discrete signals. In particular, the concepts of circular convolution and Discrete Fourier Transform, as well as the analogs of the results of chapter 1 in the finite dimensional case. The most relevant theorem is the FFT algorithm, for the calculation of the Fast Fourier Transform, as well as its application to fast convolutions.

In chapter 3 we consider bases of cosines to represent functions in an interval, and their discrete analogs. These bases, although are only small variants of the Fourier basis, play an important role in the applications, because they allow to eliminate the effects of the discontinuities in the edges of the interval. This feature is important in the treatment of images, and because of this the cosine bases will be used later in the description of the JPEG algorithm. The main results are small variations from those obtained in chapter 2.

In Chapter 4 we introduce some simple algebraic concepts from Code Theory that are used in the JPEG algorithm. In particular, Shannon's Entropy Theorem is proved in detail, showing how proper coding of data can greatly reduce the size of

a file. In addition, we carefully build the Huffman Code and show that it minimizes Shannon's entropy.

Finally, in Chapter 5, we use the tools in Chapters 3 and 4 to describe in detail the steps that make up the JPEG algorithm. This algorithm allows analyzing and compressing digital images, considerably reducing the size of the files without hardly losing visual quality. We illustrate the steps with explicit examples, and show some images before and after the compression process.

Capítulo 1

La transformada de Fourier continua

1. Convolución y Aproximaciones de la identidad

El concepto de convolución aparece en varios lugares de este trabajo en relación con el procesamiento de señales. Además es una importante herramienta del Análisis Matemático, que necesitaremos en las demostraciones de algunos teoremas sobre la transformada de Fourier. Por ello recordamos en esta sección su definición y las propiedades principales, y referimos al libro [**2**, §8.2] para más detalles.

1.1. Convolución y propriedades.

Definición 1.1.1. *Sean $f, g : \mathbb{R} \longrightarrow \mathbb{C}$ funciones medibles Lebesgue. Definimos la convolución de f y g como:*

$$f * g(x) := \int_{\mathbb{R}} f(x-y)g(y)dy, \quad x \in \mathbb{R}$$

siempre que la integral sea absolutamente convergente (es decir, $\int_{\mathbb{R}} |f(x-y)g(y)|dy < \infty$).

Veamos ahora algunas propiedades de las convoluviones que usaremos a lo largo de este trabajo:

Proposición 1.1.2. *Sean f, g, h tales que todas las convoluciones que siguen están bien definidas en el punto t. Entonces*

1. $f * (g + h)(t) = f * g(t) + f * h(t)$.
2. $(cf) * g(t) = c(f * g)(t) = f * (cg)(t)$, *para cualquier* $c \in \mathbb{C}$.
3. $(f * g)(t) = (g * f)(t)$.
4. $(f * g) * h(t) = f * (g * h)(t)$.

Demostración. Tenemos:

1. Esta propiedad es trivial se sigue de la linealidad de la integral.
2. Este caso tambien se sigue de la linealidad de la integral.
3. Aplicando la definición de convoluciones viene:

$$(f * g)(t) = \int_{\mathbb{R}} f(t-y)g(y)dy,$$

haciendo el cambio de variable $t - y = u, dy = du$ en la integral anterior resulta:

$$(f * g)(t) = \int_{\mathbb{R}} f(u)g(t-u)du = (g * f)(t).$$

4. De la definición resulta:

$$[(f*g)*h](t) = \int_{\mathbb{R}} (f*g)(t-y)h(y)dy,$$

sabemos que $(f*g)(u) = \int_{\mathbb{R}} f(u-z)g(z)dz$ por eso se sigue:

$$[(f*g)*h](t) = \int_{\mathbb{R}} \left[\int_{\mathbb{R}} f(t-y-z)g(z)dz \right] h(y)dy.$$

Haciendo el cambio de variables $z = u-y, du = dz$ resulta:

$$\begin{aligned}
[(f*g)*h](t) &= \int_{\mathbb{R}} \left[\int_{\mathbb{R}} f(t-u)g(u-y)h(y)du \right] dy \\
&= \int_{\mathbb{R}} f(t-u) \left(\int_{\mathbb{R}} g(u-y)h(y)dy \right) du \\
&= \int_{\mathbb{R}} f(t-u)(g*h)(u)du \\
&= [f*(g*h)](t).
\end{aligned}$$

En la integral anterior hemos usado el teorema de Fubini ya que si suponemos el primer término definido se tiene $|f(t-y-z)g(z)h(y)| \in L^1(\mathbb{R}^3)$.

□

Proposición 1.1.3 (Desigualdad de Young). *Si $f \in L^1, g \in L^p, 1 \leq p \leq \infty$, entonces:*

$$\|f*g\|_p \leq \|f\|_1 \|g\|_p.$$

Por tanto $f*g \in L^p$.

Demostración.

$$\|f*g\|_p \leq \| \int |f(t-y)g(y)|dy\|_{L^p(dt)} \leq \int \|f(t-y)\|_{L^p(dt)} |g(y)|dy = \|f\|_p \|g\|_1$$

donde hemos usado en el segundo paso la desigualdad integral de Minkowski. □

Proposición 1.1.4. *Sean $1 \leq p, q \leq \infty$ tales que $\frac{1}{p} + \frac{1}{q} = 1$. Sea $f \in L^p(\mathbb{R})$ y $g \in L^q \cap UC(\mathbb{R})$, la clase de funciones uniformemente continuas en $\mathbb{R}$. Entonces:*

1. $f*g \in UC(\mathbb{R})$
2. $\|f*g\|_\infty \leq \|f\|_p \|g\|_q$
3. Si $1 < p < \infty$ entonces $\lim_{|x|\to\infty} f*g(x) = 0$

Para la demostración de esta proposición haremos uso del siguiente lema:

Lema 1.1.5. *Si $1 \leq p < \infty$ y $f \in L^p(\mathbb{R})$, entonces*

$$\lim_{|h|\to 0} \|f(\cdot+h) - f(\cdot)\|_p = 0.$$

Demostración. Usaremos la densidad de $C_c^\infty(\mathbb{R})$.
Dado $\varepsilon > 0$, existe $\Phi \in C_c^\infty$ tal que $\|f-\Phi\|_p < \varepsilon$

$$\begin{aligned}
\|f(\cdot+h) - f(\cdot)\|_p &= \|f(\cdot+h) - \Phi(\cdot+h) + \Phi(\cdot+h) - \Phi + \Phi - f\|_p \\
&\leq \|f(\cdot+h) - \Phi(\cdot+h)\|_p + \|\Phi(\cdot+h) - \Phi\|_p + \|\Phi - f\|_p \\
&= 2\|\Phi - f\|_p + \|\Phi(\cdot+h) - \Phi\|_p
\end{aligned}$$

El primer sumando del último miembro sabemos que es $< \varepsilon$, nos queda ver que el segundo sumando es menor que ε, si $|h|$ es suficientemente pequeño.

Como $Sop(\Phi) \subseteq (-R, R)$ para R suficientemente grande, se tiene que

$$Sop(\Phi(\cdot + h) - \Phi(\cdot)) \subseteq (-2R, 2R)$$

si $|h| \leq R$. Como $\Phi \in UC(\mathbb{R})$, existe $\delta_0 = \delta_0(\Phi, R, \varepsilon)$ tal que si $|h| \leq \delta_0$ y $t \in \mathbb{R}$ entonces $|\Phi(t + h) - \Phi(t)| \leq \varepsilon/(4R)^{1/p}$. Por tanto, si $|h| \leq \delta_0$,

$$\|\Phi(\cdot + h) - \Phi(\cdot)\|_p = \left(\int_{-2R}^{2R} |\Phi(t + h) - \Phi(t)|^p dt \right)^{\frac{1}{p}} \leq \left(\int_{-2R}^{2R} \frac{\varepsilon^p}{4R} dt \right)^{\frac{1}{p}} = \varepsilon.$$

Por tanto,

$$\|f(\cdot + h) - f(\cdot)\|_p < 3\varepsilon, \quad si \quad |h| \leq \delta_0.$$

□

Estamos ahora en condiciones de probar la proposición anterior.

Demostración. ahora demostremos por orden de enumeración

1. podemos suponer $1 \leq p < \infty$. Usando la desigualdad de Holder

$$\begin{aligned} |f * g(t + h) - f * g(t)| &= \left| \int (f(t + h - y) - f(t - y))\, g(y) dy \right| \\ &\leq \|f(t + h - \cdot) - f(t - \cdot)\|_p \|g\|_q = (1) \end{aligned}$$

   Haciendo el cambio de variable $u = t - y$ y usando el lema anterior vemos que:

$$(1) \leq \|f(u + h) - f(u)\|_{L^p(du)} \|g\|_q \overset{|h| \to 0}{\longrightarrow} 0$$

   independientemente de t. Por tanto, $f * g$ es uniformemente continua.

2. Aplicando la desigualdad de Holder

$$\begin{aligned} |f * g(t)| &= \left| \int f(t - y) g(y) dy \right| \\ &\leq \left(\int |f(t - y)|^p \, dy \right)^{\frac{1}{p}} \left(\int |g(y)|^q dy \right)^{\frac{1}{q}} = \|f\|_p \|g\|_q. \end{aligned}$$

3. Sea $1 \leq p < \infty$. Fijamos $\varepsilon > 0$.

Tomamos $F, G \in C_c(\mathbb{R})$ tales que $\|f - F\|_p < \varepsilon/\|g\|_q$, $\|g - G\|_q < \varepsilon/\|f\|_p$ y $\|G\|_q \leq 2\|g\|_q$, lo cual implica que $F * G$ tiene soporte compacto en $(-R, R)$.

Vamos a ver que si $|t| \geq R$. Entonces

$$\begin{aligned} |(f * g)(t)| &\leq |f * (g - G)(t)| + |(f - F) * G(t)| \\ &\leq \|f\|_p \|g - G\|_q + \|f - F\|_p \|G\|_q \\ &\leq \varepsilon + \varepsilon \frac{\|G\|_q}{\|g\|_q} \leq \varepsilon + 2\varepsilon = 3\varepsilon. \end{aligned}$$

□

1.2. Aproximaciones de la Identidad.

Definición 1.1.6. *Decimos que* $\{\Phi_\varepsilon\}_{\varepsilon>0} \subseteq L^1(\mathbb{R})$ *es una aproximación de la identidad regular (AIR) cuando* $\varepsilon \to 0$ *si*

- $\int \Phi_\varepsilon = 1$
- $\sup_{\varepsilon>0} \int |\Phi_\varepsilon| < \infty$
- $\forall \delta > 0$, $\lim_{\varepsilon\to 0} \int_{|t|\geq\delta} |\Phi_\varepsilon| = 0$

Lema 1.1.7. *Si* $\Phi \in L^1(\mathbb{R})$ *y* $\int \Phi = 1$, *entonces* $\{\Phi_\varepsilon(t) = \frac{1}{\varepsilon}\Phi(\frac{t}{\varepsilon})\}_{\varepsilon>0}$ *es una aproximación de la identidad regular.*

Demostración. Haciendo el de variable $\frac{t}{\varepsilon} = u$ tenemos lo seguiente:

$$\int \frac{1}{\varepsilon}\Phi\left(\frac{t}{\varepsilon}\right) dt = \int \Phi(u)\, du = 1 \quad \forall\varepsilon,$$

por lo que verifica la primera y la segunda condición de la definición de aproximación de la identidad regular. Veamos la tercera condición. Haciendo de nuevo el cambio de variable $\frac{t}{\varepsilon} = u$, se tiene

$$\int_{|t|\geq\delta} |\Phi_\varepsilon(t)|\, dt = \int_{|u|\geq\frac{\delta}{\varepsilon}} |\Phi(u)|\, du$$

por lo que como $\frac{\delta}{\varepsilon} \longrightarrow \infty$ cuando $\varepsilon \longrightarrow 0^+$, por el Teorema de la Convergencia Dominada tenemos:

$$\int_{|t|\geq\frac{\delta}{\varepsilon}} |\Phi(u)|\, du \longrightarrow 0 \quad \text{cuando} \quad \varepsilon \longrightarrow 0^+$$

□

Pasamos ahora a estudiar algunos teoremas sobre convergencia de aproximación de la identidad regular.

Teorema 1.1.8 (Covergencia en norma L^p de AIR). *Si* $\{\Phi_\varepsilon\}_{\varepsilon>0}$*es aproximación a la identidad regular, entonces:*

a) Si $1 \leq p < \infty$ *y* $f \in L^p(\mathbb{R})$ *entoces* $\Phi_\varepsilon * f \xrightarrow{\varepsilon\to 0} f \quad en \quad L^p(\mathbb{R})$.

b) Si $f \in UC_{bded}(\mathbb{R})$ *(uniformemente continua y acotada) entonces se tiene* $\lim_{\varepsilon\to 0} \Phi_\varepsilon * f(x) = f(x)$, *uniformemente en todo* $x \in \mathbb{R}$.

Demostración. a)

$$\begin{aligned}\Phi_\varepsilon * f(t) - f(t) &= \int f(t-y)\Phi_\varepsilon(y)dy - f(t)\int \Phi_\varepsilon(y)dy \\ &= \int (f(t-y) - f(t))\, \Phi_t(y)dy\end{aligned}$$

donde la primera igualdad se da ya que Φ_ε es AIR. Por tanto se tiene:

$$\|\Phi_\varepsilon * f - f\|_{L^p(dt)} = \left\| \int (f(t-y) - f(t))\, \Phi_\varepsilon(y)dy \right\|_{L^p(dt)}$$

y aplicando la Desigualdad Integral de Minkowski resulta

$$\|\Phi_\varepsilon * f - f\|_{L^p(dt)} \leq \int \|f(t-y) - f(t)\|_{L^p(dt)} |\Phi_\varepsilon(y)| dy$$

$$\leq \int_{|y|\leq\delta} \|f(t-y)-f(t)\|_{L^p(dt)}|\Phi_\varepsilon(y)|dy + \int_{|y|\geq\delta} \|f(t-y)-f(t)\|_{L^p(dt)}|\Phi_\varepsilon(y)|dy$$
$$\leq \int_{|y|\leq\delta} \|f(t-y)-f(t)\|_{L^p(dt)}|\Phi_\varepsilon(y)|dy + \int_{|y|\geq\delta} 2\|f\|_p|\Phi_\varepsilon(y)|dy = (1)$$

Dado $\eta > 0$, sabemos que existe $\delta > 0$ tal que $\|f(\cdot + y) - f\|_p < \eta$ si $|y| < \delta$ (por el Lema 1.1.5). Por tanto:

$$(1) \leq \int_{|y|\leq\delta} \eta\,|\Phi_\varepsilon(y)|dy + \int_{|y|\geq\delta} \|2f\|_p|\Phi_\varepsilon(y)|dy$$
$$\leq \eta \sup_{\varepsilon>0}\|\Phi_\varepsilon\|_1 + 2\|f\|_p \int_{|y|\geq\delta}|\Phi_\varepsilon(y)|dy = (2)$$

donde $\sup_{\varepsilon>0}\|\Phi_\varepsilon\|_1$ es finito por la segunda condición de la difinición de AIR. Por la tercera condición de la misma definición, $\exists\varepsilon_0 = \varepsilon_0(\eta,\delta)$ tal que $\int_{|y|\geq\delta}|\Phi_\varepsilon| \leq \eta, \quad \forall\varepsilon < \varepsilon_0$, con lo cual

$$(2) \leq \eta\Big(\sup_{\varepsilon>0}\|\Phi_\varepsilon\|_1 + 2\|f\|_p\Big), \quad \text{si } \varepsilon < \varepsilon_0,$$

y por tanto , $\lim_{\varepsilon\to 0}\|\Phi_\varepsilon * f - f\|_p = 0$.

b) Aqui el resultado es análogo usando $\|\cdot\|_\infty$ en lugar de norma L^p, y usando la continuidad uniforme de f en lugar del Lema 1.1.5. □

Usando una prueba parecida se extrae el siguiente resultado, que garantiza la convergencia en un punto t_0:

PROPOSICIÓN 1.1.9. *Sea $\{\Phi_\varepsilon\}_{\varepsilon>0}$ una AIR. Si $f \in L^\infty(\mathbb{R})$ y f es continua en un punto $t_0 \in \mathbb{R}$, entonces existe*

$$\lim_{\varepsilon\to 0} \Phi_\varepsilon * f(t_0) = f(t_0).$$

Con condiciones ligeramente más fuertes en la AIR, es posible demostrar también la convergencia puntual en casi todo punto.

TEOREMA 1.1.10 (Convergencia ctp de AIR). *Sea $\{\Phi_\varepsilon(t) = \frac{1}{\varepsilon}\Phi(t/\varepsilon)\}_{\varepsilon>0}$ donde $\Phi \in L^1(\mathbb{R})$ cumple $\int\Phi = 1$ y, para algún $\gamma > 0$,*

$$|\Phi(t)| \leq \frac{C}{(1+|t|)^{1+\gamma}}, \quad t \in \mathbb{R}.$$

Entonces, si $1 \leq p < \infty$, para toda $f \in L^p(\mathbb{R})$ se tiene

$$\exists \lim_{\varepsilon\to 0^+} f * \Phi_\varepsilon(t) = f(t) \quad \text{en casi todo } t \in \mathbb{R}.$$

Como no utilizaremos este resultado en el trabajo, omitimos su demostración, que puede consultarse en el texto [**2**, Theorem 8.15].

2. Transformada de Fourier de funciones en $L^1(\mathbb{R})$

En esta sección vamos a tratar la transformada de Fourier y sus propiedades principales. Para ello seguimos la notación del libro de Mallat [**3**, §2.2], complementada en parte con resultados del libro de Folland [**2**, §8.3].

2.1. Definición y primeras propriedades.

Definición 1.2.1. *Para $f \in L^1(\mathbb{R})$, la transformada de Fourier de f se define como*

$$\hat{f}(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t}dt, \quad \omega \in \mathbb{R}. \tag{1.2.2}$$

La integral en 1.2.2 converge por

$$|\hat{f}(\omega)| \leq \int_{-\infty}^{\infty} |f(t)|dt < \infty.$$

Además, por el Teorema de Convergencia Dominada de Lebesgue, tenemos que $\hat{f}(\omega)$ es continua, es decir, si $\omega_n \to \omega$, entonces

$$\lim_{n\to\infty} \hat{f}(\omega_n) = \lim_{n\to\infty} \int_{\mathbb{R}} f(t)e^{-it\omega_n}dt = \hat{f}(\omega) \tag{1.2.3}$$

Proposición 1.2.4 (Propriedades algebraicas de la transformada de Fourier). *Sean $f, g \in L^1(\mathbb{R})$.*

i) Linealidad

$$\left(\widehat{\alpha f + \beta g}\right)(\omega) = \alpha\widehat{f}(\omega) + \beta\widehat{g}(\omega), \quad \alpha, \beta \in \mathbb{C}.$$

ii) Conjugación

$$\widehat{\overline{f}}(\omega) = \overline{\widehat{f}(-\omega)}.$$

iii) Traslación: Llamo $T_h f(t) = f(t+h)$, tenemos:

$$\widehat{T_h f}(\omega) = e^{i\omega t}\widehat{f}(\omega).$$

iv) Dilatación

$$\left[\widehat{f\left(\frac{\cdot}{R}\right)}\right] = R\widehat{f}\,(R\omega)$$

v) Modulación: Sea $g(t) = e^{ith}f(t)$, entonces

$$\widehat{g}(\omega) = \widehat{f}(\omega - h).$$

Demostración. i) Se prueba trivialmente usando la linealidad de la integral de Lebesgue y sacando las constantes fuera de la integral.

ii)

$$\widehat{\overline{f}}(\omega) = \int_{\mathbb{R}} \overline{f}(t)e^{-i\omega t}dt = \overline{\int_{\mathbb{R}} f(t)e^{i\omega t}dt} = \overline{\widehat{f}(-\omega)}.$$

iii)

$$\widehat{T_h f}(\omega) = \int_{\mathbb{R}} f(t+h)e^{-i\omega t}dt$$

Haciendo el cambio de variable $u = t + h$, la integral anterior se queda:

$$\widehat{T_h f(\omega)} = \int_{\mathbb{R}} f(u)e^{-i\omega(u-h)}du = \int_{\mathbb{R}} f(u)e^{i\omega h}e^{-i\omega u}du = e^{i\omega h}\widehat{f}(\omega)$$

iv)

$$\int_{\mathbb{R}} f(\frac{t}{R})e^{-i\omega t}dt = R\int_{\mathbb{R}} f(u)e^{-i\omega Ru}du = R\widehat{f}(R\omega),$$

donde en la integral hemos hecho el siguiente cambio de variables $\frac{t}{R} = u$.

v)

$$\widehat{g}(\omega) = \int_{\mathbb{R}} g(t)e^{-i\omega t}dt = \int_{\mathbb{R}} e^{iht}f(t)e^{-i\omega t}dt = \int_{\mathbb{R}} f(t)e^{-i(\omega-h)t}dt = \widehat{f}(\omega - h).$$

□

2.2. El Teorema de Inversión de la Transformada de Fourier. El teorema principal de esta sección nos dice como obtener $f(t)$ a partir de su transformada $\hat{f}(\omega)$. Ver [**3**, §2.1], o bien [**2**, §8.3].

TEOREMA 1.2.5 (Inversión de la transformada de Fourier). *Si* $f \in L^1(\mathbb{R})$ *y* $\hat{f} \in L^1(\mathbb{R})$ *entonces se tiene*

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \hat{f}(\omega)e^{i\omega t}d\omega, \quad \text{ctp } t \in \mathbb{R}. \tag{1.2.6}$$

Para demostrar el teorema necesitaremos algunos resultados auxiliares.

LEMA 1.2.7. *Si* $\Phi(t) = e^{-t^2}$ *entonces* $\hat{\Phi}(\omega) = \sqrt{\pi}e^{-\frac{\omega^2}{4}}$

DEMOSTRACIÓN. Obsérvese que

$$\begin{aligned}
\hat{\Phi}(\omega) &= \int_{-\infty}^{\infty} \Phi(t)e^{-i\omega t}dt = \int_{-\infty}^{\infty} e^{-t^2}e^{-i\omega t}dt \\
&= \int_{-\infty}^{\infty} e^{-(t^2+i\omega t)}dt = \int_{-\infty}^{\infty} e^{-[t^2+i\omega t+(\frac{i\omega}{2})^2-(\frac{iw}{2})^2]}dt \\
&= e^{\frac{-w^2}{4}}\int_{-\infty}^{\infty} e^{-(t+\frac{\omega i}{2})^2}dt.
\end{aligned}$$

Ahora, haciendo el cambio de variables $t + \frac{i\omega}{2} = z$. entonces tenemos $dt = dz$, y por tanto:

$$\hat{\Phi}(\omega) = e^{-\frac{\omega^2}{4}}.\int_{-\infty}^{\infty} e^{-z^2}dz. \tag{1.2.8}$$

Veamos a continuación que

$$I = \int_{-\infty}^{\infty} e^{-z^2}dz = \sqrt{\pi} \tag{1.2.9}$$

En efecto, tenemos

$$I^2 = \int_{-\infty}^{\infty} e^{-u^2}du.\int_{-\infty}^{\infty} e^{-v^2}dv = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-u^2-v^2}dudv.$$

Cambiando variables, hacemos $u = r\cos\theta, v = r\,\text{sen}\,\theta$, tenemos : $-u^2 - v^2 = -r^2$ y

$$\left|\frac{\partial(u,v)}{\partial(\theta,r)}\right| = \begin{vmatrix} \frac{\partial u}{\partial\theta} & \frac{\partial u}{\partial r} \\ \frac{\partial v}{\partial\theta} & \frac{\partial v}{\partial\theta} \end{vmatrix} = \begin{vmatrix} -r\,\text{sen}\,\theta & \cos\theta \\ r\cos\theta & \text{sen}\,\theta \end{vmatrix} = |-r\,\text{sen}^2\,\theta - r\cos^2\theta| = r,$$

Resulta:

$$\begin{aligned} I^2 &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-u^2-v^2}dudv \\ &= \int_0^{2\pi}\int_0^{\infty} e^{-r^2} r\, d\theta dr = 2\pi\int_0^{\infty} e^{-r^2} r\, dr \;=\; 2\pi.\frac{1}{2} = \pi. \end{aligned}$$

Esto demuestra (1.2.9). Combinandolo con (1.2.8) tenemos:

$$\hat{\Phi}(\omega) = \sqrt{\pi}e^{-\frac{\omega^2}{4}}.$$

□

Considero $\Phi(t) = e^{-t^2}$, como en el Lema 1.2.7, y la familia de funciones

$$\Phi_\varepsilon(t) = \varepsilon^{-1}\Phi(t/\varepsilon), \quad \varepsilon > 0.$$

Obsérvese que $\{\frac{1}{\sqrt{\pi}}\Phi_\varepsilon\}_{\varepsilon\,0}$ es una AIR, por el Lema 1.1.7. La normalización con $\frac{1}{\sqrt{\pi}}$ es debida a que $\int_{\mathbb{R}}\Phi = \hat{\Phi}(0) = \sqrt{\pi}$, por el Lema 1.2.7. Además, usando las propiedades de la transformada de Fourier tenemos,

$$\widehat{\Phi_\varepsilon}(\omega) = \hat{\Phi}(\epsilon\omega) = \sqrt{\pi}e^{-\frac{\epsilon^2\omega^2}{4}}.$$

Sea

$$I_\epsilon(t) = \frac{1}{2\pi}\int_{\mathbb{R}} \hat{f}(\omega)e^{-\frac{\epsilon^2\omega^2}{4}}e^{it\omega}\, d\omega.$$

Es claro que :

$$|\hat{f}(\omega)e^{-\frac{\epsilon^2\omega^2}{4}}e^{it\omega}| \le |\hat{f}(\omega)| \in L^1(\mathbb{R}).$$

Por el Teorema de la Convergencia Dominada de Lebesgue tenemos :

$$\begin{aligned} \frac{1}{2\pi}\int_{\mathbb{R}} \hat{f}(\omega)e^{it\omega}dt &= \lim_{\epsilon\to 0} I_\epsilon(t) \\ &= \lim_{\varepsilon\to 0}\frac{1}{2\pi}\int_{\mathbb{R}}\left[\int_{\mathbb{R}} f(y)e^{-iyt}.e^{-\frac{\epsilon^2\omega^2}{4}}\, dy\right] e^{it\omega}\, d\omega \\ &\overset{\text{Fubini}}{=} \lim_{\epsilon\to 0}\frac{1}{2\pi}\int_{\mathbb{R}} f(y)\left(\int_{\mathbb{R}} e^{i(t-y)\omega}.e^{-\frac{\epsilon^2\omega^2}{4}}d\omega\right) dy. \end{aligned} \tag{1.2.10}$$

Considero la integral de dentro del paréntesis, que escribo como

$$J = \int_R e^{-i(y-t)\omega}\Phi(\frac{\epsilon\omega}{2})d\omega. \tag{1.2.11}$$

Cambiando variables $\frac{\epsilon\omega}{2} = \xi$ entonces $d\omega = \frac{2}{\epsilon}d\xi$, y tenemos:

$$\begin{aligned} J &= \frac{2}{\epsilon}\int_R e^{-\frac{2i(y-t)\xi}{\epsilon}}\Phi(\xi)d\xi = \frac{2}{\epsilon}\hat{\Phi}\left[\frac{2(y-t)}{\epsilon}\right] \\ &\overset{\text{Lema 1}}{=} \frac{2}{\epsilon}\sqrt{\pi}e^{-\frac{(y-t)^2}{\varepsilon^2}} = 2\sqrt{\pi}\Phi_\epsilon(t-y) \end{aligned}$$

Por tanto, podemos escribir

$$I_\epsilon(t) = \frac{1}{\sqrt{\pi}}\int_{\mathbb{R}} f(y)\Phi_\epsilon(t-y)dy = \frac{1}{\sqrt{\pi}}f * \Phi_\epsilon(t)$$

Como $\{\frac{1}{\sqrt{\pi}}\Phi_\epsilon(t)\}_{\varepsilon>0}$ es una AIR, usando el Teorema 1.1.8 (con $p = 1$), obtenemos

$$I_\epsilon(t) = \frac{1}{\sqrt{\pi}} f * \Phi_\epsilon(t) = f * k_\epsilon(t) \overset{\epsilon \to 0}{\to} f(t) \ .$$

La convergencia es en la norma $L^1(\mathbb{R})$, pero existe una subsucesión que converge en casi todo punto $t \in \mathbb{R}$. Insertando esta expresión en (1.2.10), obtendremos la fórmula (1.2.6) buscada. □

La fórmula de inversión del Teorema 1.2.5 podría interpretarse como una descomposición de f como "suma infinita" de ondas $e^{i\omega t}$ con amplitud $\hat{f}(\omega)$.

Como consecuencia de (1.2.6), obsérverse que las hipótesis $f, \hat{f} \in L^1(\mathbb{R})$, implican que f es continua (tras modificarla, si fuera necesario, en un conjunto de medida nula), usando el mismo argumento que en (1.2.3). Existen también variantes de la fórmula de inversión para funciones f discontinuas a trozos, aunque aquí no las usaremos.

3. Transformada de Fourier en $L^2(\mathbb{R})$ y Teorema de Plancherel

Además de la definición dada en (1.2.2), se puede extender la definición de transformada de Fourier a funciones del espacio $L^2(\mathbb{R})$, es decir, funciones $f : \mathbb{R} \to \mathbb{C}$ medibles tales que

$$\|f\|_2^2 = \int_{-\infty}^{\infty} |f(t)|^2 dt < \infty.$$

En las aplicaciones a la teoría de señales, a la expresión $\|f\|_2^2$ se le denomina "energía" de la señal f, y el espacio L^2 es el conjunto de todas las señales de energía finita.

Consideremos el siguiente ejemplo $f_T = \frac{1}{T}\chi_{[-T,T]}$. Entonces, un cálculo sencillo da

$$\hat{f_T}(\omega) = \frac{1}{T}\int_{-T}^{T} e^{-i\omega t} dt = \frac{2\,\mathrm{sen}(T\omega)}{T\omega}. \tag{1.3.1}$$

Es posible demostrar que $\hat{f}_T \notin L^1(\mathbb{R})$, y por tanto el Teorema de Inversión no es aplicable en este caso. Sin embargo, sí se tiene que $f_T, \hat{f}_T \in L^2(\mathbb{R})$. A continuación veremos que la transformada de Fourier puede definirse en $L^2(\mathbb{R})$ y que hay una fórmula xde inversión también en este caso.

Para ello, utilizaremos que $L^2(\mathbb{R})$ es un espacio de Hilbert. En particular, la norma proviene de un producto escalar

$$\|f\|_2^2 = \langle f, f\rangle, \quad \text{donde } \langle f, g\rangle = \int_{-\infty}^{\infty} f(t)\overline{g(t)} dt.$$

La extensión de la transformada de Fourier de f a $L^2(\mathbb{R})$ se hace usando la densidad de $L^1(\mathbb{R}) \cap L^2(\mathbb{R})$ en $L^2(\mathbb{R})$. Es más, la transformada de Fourier tiene la propiedad de ser una *isometría* en L^2, o en lenguaje de teoría de señales, de conservar la energía. Este importante resultado se conoce como *Teorema de Plancherel*.

Teorema 1.3.2 (de Plancherel). *Si $f \in L^1 \cap L^2(\mathbb{R})$, entonces $\hat{f} \in L^2(\mathbb{R})$ y además se tiene*

$$\int_{\mathbb{R}} |\hat{f}(\omega)|^2 \, d\omega \;=\; 2\pi \int_{\mathbb{R}} |f(t)|^2 \, dt. \tag{1.3.3}$$

Demostración. Utilizamos la función auxiliar $\Phi(t) = e^{-t^2}$ del Lema 1.2.7, y la AIR asociada $\Phi_\varepsilon(t) = \frac{1}{\varepsilon}\Phi(\frac{t}{\varepsilon})$. Calculamos en primer lugar

$$I_\varepsilon = \int_{\mathbb{R}} |\hat{f}(\omega)|^2 e^{-\frac{\omega^2\varepsilon^2}{4}} d\omega.$$

Usando que $|\hat{f}(\omega)|^2 = \hat{f}(\omega)\overline{\hat{f}(\omega)}$ y la propiedad de conjugación de la transformada de Fourier, se sigue:

$$I_\varepsilon = \int_{\mathbb{R}} \left(\int_{\mathbb{R}} f(t) e^{-it\omega} dt \right) \left(\int_{\mathbb{R}} \overline{f(y)} e^{iy\omega} dy \right) e^{\frac{-\varepsilon^2\omega^2}{4}} d\omega.$$

Como $|f(t)e^{-it\omega}\overline{f(y)}e^{ity}e^{\frac{-\varepsilon^2\omega^2}{4}}| = |f(t)||f(y)|e^{\frac{-\varepsilon^2\omega}{4}} \in L^1(\mathbb{R}^3)$ podemos usar el teorema de Fubini en la integral I_ε, y tenemos:

$$I_\varepsilon = \int_{\mathbb{R}} \int_{\mathbb{R}} f(t)\overline{f}(y) \left(\int_{\mathbb{R}} e^{-i(t-y)\omega} e^{\frac{-\varepsilon^2\omega^2}{4}} d\omega \right) dt dy.$$

La integral dentro del paréntesis es la integral J que nos apareció en (1.2.11), y cuyo valor calculamos en la demostración anterior $J = 2\sqrt{\pi}\Phi_\varepsilon(t-y)$. Entonces tenemos:

$$\begin{aligned} I_\varepsilon &= 2\sqrt{\pi} \int_R f(t) \left[\overline{\int_R f(y)\Phi_\varepsilon(t-y)dy} \right] dt \\ &= 2\sqrt{\pi} \int_R f(t) \overline{\Phi_\varepsilon * f(t)} dt = 2\sqrt{\pi} < f, \Phi_\varepsilon * f > . \end{aligned}$$

Como $\frac{1}{\sqrt{\pi}}\Phi_\varepsilon * f \longrightarrow f \in L^2(\mathbb{R})$ por el Teorema 1.1.8 de convergencia de AIR, tenemos que existe

$$\lim_{\varepsilon\to 0} I_\varepsilon = 2\sqrt{\pi}\sqrt{\pi} < f, f >= 2\pi \int_{\mathbb{R}} |f(t)|^2 \, dt.$$

Es decir, hemos llegado a:

$$\lim_{\varepsilon\to 0} \int_{\mathbb{R}} |\hat{f}(\omega)|^2 e^{\frac{-\omega^2\varepsilon^2}{4}} d\omega = 2\pi \int_{\mathbb{R}} |f(t)|^2 dt$$

Por otro lado, usando en la primera integral el Teorema de Convergencia Monótona de Lebesgue, queda:

$$\lim_{\varepsilon\to 0} \int_{\mathbb{R}} |\hat{f}(\omega)|^2 e^{\frac{-\omega^2\varepsilon^2}{4}} d\omega = \int_{\mathbb{R}} |\widehat{f}(\omega)|^2 \lim_{\varepsilon\to 0} e^{-\frac{\omega^2\varepsilon^2}{4}} d\omega = \int_{\mathbb{R}} |\widehat{f}(\omega)|^2 d\omega.$$

Por tanto,

$$\widehat{f} \in L^2(\mathbb{R}) \quad \text{y} \quad \|\hat{f}\|_2^2 = 2\pi\|f\|_2^2.$$

□

Utilizando la densidad de $L^1 \cap L^2$ en $L^2(\mathbb{R})$, podemos considerar la siguiente definición.

Definición 1.3.4. *Si $f \in L^2(\mathbb{R})$, definimos su transformada de Fourier como*

$$\mathcal{F}f = \ L^2\text{-}\lim_{n\to\infty} \hat{f}_n, \tag{1.3.5}$$

donde $\{f_n\}_{n=1}^{\infty}$ es una sucesión cualquiera en $L^1 \cap L^2(\mathbb{R})$ tal que L^2-$\lim_{n\to\infty} f_n = f$.

La definición anterior no depende de la sucesión elegida, pues si $f_n, g_n \in L^1 \cap L^2$ con L^2-$\lim_{n\to\infty} f_n = L^2$-$\lim_{n\to\infty} g_n = f$, entonces la igualdad de Plancherel (1.3.3) nos da

$$\|\hat{f}_n - \hat{g}_n\|_2^2 = \|\widehat{(f_n - g_n)}\|_2^2 = 2\pi\|f_n - g_n\|_2^2 \to 0,$$

y por tanto L^2-$\lim_{n\to\infty} \hat{f}_n = L^2$-$\lim_{n\to\infty} \hat{g}_n = \mathcal{F}f$. Además es claro que $\mathcal{F}f \in L^2(\mathbb{R})$ y se tiene

$$\|\mathcal{F}f\|_2 = \lim_{n\to\infty} \|\hat{f}_n\|^2 = \sqrt{2\pi} \lim_{n\to\infty} \|f_n\|_2 = \sqrt{2\pi}\|f\|_2,$$

que es una extensión de la fórmula de Plancherel (1.3.3) a funciones $f \in L^2(\mathbb{R})$.

Corolario 1.3.6. *El operador $\mathcal{F} : L^2(\mathbb{R}) \to L^2(\mathbb{R})$ es un isomorfismo que cumple*

$$\|\mathcal{F}f\|_2 \ = \ \sqrt{2\pi}\, \|f\|_2, \quad \forall\ f \in L^2(\mathbb{R}). \tag{1.3.7}$$

Además, el operador inverso viene dado por

$$\mathcal{F}^{-1}f(\omega) = \frac{1}{2\pi}\mathcal{F}f(-\omega), \quad f \in L^2(\mathbb{R}). \tag{1.3.8}$$

Demostración. La fórmula (1.3.7) ya la justificamos antes del corolario. En particular implica que $\mathcal{F}$ es inyectivo, pues si $\mathcal{F}f = 0$, entonces $f = 0$. Falta ver que $\mathcal{F}$ es sobreyectivo, y que la inversa viene dada por (1.3.8). Considero el conjunto

$$\mathcal{D} = \Big\{ f \in L^1 \cap L^2(\mathbb{R}) \ : \ \ \hat{f} \in L^1 \cap L^2(\mathbb{R}) \Big\}.$$

Este conjunto es denso en L^2 pues contiene a las funciones $C_c^{\infty}(\mathbb{R})$. Si $f \in \mathcal{D}$, el Teorema de inversión 1.2.5 nos da

$$f(t) = \frac{1}{2\pi}\int_{\mathbb{R}} \hat{f}(\omega)\, e^{i\omega t}\, d\omega = \frac{1}{2\pi}\int_{\mathbb{R}} \hat{f}(-\omega)\, e^{-i\omega t}\, d\omega = \mathcal{F}\Big[\frac{\mathcal{F}f(-\omega)}{2\pi}\Big](t). \tag{1.3.9}$$

Utilizando la densidad de $\mathcal{D}$, y la continuidad de $\mathcal{F}$, la fórmula (1.3.9) se extiende a toda $f \in L^2(\mathbb{R})$. Esto implica la identidad buscada (1.3.8). □

Nota: En el resto del trabajo usaremos indistintamente la notación $\hat{f} = \mathcal{F}f$ cuando $f \in L^1(\mathbb{R}) \cup L^2(\mathbb{R})$.

4. El teorema de muestreo de Shannon

Una función $f \in L^2(\mathbb{R})$ se dice de *banda limitada* B cuando su transformada de Fourier $\hat{f}$ cumple $\operatorname{sop} \hat{f} \subset [-B, B]$, es decir

$$\hat{f}(\omega) = 0, \quad \forall\ |\omega| > B.$$

Intuitivamente, la función f sólo "contiene" frecuencias $\leq B$, y por tanto no puede oscilar demasiado. El siguiente teorema garantiza que para tales funciones se puede recuperar cualquier valor de $f(t)$, $t \in \mathbb{R}$, a partir de un conjunto *discreto* de muestras $\{f(nT)\}_{n\in\mathbb{Z}}$, donde el paso T está relacionado con B mediante la relación

$$T = \pi/B.$$

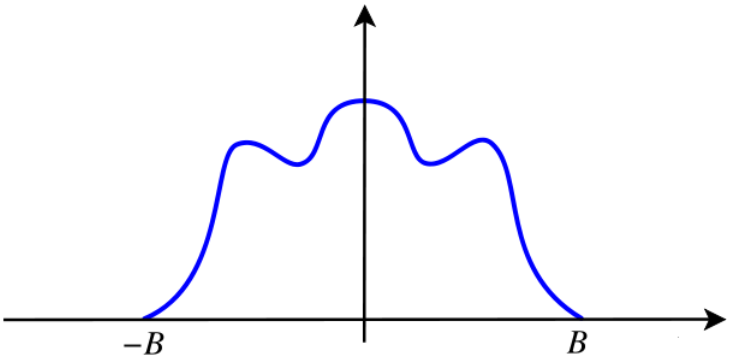


FIGURA 1. shannon

TEOREMA 1.4.1. (Whittaker (1915), Shannon (1948))
Si $f \in L^2(\mathbb{R})$ *y* $\operatorname{sop} \hat{f} \subset [-B, B]$ *(es decir,* $\hat{f}(\omega) = 0$, $\forall\ |\omega| > B$*), entonces*

$$f(t) = \sum_{n\in\mathbb{Z}} f\Big(\frac{n\pi}{B}\Big)\,\frac{\operatorname{sen}(Bt - \pi n)}{Bt - \pi n}, \quad t \in \mathbb{R}, \tag{1.4.2}$$

donde la serie anterior converge en la norma de L^2*, y también absoluta y uniformemente en todo* $t \in \mathbb{R}$*.*

DEMOSTRACIÓN. Desarrollamos la función $\hat{f}(\omega) \in L^2([-B, B])$ como serie de Fourier en el intervalo $[-B, B]$, es decir:

$$\hat{f}(\omega) = \frac{1}{2B}\sum_{n\in\mathbb{Z}} \langle \hat{f}, e^{\frac{2\pi i n\xi}{2B}}\rangle\, e^{\frac{2\pi i n\xi}{2B}} \cdot \chi_{[-B,B]}(\omega) \tag{1.4.3}$$

donde la serie convege en la norma de $L^2[-B, B]$. Notar que la igualdad (1.4.3) es cierta en todo $\omega \in \mathbb{R}$, y la convergencia se tiene en $L^2(\mathbb{R})$, porque $\operatorname{sop} \hat{f} \subset [-B, B]$. Además, podemos escribir

$$\langle \hat{f}, e^{\frac{2\pi i n\omega}{2B}}\rangle = \int_{-B}^{B} \hat{f}(\omega) e^{-\frac{i\pi n\omega}{B}} d\omega = \int_{\mathbb{R}} \hat{f}(\omega) e^{-\frac{i\pi n\omega}{B}} d\omega = 2\pi f\big(\tfrac{-\pi n}{B}\big), \quad n \in \mathbb{Z}, \tag{1.4.4}$$

por la fórmula (1.2.6) de inversión de la transformada de Fourier. Es decir, tenemos:

$$\hat{f}(\omega) = \tfrac{\pi}{B}\sum_{n\in\mathbb{Z}} f\big(\tfrac{\pi n}{B}\big)\, e^{-\frac{\pi i n\omega}{B}} \cdot \chi_{[-B,B]}(\omega),$$

con convergencia en $L^2(\mathbb{R})$. Tomando la transformada de Fourier inversa $\mathcal{F}^{-1}$, y usando el Corolario 1.3.6, tenemos

$$f(t) = \tfrac{\pi}{B}\sum_{n\in\mathbb{Z}} f\big(\tfrac{\pi n}{B}\big)\, \mathcal{F}^{-1}\Big[e^{-\frac{i\pi n\omega}{B}}\chi_{[-B,B]}\Big](t),$$

con convergencia en $L^2(\mathbb{R})$. Por último, usando las propiedades de la transformada de Fourier tenemos

$$\mathcal{F}^{-1}\Big[e^{-\frac{i\pi n\omega}{B}}\chi_{[-B,B]}\Big](t) \;=\; \tfrac{1}{2\pi}\,\widehat{\Big[e^{-\frac{i\pi n\omega}{B}}\chi_{[-B,B]}\Big]}(-t) = \tfrac{1}{2\pi}\,\widehat{\chi_{[-B,B]}}\big(\tfrac{\pi n}{B} - t\big),$$

y usando $\widehat{\chi_{[-B,B]}}(x) = 2\text{sen}(Bx)/x$ ahora tenemos:

$$f(t) = \sum_{n\in\mathbb{Z}} f\big(\tfrac{\pi n}{B}\big)\, \frac{\text{sen}(tB-\pi n)}{tB-\pi n}, \quad t\in\mathbb{R},$$

como queríamos demostrar.

Por último, para justificar que la convergencia es uniforme obsérvese que, por la desigualdad de Cauchy-Schwarz, las colas de la serie pueden acotarse por

$$E_N(t) = \sum_{|n|>N} \Big| f\big(\tfrac{\pi n}{B}\big)\, \frac{\sin(tB-\pi n)}{tB-\pi n}\Big| \le \Big(\sum_{|n|>N} |f\big(\tfrac{\pi n}{B}\big)|^2\Big)^{\frac12} \sup_{x\in\mathbb{R}} \Big(\sum_{n\in\mathbb{Z}} \Big|\frac{\sin(x-\pi n)}{x-\pi n}\Big|^2\Big)^{\frac12}.$$

Por un lado, la teoría de series de Fourier y la fórmula (1.4.4) nos dicen que $\{f\big(\tfrac{\pi n}{B}\big)\} \in \ell^2(\mathbb{Z})$. Por otro lado, por π-periodicidad, tenemos que

$$\begin{aligned}\sup_{x\in\mathbb{R}} \sum_{n\in\mathbb{Z}} \Big|\frac{\sin(x-\pi n)}{x-\pi n}\Big|^2 &= \sup_{x\in[-\frac{\pi}{2},\frac{\pi}{2}]} \sum_{n\in\mathbb{Z}} \Big|\frac{\sin(x-\pi n)}{x-\pi n}\Big|^2 \\ &\le \sup_{x\in[-\frac{\pi}{2},\frac{\pi}{2}]} 1 + \sum_{n\neq 0} \frac{1}{|x-n\pi|^2} \le 1 + \sum_{n\neq 0} \frac{1}{|n\pi/2|^2} = cte,\end{aligned}$$

usando en la última desigualdad que $|x-n\pi| \ge |n\pi| - |x| \ge |n\pi| - \pi/2 \ge |n|\pi/2$. De estas observaciones se sigue que

$$\lim_{N\to\infty} \sup_{t\in\mathbb{R}} E_N(t) = 0,$$

y por tanto la convergencia uniforme (y absoluta) de la serie en (1.4.2). □

Capítulo 2

La transformada de Fourier discreta

En este capítulo desarrollamos la teoría de Fourier cuando las funciones $f(t)$ son discretas y finitas, es decir cuando la variable t tiene como dominio un conjunto finito de N puntos. En este capítulo seguimos en parte la notación y las demostraciones del libro de Mallat [**3**, §3,3].

1. Señales Finitas

Llamamos *señal discreta* de tamaño N a cualquier vector

$$\mathbf{f} = \big(f[0], f[1], \ldots, f[N-1]\big) \in \mathbb{C}^N.$$

El conjunto de las señales discretas de tamaño N es el espacio vectorial $\mathbb{C}^N$, que dotaremos con la norma euclídea habitual

$$\|\mathbf{f}\|^2 = |f[0]|^2 + |f[1]|^2 + \ldots + |f[N-1]|^2.$$

A la expresión anterior se le llama a veces *energía* de la señal $\mathbf{f}$. Esta norma proviene del producto escalar

$$\langle \mathbf{f}, \mathbf{g} \rangle = \sum_{0 \le n < N} f[n]\, \overline{g[n]}, \quad \mathbf{f}, \mathbf{g} \in \mathbb{C}^N. \tag{2.1.1}$$

En particular, el conjunto de las señales discretas de tamaño N se identifica con el espacio de Hilbert $\ell^2(N) = \big(\mathbb{C}^N, \langle \cdot, \cdot \rangle\big)$.

Sea $\{\mathbf{E}_n\}_{0 \le n < N}$ la base canónica de $\mathbb{C}^N$, es decir $\mathbf{E}_n[j] = 0$ si $n \neq j$ y $\mathbf{E}_n[n] = 1$. Entonces

$$\mathbf{f} = f[0]\mathbf{E}_0 + \ldots + f[N-1]\mathbf{E}_{N-1}.$$

En la próxima sección construimos una base alternativa, de tipo trigonométrico, para representar las señales de $\mathbb{C}^N$. Los coeficientes en esta nueva base permitirán definir la transformada de Fourier de $\mathbf{f}$.

2. Transformada de Fourier Discreta (TFD)

Definición 2.2.1. *Si* $\mathbf{f} = \big(f[0], \ldots, f[N-1]\big) \in \mathbb{C}^N$, *definimos su* Transformada de Fourier Discreta *(TFD) como la señal* $\hat{\mathbf{f}} = \big(\hat{f}[0], \ldots, \hat{f}[N-1]\big)$ *con coeficientes:*

$$\hat{f}[k] := \sum_{n=0}^{N-1} f[n] e^{-\frac{2\pi i k n}{N}}, \quad 0 \le k < N. \tag{2.2.2}$$

A veces se escribe $\hat{\mathbf{f}} = \mathrm{TFD}(\mathbf{f})$.

Consideramos la siguiente familia de vectores $\mathcal{E}_N = \{\mathbf{e}_0, \mathbf{e}_1, \ldots, \mathbf{e}_{N-1}\}$ de $\mathbb{C}^N$, donde

$$\mathbf{e}_k[n] = e^{\frac{2\pi i k n}{N}}, \quad 0 \leq n < N. \tag{2.2.3}$$

Obsérvese que

$$\hat{f}[k] \;=\; \langle \mathbf{f}, \mathbf{e}_k \rangle. \tag{2.2.4}$$

Teorema 2.2.5. *La familia* $\{\mathbf{e}_0, \mathbf{e}_1, \ldots, \mathbf{e}_{N-1}\}$ *definida en* (2.2.3) *es una base ortogonal de* $\big(\mathbb{C}^N, \langle \cdot, \cdot \rangle\big)$.

Demostración. Como la dimensión del espacio es N, cualquier familia ortogonal de N vectores es una base ortogonal. Por tanto basta probar que

$$\langle \mathbf{e}_k, \mathbf{e}_l \rangle = \delta_{k,l}.$$

- Si $k = l$, el produto interno queda

$$\langle \mathbf{e}_l, \mathbf{e}_l \rangle \;=\; \sum_{n=0}^{N-1} e^{\frac{2\pi i l n}{N}} e^{-\frac{2\pi i l n}{N}} \;=\; \sum_{n=0}^{N-1} 1 \;=\; N. \tag{2.2.6}$$

- Si $k \neq l$, el produto interno queda

$$\begin{aligned} \langle \mathbf{e}_k, \mathbf{e}_l \rangle \;&=\; \sum_{n=0}^{N-1} e^{\frac{2\pi i (k-l) n}{N}} \;=\; \frac{1 - (e^{\frac{2\pi i (k-l)}{N}})^N}{1 - e^{\frac{2\pi i (k-l)}{N}}} \\ &=\; \frac{1 - e^{2\pi i (k-l)}}{1 - e^{\frac{2\pi i (k-l)}{N}}} \;=\; \frac{1-1}{1 - e^{\frac{2\pi i (k-l)}{N}}} = 0, \end{aligned}$$

  usando la fórmula de sumación de una progresión geométrica, y que el denominador no se anula (pues $|k-l| < N$ y $k \neq l$).

□

Como consecuencia obtenemos

Proposición 2.2.7. *Sea* $\mathbf{f} \in \mathbb{C}^N$ *una señal discreta. Entonces*

- Fórmula de inversión*:*

$$f[n] = \frac{1}{N} \sum_{k=0}^{N-1} \hat{f}[k] e^{\frac{2\pi i k n}{N}}, \quad 0 \leq n < N \tag{2.2.8}$$

- Fórmula de Plancherel*:*

$$\|f\|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |\hat{f}[k]|^2. \tag{2.2.9}$$

Demostración. Por el Teorema 2.2.5, cualquier señal $\mathbf{f} \in \mathbb{C}^N$ puede descomponerse como

$$\mathbf{f} = \textstyle\sum_{0 \leq n < N} \lambda_k \mathbf{e}_k,$$

con

$$\langle \mathbf{f}, \mathbf{e}_m \rangle \;=\; \langle \sum_{k=0}^{N-1} \lambda_k \mathbf{e}_k, \mathbf{e}_m \rangle \;=\; \lambda_m \|\mathbf{e}_m\|^2.$$

Con eso tenemos:

$$\mathbf{f} = \sum_{k=0}^{N-1} \frac{\langle \mathbf{f}, \mathbf{e}_k \rangle}{\|\mathbf{e}_k\|^2} \mathbf{e}_k \tag{2.2.10}$$

Usando (2.2.6) y (2.2.4) tenemos

$$\frac{\langle \mathbf{f}, \mathbf{e}_k \rangle}{\|\mathbf{e}_k\|^2} = \frac{\hat{f}[k]}{N},$$

y por tanto

$$f[n] = \frac{1}{N} \sum_{k=0}^{N-1} \hat{f}[k] e^{\frac{2\pi i k n}{N}},$$

que es la fórmula para invertir la TFD. De las relaciones de ortogonalidad del teorema 2.2.5 también se deduce

$$\begin{aligned} \|\mathbf{f}\|^2 &= \langle \mathbf{f}, \mathbf{f} \rangle = \Big\langle \sum_{k=0}^{N-1} \frac{\langle \mathbf{f}, \mathbf{e}_k \rangle}{\|\mathbf{e}_k\|^2} \mathbf{e}_k, \sum_{k=0}^{N-1} \frac{\langle f, \mathbf{e}_k \rangle}{\|\mathbf{e}_k\|^2} \mathbf{e}_k \Big\rangle \\ &= \sum_{k=0}^{N-1} \Big| \hat{f}[k] \Big|^2 \frac{1}{N^2} \langle \mathbf{e}_k, \mathbf{e}_k \rangle = \frac{1}{N} \sum_{k=0}^{N-1} |\hat{f}[k]|^2, \end{aligned}$$

que es la fórmula de Plancherel. □

Definición 2.2.11. *Se define la* Transformada de Fourier Inversa *de una señal* $\mathbf{g} \in \mathbb{C}^N$ *como*

$$\mathrm{IDFT}(\mathbf{g})[n] = \frac{1}{N} \sum_{k=0}^{N-1} g[k] e^{\frac{2\pi i k n}{N}}, \quad 0 \le n < N. \tag{2.2.12}$$

Como corolario de (2.2.8) se obtiene

$$\mathrm{IDFT}\big(\mathrm{TFD}(\mathbf{f})\big) = \mathbf{f}, \quad \forall\, \mathbf{f} \in \mathbb{C}^N. \tag{2.2.13}$$

3. Convoluciones Circulares

Dadas dos señales $\mathbf{f}, \mathbf{h} \in \mathbb{C}^N$, nos gustaría definir la operación de convolución como

$$\mathbf{f} * \mathbf{h}[n] = \sum_{0 \le m < N} f[m]\, h[n-m],$$

pero para ello es necesario tener valores de $f[n]$ y $h[n]$ fuera de $0 \le n < N$. Una posibilidad es extender las señales $\mathbf{f} \in \mathbb{C}^N$ como sucesiones N-periódicas con índices en $\mathbb{Z}$, es decir

$$f[n] := f[n, \text{ mód } N], \quad n \in \mathbb{Z}.$$

En esta sección utilizaremos esta extensión, identificando así $(\mathbb{C}^N, \langle \cdot, \cdot \rangle)$ con el espacio de Hilbert $\ell^2(\mathbb{Z}_N)$. Por ejemplo, las exponenciales discretas $\mathbf{e}_k$ de la sección anterior son automáticamente N-periódicas, pues $\mathbf{e}_k[n] = e^{\frac{2\pi i k n}{N}}$ tiene el mismo valor en $[n \text{ mód } N]$.

Definición 2.3.1. *Dadas* $\mathbf{f}, \mathbf{h} \in \mathbb{C}^N$*, se define su* Convolución Circular *como la señal* $\mathbf{f} \circledast \mathbf{h} \in \mathbb{C}^N$ *dada por*

$$\mathbf{f} \circledast \mathbf{h}[n] := \sum_{m=0}^{N-1} f[m]h[n-m] \quad , \, 0 \leq n < N. \tag{2.3.2}$$

Nótese que (2.3.2) define automáticamente una señal de N-periódica, cuando consideramos $f[n]$ y $h[n]$ como señales N-periódicas. Además, es fácil comprobar que

$$\mathbf{f} \circledast \mathbf{h}[n] = \sum_{m=0}^{N-1} f[n-m]h[m] = \mathbf{h} \circledast \mathbf{f}[n] \quad , \, 0 \leq n < N.$$

Dada una señal $h[n]$ de período N, consideramos el operador de convolución $L_{\mathbf{h}}$ definido por

$$\mathbf{f} \in \mathbb{C}^N \longmapsto L_{\mathbf{h}}\mathbf{f}[n] = \mathbf{f} \circledast \mathbf{h}[n].$$

Proposición 2.3.3. *Las exponenciales discretas* $\{\mathbf{e}_0, \ldots, \mathbf{e}_{N-1}\}$ *forman una base de autovectores de* $L_{\mathbf{h}}$*. Es más*

$$L_{\mathbf{h}}\mathbf{e}_k = \hat{h}[k]\, \mathbf{e}_k, \quad 0 \leq k < N.$$

Demostración.

$$\begin{aligned} L_{\mathbf{h}}\mathbf{e}_k[n] &= \mathbf{e}_k \circledast \mathbf{h}[n] = \sum_{p=0}^{N-1} h[p]\mathbf{e}_k[n-p] = \sum_{p=0}^{N-1} h[p]e^{\frac{2\pi i(n-p)k}{N}} \\ &= e^{\frac{2\pi ink}{N}} \sum_{p=0}^{N-1} h[p]e^{\frac{-2\pi ipk}{N}} = \mathbf{e}_k[n]\, \hat{h}[k]. \end{aligned}$$

□

Teorema 2.3.4. *Si* $\mathbf{f}, \mathbf{h} \in \mathbb{C}^N$*, entonces la señal convolución* $\mathbf{g} = \mathbf{f} \circledast \mathbf{h}$ *cumple*

$$\hat{g}[k] = \hat{f}[k]\hat{h}[k], \quad 0 \leq k \leq N-1. \tag{2.3.5}$$

Demostración. Aplicando la definición tenemos

$$\begin{aligned} \hat{g}[k] &= \sum_{n=0}^{N-1} g[n]e^{-\frac{2\pi ikn}{N}} = \sum_{n=0}^{N-1} \left[\sum_{p=0}^{N-1} f[p]h[n-p] \right] e^{-\frac{2\pi ikn}{N}} \\ &= \sum_{n=0}^{N-1} \left[\sum_{p=0}^{N-1} f[p]h[n-p] \right] e^{-\frac{2\pi ik(n-p)}{N}} \cdot e^{-\frac{2\pi ikp}{N}} \\ &= \sum_{p=0}^{N-1} \left[\sum_{n=0}^{N-1} h[n-p]e^{-\frac{2\pi ik(n-p)}{N}} \right] f[p]e^{-\frac{2\pi ikp}{N}}. \end{aligned} \tag{2.3.6}$$

Haciendo el cambio $m = n - p$ en la suma de dentro del paréntesis de la expresión anterior tenemos:

$$\begin{aligned}\sum_{n=0}^{N-1} h[n-p]e^{-\frac{2\pi ik(n-p)}{N}} &= \sum_{m=-p}^{N-1-p} h[m]e^{-\frac{2\pi ikm}{N}} \\ &= \sum_{m=-p}^{-1} h[m]e^{-\frac{2\pi ikm}{N}} + \sum_{m=0}^{N-1-p} h[m]e^{-\frac{2\pi ikm}{N}} \\ &= \sum_{m=-p}^{-1} h[m+N]e^{-\frac{2\pi ik(m+N)}{N}} + \sum_{m=0}^{N-1-p} h[m]e^{-\frac{2\pi ikm}{N}},\end{aligned}$$

usando en el último paso la N-periodicidad de las señales. Cambiando de nuevo variables $l = m + N$ en el primer sumando resulta:

$$\begin{aligned}\sum_{n=0}^{N-1} h[n-p]e^{-\frac{2\pi ik(n-p)}{N}} &= \sum_{l=N-p}^{N-1} h[l]e^{-\frac{2\pi ikl}{N}} + \sum_{m=0}^{N-1-p} h[m]e^{\frac{-2\pi ikm}{N}} \\ &= \sum_{l=0}^{N-1} h[l]e^{-\frac{2\pi ikl}{N}} = \hat{h}[k].\end{aligned}$$

Sustituyendo lo que obtenemos en (2.3.6) se obtiene finalmente

$$\hat{g}[k] = \sum_{p=0}^{N-1} \hat{h}[k]f[p]e^{-\frac{2\pi kp}{N}} = \hat{h}[k]\sum_{p=0}^{N-1} f[p]e^{-\frac{2\pi kp}{N}} = \hat{h}[k]\hat{f}[k].$$

□

4. La Transformada de Fourier Rápida (FFT)

Para una señal discreta $\mathbf{f} \in \mathbb{C}^N$ el cálculo de su señal transformada $\hat{\mathbf{f}}$ puede ser operacionalmente muy costoso. En efecto, para obtener todos los coeficientes

$$\hat{f}[k] = \sum_{n=0}^{N-1} f[n]e^{-\frac{2\pi ikn}{N}}, \qquad 0 \le k < N \tag{2.4.1}$$

es necesario hacer $O(N^2)$ operaciones de numeros complejos. Para ser más preciso, se necesitan N^2 multiplicaciones y $N(N-1)$ sumas.

Ejemplo 2.4.2. Por ejemplo, si la señal tiene tamaño $N = 2^{20} \approx 10^6$ y un ordenador hace $2^{30} \approx 10^9$ operaciones por segundo, se tardaría aproximadamente

$$\frac{2 \times 2^{40} - 2^{20}}{2^{30}} \approx 2 \times 2^{10} = 2048 \text{ seg } \approx 34 \text{ min}$$

en calcular la TFD usando (2.4.1).

La trasformada rápida de Fourier (FFT) es un algoritmo que permite calcular la TFD reduciendo considerablemente la complejidud de los cálculos, hasta llegar a sólo a $O(N \log_2 N)$ operaciones. El único requisito es que el tamaño de la señal sea una potencia de 2, es decir $N = 2^l$.

Teorema 2.4.3. **(Cooley, Tukey, 1965).**
Si $N = 2^l$, entonces existe un algoritmo que calcula la TFD de cada $\mathbf{f} \in \mathbb{C}^N$ utilizando a lo sumo $\kappa N \log_2 N$ operaciones complejas, pudiendo tomarse $\kappa = 3/2$.

Ejemplo 2.4.4. Así, en el ejemplo anterior, si $N = 2^{20}$ y el ordenador que hace 2^{30} operaciones por segundo, este algoritmo tardaría

$$\frac{1'5 \times 2^{20} \log_2 2^{20}}{2^{30}} = \frac{30 \times 2^{20}}{2^{30}} = \frac{30}{2^{10}} \approx \frac{30}{10^6} = 3 \cdot 10^{-5} \text{ seg.}$$

Ahora expliquemos con detalles el algoritmo de la *transformada rápida de Fourier*. Sea $N = 2^l$. Para frecuencias pares agrupamos los sumandos que definen $\hat{f}[2k]$ como sigue:

$$\hat{f}[2k] = \sum_{n=0}^{\frac{N}{2}-1} f[n] e^{-\frac{2\pi i(2k)n}{N}} + \sum_{n=\frac{N}{2}}^{N-1} f[n] e^{-\frac{2\pi i(2k)n}{N}}.$$

Haciendo en el segundo sumando el cambio $n = m + \frac{N}{2}$ tenemos:

$$\begin{aligned}
\hat{f}[2k] &= \sum_{n=0}^{\frac{N}{2}-1} f[n] e^{-\frac{2\pi ikn}{\frac{N}{2}}} + \sum_{m=0}^{\frac{N}{2}-1} f\left[m + \frac{N}{2}\right] e^{-\frac{2\pi ik(m+\frac{N}{2})}{\frac{N}{2}}} \\
&= \sum_{n=0}^{\frac{N}{2}-1} f[n] e^{-\frac{2\pi ikn}{\frac{N}{2}}} + \sum_{m=0}^{\frac{N}{2}-1} f\left[m + \frac{N}{2}\right] e^{-\frac{2\pi km}{\frac{N}{2}}} e^{-\frac{2\pi ik(\frac{N}{2})}{\frac{N}{2}}} \\
&= \sum_{n=0}^{\frac{N}{2}-1} f[n] e^{-\frac{2\pi ikn}{\frac{N}{2}}} + \sum_{m=0}^{\frac{N}{2}-1} f\left[m + \frac{N}{2}\right] e^{-\frac{2\pi ikn}{\frac{N}{2}}} e^{-2\pi ik} \\
&= \sum_{n=0}^{\frac{N}{2}-1} \left(f[n] + f\left[n + \frac{N}{2}\right] \right) e^{-\frac{2\pi ikn}{\frac{N}{2}}} \qquad (2.4.5)
\end{aligned}$$

Esta fórmula muestra que cuando las frecuencias son pares, $\hat{f}[2k]$ puede calcularse haciendo la TFD de la señal

$$f_p[n] = f[n] + f[n + \frac{N}{2}], \quad 0 \le n < N/2,$$

que es $N/2$ periódica. Si el indice de frecuencias es impar hacemos el mismo tipo de reagrupamiento para obtener:

$$\hat{f}[2k+1] = \sum_{n=0}^{\frac{N}{2}-1} f[n] e^{-\frac{2\pi i(2k+1)n}{N}} + \sum_{n=\frac{N}{2}}^{N-1} f[n] e^{-\frac{2\pi i(2k+1)n}{N}},$$

y con el cambio $n = m + \frac{N}{2}$ en la segunda suma obtenemos:

$$\begin{aligned}
\hat{f}[2k+1] &= \sum_{n=0}^{\frac{N}{2}-1} e^{-\frac{2\pi i n}{N}} f[n] e^{-\frac{2\pi i k n}{\frac{N}{2}}} + \sum_{m=0}^{\frac{N}{2}-1} f\left[m+\frac{N}{2}\right] e^{-\frac{2\pi i(2k+1)(m+\frac{N}{2})}{N}} \\
&= \sum_{n=0}^{\frac{N}{2}-1} e^{-\frac{2\pi i n}{N}} f[n] e^{-\frac{2\pi i k n}{\frac{N}{2}}} + \sum_{m=0}^{\frac{N}{2}-1} f\left[m+\frac{N}{2}\right] e^{-\frac{2\pi i(2k+1)m}{N}} e^{-\frac{2\pi i(2k+1)\frac{N}{2}}{N}} \\
&= \sum_{n=0}^{\frac{N}{2}} e^{-\frac{2\pi i n}{N}} f[n] e^{-\frac{2\pi i k n}{\frac{N}{2}}} - \sum_{m=0}^{\frac{N}{2}-1} e^{-\frac{2\pi i m}{N}} f\left[m+\frac{N}{2}\right] e^{-\frac{2\pi i k m}{\frac{N}{2}}} \\
&= \sum_{n=0}^{\frac{N}{2}-1} e^{-\frac{2\pi i n}{N}} \left(f[n] - f[n+\frac{N}{2}]\right) e^{-\frac{2\pi i k n}{\frac{N}{2}}}. \qquad (2.4.6)
\end{aligned}$$

La expresión anterior muestra que cuando las frecuencias son impares, $\hat{f}[2k+1]$ puede calcularse haciando la TFD de la señal

$$f_i[n] = e^{-\frac{2\pi i n}{N}} \left(f[n] - f\left[n+\frac{N}{2}\right]\right), \quad 0 \le n < N/2,$$

que también es $N/2$-periódica. En efecto,

$$\begin{aligned}
f_i\left[n+\frac{N}{2}\right] &= e^{-2\pi i(\frac{n}{N}+\frac{1}{2})}\left(f\left[n+\frac{N}{2}\right] - f[n+N]\right) \\
&= -e^{-\frac{2\pi i n}{N}}\left(f\left[n+\frac{N}{2}\right] - f[n]\right) = f_i[n].
\end{aligned}$$

Las fórmulas (2.4.5) y (2.4.6) muestran que la TFD de una señal discreta de tamaño $N = 2^l$ puede calcularse haciendo dos TFD de señales de tamaño $\frac{N}{2} = 2^{l-1}$.

Denotaremos por $C(N)$ el número de sumas y multiplicaciones complejas que es necesario para calcular TFD usando el algoritmo FFT descrito arriba. Dada $\mathbf{f} \in \mathbb{C}^N$, la construcción de $\mathbf{f}_p$ y $\mathbf{f}_i$ require de N sumas complejas y $\frac{N}{2}$ multiplicaciones complejas, es decir $\frac{3N}{2}$ operaciones, que denotaremos por κN. Por tanto tenemos:

$$C(N) = 2C\left(\frac{N}{2}\right) + \frac{3}{2}N = 2C\left(\frac{N}{2}\right) + \kappa N \quad , \qquad N = 2^l$$

Iterando esa fórmula se obtiene

$$\begin{aligned}
C(2^l) &= 2C(2^{l-1}) + \kappa 2^l \\
&= 2\left[2C(2^{l-2}) + \kappa 2^{l-1}\right] + \kappa 2^l \\
&= 2^2 C(2^{l-2}) + k2^l + k2^l \\
&= 2^2 C(2^{l-2}) + 2\kappa 2^l \\
&= 2^2\left[2C(2^{l-3}) + k2^{l-2}\right] + 2\kappa 2^l \\
&= 2^3 C(2^{l-3}) + 3\kappa 2^l \\
&= \cdots\cdots\cdots\cdots\cdots\cdots \\
&= 2^l C(1) + l\kappa 2^l
\end{aligned}$$

Pero $C(1) = 0$ porque si $\mathbf{f} \in \mathbb{C}$ es una señal de una sola muestra, entonces $\hat{f}[0] = f[0]$. Por lo tanto,

$$C(N) = C(2^l) = \kappa 2^l\, l = \kappa N \log_2 N, \qquad \kappa = \frac{3}{2}.$$

Nota 2.4.7. Para calcular la transformada inversa, IDFT, basta observar que

$$\text{IDFT}(\mathbf{g})[n] = \frac{1}{N}\sum_{k=0}^{N-1} g[k] e^{\frac{2\pi i k n}{N}} = \frac{1}{N}\sum_{k=0}^{N-1} \overline{\bar{g}[k] e^{\frac{-2\pi i k n}{N}}} = \frac{1}{N}\,\overline{\text{TFD}(\bar{\mathbf{g}})[n]}. \tag{2.4.8}$$

Por tanto la TFD inversa se puede calcular como el conjugado de una TFD, para lo que se puede usar el algoritmo de la FFT.

5. Convolución Rápida

Si recordamos la definición de convolución en (2.3.2), es decir

$$\mathbf{f} \circledast \mathbf{h}[n] := \sum_{m=0}^{N-1} f[m] h[n-m] \quad , \; 0 \le n < N,$$

vemos que son necesarias N^2 multiplicaciones y $N(N-1)$ sumas para poder determinar la señal $f \circledast h$, es decir, $O(N^2)$ operaciones. Usando el algoritmo FFT y el Teorema 2.3.4 puede desarrollarse un algoritmo para calcular la convolución de dos señales con $O(N \log_2 N)$ operaciones.

Teorema 2.5.1. *Si $\mathbf{f}, \mathbf{h} \in \mathbb{C}^N$ con $N = 2^\ell$, entonces se puede calcular la convolución circular $\mathbf{f} \circledast \mathbf{h}$ utilizando $3\kappa N \log_2 N + 4N$ operaciones complejas.*

Demostración. Sea $\mathbf{g} = \mathbf{f} \circledast \mathbf{h}$. Usando la fórmula (2.3.5) tenemos que

$$\hat{g}[k] = TFD(\mathbf{f} \circledast \mathbf{h})[k] = \hat{f}[k]\,\hat{h}[k], \quad 0 \le n < N.$$

Por el Teorema 2.4.3, para calcular todas estas expresiones necesitamos N multiplicaciones más $2\kappa N \log_2 N + N$ operaciones para las transformadas. A continuación, podemos utilizar la fórmula de inversión (2.2.13) y (2.4.8)

$$\mathbf{f} \circledast \mathbf{h} = \text{IDFT}\big[\text{TFD}(\mathbf{f} \circledast \mathbf{h})\big] = \tfrac{1}{N}\,\overline{\text{TFD}(\bar{\mathbf{g}})},$$

lo cual requiere otras $\kappa N \log_2 N$ operaciones para la transformada, y 3N operaciones para la multiplicación por $1/N$ y las conjugaciones. En total $3\kappa N \log_2 N + 4N$ operaciones complejas. □

Capítulo 3

Bases de cosenos

En esta sección introducimos las bases de cosenos, y la transformada coseno discreta, que en procesamiento de señales es más habitual que la TFD. Seguiremos para ello el libro de Mallat [**3**, §8.3], y los apuntes [**1**].

1. Bases de cosenos en $L^2[0,1]$

Sabemos que la colección de funciones $\{e_k(x) = e^{2\pi ikx}\}_{k=-\infty}^{\infty}$ es una base ortonormal de $[0,1)$. En particular, toda función $f \in L^2[0,1)$ se puede escribir como

$$f(x) = \sum_{k=-\infty}^{\infty} \langle f, e_k\rangle e_k(x) \;=\; \sum_{k=-\infty}^{\infty} \hat{f}(k) e^{2\pi ikx} \tag{3.1.1}$$

con convergencia en $L^2[0,1)$, donde

$$\hat{f}(k) = \langle f, e_k\rangle = \int_0^1 f(x) e^{-2\pi ikx} dx, \quad k \in \mathbb{Z},$$

se denomina *coeficiente de Fourier k-ésimo de f*.

Supongamos que $f(x)$ es suave en $[0,1)$, de modo que podemos integrar por partes, y obtener, para $k \neq 0$,

$$\begin{aligned}
\hat{f}(k) &= \Big[f(x)\frac{e^{-2\pi ikx}}{-2\pi ik}\Big]_0^1 + \int_0^1 f'(x)\frac{e^{-2\pi ikx}}{2\pi ik} dx \\
&= \frac{f(1^-) - f(0)}{-2\pi ik} + \int_0^1 f'(x)\frac{e^{-2\pi ikx}}{2\pi ik} dx \\
&= \frac{f(1^-) - f(0)}{-2\pi ik} + \frac{f'(1^-) - f'(0)}{-(2\pi ik)^2} + \int_0^1 \frac{1}{(2\pi ik)^2} f''(x) dx \qquad (3.1.2)
\end{aligned}$$

Cuando $f(1^-) = f(0)$, es decir, cuando la extensión 1-periódica de f es continua, entonces la fórmula (3.1.2) muestra que

$$|\hat{f}(k)| \leq \frac{C_f}{|2\pi k|^2}, \quad \text{con} \quad C_f = 2\|f'\|_\infty + \|f''\|_\infty,$$

y por tanto la serie en (3.1.1) converge uniforme y absolutamente. Es más, podemos aproximar la serie por una truncación

$$S_N f(x) = \sum_{|k|\leq N} \hat{f}(k) e^{2\pi ikx},$$

y el error será relativamente pequeño

$$E_N f(x) := |S_N f(x) - f(x)| \leq \sum_{|k|>N} |\hat{f}(k)| \leq \sum_{|k|>N} \frac{C_f}{|k|^2} \leq \frac{C'}{N}.$$

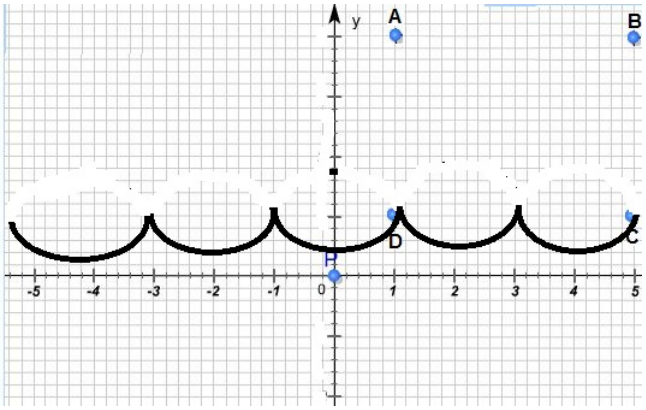


FIGURA 1. Extensión de f de $[0,1]$ a $\mathbb{R}$ para las bases coseno-I

Sin embargo, cuando $f(1^-) \neq f(0)$, es decir, cuando la extensión 1-periódica de f *no* es continua en $x = 0$, entonces los coeficientes $\hat{f}(k)$ pueden decaer de forma muy lenta, y los errores de truncación ser grandes.

1.1. Bases Cosenos *I* y *IV* en $L^2[0,1]$. El efecto no deseado producido por la discontinuidad de f en la frontera se puede reducir utilizando las llamadas *bases de cosenos*. Dada una funcion f en $[0,1]$, consideramos su extensión par

$$\widetilde{f}(x) = \begin{cases} f(-x) & \text{si} \quad x \in [-1,0] \\ f(x) & \text{si} \quad x \in [0,1] \end{cases} \tag{3.1.3}$$

y a continuación su extensión 2-periódica a todo $\mathbb{R}$; ver Figura 2. Si f es continua en $[0,1]$, es claro que $\widetilde{f}(-1) = \widetilde{f}(1)$ y que $\widetilde{f}$ es continua en todo $\mathbb{R}$.

Usando la base trigonométrica en el intervalo $[-1,1]$

$$\left\{1,\ \cos\frac{2\pi kx}{2},\ \operatorname{sen}\frac{2\pi kx}{2}\right\}_{k=1,2,\ldots}$$

podemos escribir $\widetilde{f}$ como:

$$\widetilde{f}(x) = a_0 + \sum_{k=1}^{\infty} a_k \cos(\pi kx) + \sum_{k=1}^{\infty} b_k \operatorname{sen}(\pi kx), \quad -1 \leq x \leq 1, \tag{3.1.4}$$

al menos con convergencia en $L^2(-1,1)$. Como $\tilde{f}$ es par con respecto al origen, los terminos b_k son todos cero (porque $\operatorname{sen}(\pi kx)$ es impar). Además, siendo $\tilde{f}(x) = f(x)$ cuando $x \in [0,1]$, observamos que (3.1.4) quedaría

$$f(x) = a_0 + \sum_{k=1}^{\infty} a_k \cos(\pi kx), \quad 0 \leq x \leq 1,$$

al menos con convergencia en $L^2[0,1]$. A esta expansión se le llama *serie de Fourier coseno de* f.

TEOREMA 3.1.5 (Base coseno I). *La colección* $\{1, \sqrt{2}\cos(k\pi x)\}_{k=1}^{\infty}$ *es una base ortonormal de* $L^2([0,1])$

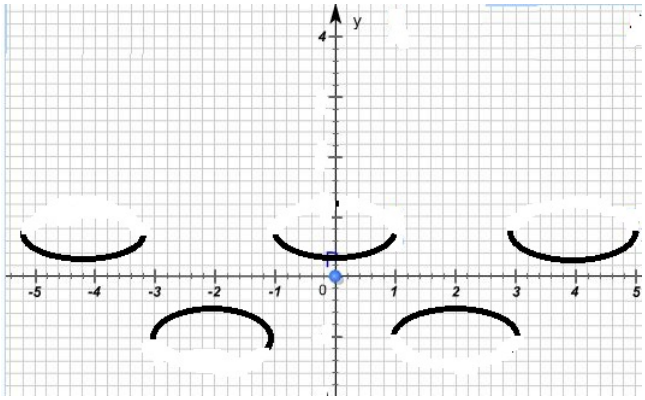


FIGURA 2. Extensión de f de $[0,1]$ a $\mathbb{R}$ para las bases coseno-IV

DEMOSTRACIÓN. Yá sabemos que esta coleción es una base, y sólo necesitamos probar que es ortonormal. Para ello, si $k \geq 1$,

$$\langle \sqrt{2}\cos(k\pi x), \sqrt{2}\cos(k\pi x)\rangle = 2\int_0^1 cos^2(k\pi x)dx = 2\int_0^1 \frac{1+cos(2k\pi x)}{2}dx = 2\times\frac{1}{2} = 1,$$

y si $k \neq l$ acontece:

$$\begin{aligned}\langle \sqrt{2}\cos(k\pi x), \sqrt{2}\cos(l\pi x)\rangle &= 2\int_0^1 cos(k\pi x)cos(l\pi x)dx \\ &= \int_0^1 [cos(k-l)\pi x + cos(k+l)\pi x]dx = 0.\end{aligned}$$

Los cálculos son similares si $k = 0$, usando la función constante 1. □

Otras bases de tipo coseno pueden construirse por este procedimento, pero haciendo diferentes extensiones de f en $\mathbb{R}$. Ahora describimos la llamada *base coseno IV*, que luego aparecerá en algunas aplicaciones numéricas.

Dada una función $f \in L^2[0,1]$, consideramos su extensión, primero simétrica con respecto al origen y después antisimétrica con respecto a 1 y -1, definida por:

$$\tilde{f}(x) = \begin{cases} f(x) \text{ si } x \in [0,1] \\ f(-x) \text{ si } x \in [-1,0] \\ -f(2-x) \text{ si } x \in (1,2] \\ -f(2+x) \text{ si } x \in (-2,-1] \end{cases}$$

que finalmente extendemos de forma 4-periódica a todo $\mathbb{R}$; ver Figura 2.

Si $f(1) \neq 0$, $\tilde{f}$ es discontinua en los enteros impares, por tanto esta extensión es menos regular que la asociada a la base coseno I. Como $\tilde{f}$ tiene periodo 4, se puede escribir como serie de Fourier en la base trigonométrica de $[-2,2]$

$$\left\{1, \cos\frac{2\pi kx}{4}, \text{sen}\,\frac{2\pi kx}{4}\right\}_{k=1}^{\infty}$$

es decir tenemos

$$\tilde{f}(x) = a_0 + \sum_{k=1}^{\infty} a_k cos\frac{\pi kx}{2} + \sum_{k=1}^{\infty} b_k sen\frac{\pi kx}{2}, \quad -2 \leq x \leq 2, \tag{3.1.6}$$

con convergencia en $L^2[-2,2]$. Como $\tilde{f}$ es par, $b_k = 0$. Además, la antisimetria de $\tilde{f}$ en 1 y -1 produce

$$\begin{aligned} a_0 &= \frac{1}{4}\int_{-2}^{2} \tilde{f}(x)dx = \frac{1}{2}\int_{0}^{2} \tilde{f}(x)dx \\ &= \frac{1}{2}\Big[\int_0^1 f(x)dx + \int_1^2 -f(2-x)dx\Big] \\ &= \frac{1}{2}\Big[\int_0^1 f(y)dy - \int_0^1 f(y)dy\Big] = 0 \end{aligned}$$

y de forma similar

$$\begin{aligned} a_{2k} &= \frac{1}{2}\int_{-2}^{2} \tilde{f}(x)\cos\frac{2\pi(2k)x}{4}dx = \int_0^2 \tilde{f}(x)\cos(\pi kx)dx \\ &= \int_0^1 f(x)\cos(\pi kx)dx - \int_1^2 f(2-x)\cos(\pi kx)dx \\ &= \int_0^1 f(y)\cos(\pi ky)dy - \int_0^1 f(y)\cos(\pi ky)dy = 0. \end{aligned}$$

Por tanto de (3.1.6), restringido a $[0,1]$, deducimos

$$f(x) = \sum_{k=0}^{\infty} a_{2k+1}\cos\pi(k+\tfrac{1}{2})x, \quad 0 \le x \le 1, \tag{3.1.7}$$

con convergencia en $L^2[0,1]$.

Teorema 3.1.8 (Base coseno IV). *La colección*

$$\big\{u_k(x) = \sqrt{2}\cos\pi(k+\tfrac{1}{2})x\big\}_{k=0}^{\infty} \tag{3.1.9}$$

es una base ortonormal de $L^2[0,1]$.

Demostración. Como antes, basta probar la ortonormalidad.

- si $k = l$,

$$\begin{aligned} \langle u_k, u_k\rangle &= 2\int_0^1 \cos^2(\pi(k+\tfrac{1}{2})x)dx = 2\int_0^1 \frac{1+\cos(2k+1)\pi x}{2}dx \\ &= \Big[x + \frac{\operatorname{sen}(2k+1)\pi x}{(2k+1)\pi}\Big]_0^1 = 1 \end{aligned}$$

- si $k \neq l$,

$$\begin{aligned} \langle u_k, u_l\rangle &= 2\int_0^1 \cos(\pi(k+\tfrac{1}{2})x)\cos(\pi(l+\tfrac{1}{2})x)dx \\ &= \int_0^1 [\cos(k+l+1)\pi x + \cos(k-l)\pi x]dx \\ &= \Big[\frac{\operatorname{sen}(k+l+1)\pi x}{(k+l+1)\pi} + \frac{\operatorname{sen}(k-l)\pi x}{(k-l)\pi}\Big]_0^1 = 0. \end{aligned}$$

□

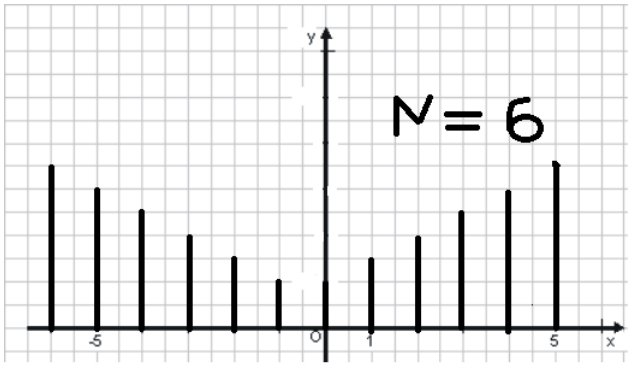


FIGURA 3. Extensión par de una señal con $N = 6$ para la base coseno I discreta

2. Bases de cosenos discretas I y IV

Sea F_N el espacio de señales $\mathbf{f} = (f[n])_{0\le n<N}$ de tamaño N. Por el Teorema 2.2.5 una base ortonormal de F_N viene dada por la familia de vectores

$$\left\{\mathbf{e}_k^{(N)} = \frac{1}{\sqrt{N}}(e^{\frac{2\pi ikn}{N}})_{0\le n<N}\right\}_{k=0}^{N-1} \tag{3.2.1}$$

En está sección construimos bases coseno discretas por procedimentos parecidos a los usados anteriormente.

2.1. Base coseno discreta I. El señal discreta $\mathbf{f} = (f[n])_{0\le n<N} \in F_N$ se extiende por simetría con respecto a $-\frac{1}{2}$ a una señal $\tilde{\mathbf{f}} \in F_{2N}$ dada por:

$$\tilde{f}[n] = \begin{cases} f[n] & \text{si} \quad 0 \le n \le N-1 \\ f[-n-1] & \text{si} \quad -N \le n \le -1 \end{cases}$$

Ver Figura 3.

TEOREMA 3.2.2. *Para el espacio de señales F_{2N} con tamaño $2N$, la colección*

$$\left\{u_k^{(2N)} = \left(\tfrac{1}{\sqrt{2N}} e^{\frac{k\pi i}{N}(n+\frac{1}{2})}\right)_{n=-N}^{N-1}\right\}_{k=-N}^{N-1} \tag{3.2.3}$$

es una base ortonormal.

DEMOSTRACIÓN. Usando la fórmula de la suma de una progresión geométrica tenemos

- si $k \neq l$,

$$\begin{aligned} \langle u_k^{(2N)}, u_l^{(2N)} \rangle &= \frac{1}{2N} \sum_{n=-N}^{N-1} e^{\frac{k\pi i}{N}(n+\frac{1}{2})} e^{-\frac{l\pi i}{N}(n+\frac{1}{2})} = \frac{1}{2N} \sum_{n=-N}^{N-1} e^{\frac{(k-l)\pi i}{N}(n+\frac{1}{2})} \\ &= \frac{1}{2N} \frac{e^{\frac{(k-l)\pi i(N+\frac{1}{2})}{N}} - e^{\frac{(k-l)\pi i(-N+\frac{1}{2})}{N}}}{e^{\frac{(k-l)\pi i}{N}} - 1} \\ &= \frac{(-1)^{k-l}}{2N} \frac{e^{(k-l)\frac{\pi i}{2N}} - e^{(k-l)\frac{\pi i}{2N}}}{e^{\frac{(k-l)\pi i}{N}} - 1} = 0, \end{aligned}$$

ya que el denominador no se anula, pues $0 < |k-l| < 2N$.

- si $k = l$,

$$\langle u_k^{(2N)}, u_k^{(2N)} \rangle = \frac{1}{2N} \sum_{n=-N}^{N-1} e^0 = \frac{1}{2N} 2N = 1$$

□

Por tanto, con esta prueba si $\mathbf{g} \in F_{2N}$ entonces

$$g[n] = \sum_{k=-N}^{N-1} \lambda_k u_k^{(2N)}[n] \quad , \quad n = -N, ..., N-1 \tag{3.2.4}$$

Usando $e^{i\theta} = \cos\theta + i \operatorname{sen}\theta$ deducimos que cualquier $g \in F_{2N}$ puede ser escrita como combinación linear de elementos de la colección:

(3.2.5)

$$\left\{ \mathbf{C}_k^{(2N)} = \left(\cos \frac{k\pi}{N}(n+\frac{1}{2}) \right)_{n=-N}^{N-1} \right\}_{k=0}^{N-1} \bigcup \left\{ \mathbf{S}_k^{(2N)} = \left(\sin \frac{k\pi}{N}(n+\frac{1}{2}) \right)_{n=-N}^{N-1} \right\}_{k=1}^{N}.$$

Como $\tilde{\mathbf{f}} \in F_{2N}$ deducimos

$$\tilde{f}[n] = \sum_{k=0}^{N-1} a_k \mathbf{C}_k^{(2N)}[n] + \sum_{k=1}^{N} b_k \mathbf{S}_k^{(2N)}[n], \quad \text{para} \quad -N \leq n \leq N-1. \tag{3.2.6}$$

Las señales $(\mathbf{S}_k^{(2N)}[n])_{n=-N}^{N-1}$ son antisimétricas con respecto a $-\frac{1}{2}$ porque

$$\mathbf{S}_k^{(2N)}[-1-n] = \operatorname{sen} \frac{k\pi}{N}(-1-n+\frac{1}{2}) = \operatorname{sen} \frac{k\pi}{N}(-n-\frac{1}{2}) = -\mathbf{S}_k^{(2N)}[n],$$

por tanto los elementos en $b_k = 0$, $k = 1, 2, ..., N$. Por tanto, restringiendo en (3.2.6) a $0 \leq n < N$ tenemos

$$f[n] = \sum_{k=0}^{N-1} a_k \mathbf{C}_k^{(2N)}[n], \quad 0 \leq n < N.$$

Teorema 3.2.7 (Base Coseno Discreta *I* (DC-I)). *La colección*

$$\left\{ \lambda_k \sqrt{\frac{2}{N}} \mathbf{C}_k^{(2N)} = \lambda_k \sqrt{\frac{2}{N}} \left(\cos \frac{k\pi}{N}(n+\frac{1}{2}) \right)_{0 \leq n < N} \right\}_{k=0}^{N-1}$$

con $\lambda_0 = \frac{1}{\sqrt{2}}$ *y* $\lambda_k = 1$ *si* $1 \leq k < N$*, es una base ortonormal de* F_N*.*

Demostración. Ya sabemos que esa colección es una base, ahora probemos que es ortonormal. Usando la fórmula

$$\sum_{n=0}^{N-1} \cos\left(\frac{\pi k}{N}(n+\frac{1}{2}) \right) = 0, \quad k = 1, 2, ..., 2N-1 \tag{3.2.8}$$

sigue que $\mathbf{C}_0^{(2N)} \perp \mathbf{C}_k^{(2N)}$, $k = 1, 2, ..., N-1$. La fórmula (3.2.8) se puede probar como sigue:

$$\begin{aligned}\sum_{n=0}^{N-1} \cos(\frac{\pi k}{N}(n+\frac{1}{2})) &= \Re\left[\sum_{n=0}^{N-1} e^{\frac{\pi k i}{N}(n+\frac{1}{2})}\right] = \Re\left[e^{\frac{\pi k i}{2N}} \frac{e^{\frac{\pi k i N}{N}}-1}{e^{\frac{\pi k i}{N}}-1}\right] \\ &= \Re\left[\frac{(-1)^k-1}{2i\sin\frac{k\pi}{2N}}\right] = 0\end{aligned}$$

Para mostrar que $\mathbf{C}_k^{(2N)} \perp \mathbf{C}_l^{(2N)}$, si $1 \le l < k \le N-1$, usamos $2\cos\alpha\cos\beta = \cos(\alpha+\beta) + \cos(\alpha-\beta)$ y la fórmula anterior (3.2.8):

$$\begin{aligned}\langle \mathbf{C}_k^{(2N)}, \mathbf{C}_l^{(2n)} \rangle &= \sum_{n=0}^{N-1} \cos\frac{k\pi}{N}(n+\frac{1}{2}) \cos\frac{l\pi}{N}(n+\frac{1}{2}) \\ &= \frac{1}{2}\sum_{n=0}^{N-1}\left[\cos\frac{(k+l)\pi}{N}(n+\frac{1}{2}) + \cos\frac{(k-l)\pi}{N}(n+\frac{1}{2})\right] = 0,\end{aligned}$$

yá que $0 < k-l < N$ y $0 < k+l < 2N$ por lo cual (3.2.8) es valido.

Por último, es claro que $\|\mathbf{C}_0^{(2N)}\|^2 = N$, por lo que $\|\lambda_0\sqrt{\frac{2}{N}}\mathbf{C}_0^{(2N)}\| = 1$. Si $1 \le k < N$, entonces usando $2\cos^2\alpha = 1 + \cos 2\alpha$ y (3.2.8)

$$\begin{aligned}\langle \mathbf{C}_k^{(2N)}, \mathbf{C}_k^{(2N)} \rangle &= \sum_{n=0}^{N-1} \cos^2\frac{k\pi}{N}(n+\frac{1}{2}) = \frac{1}{2}\sum_{n=0}^{N-1}\left(1+\cos\frac{2k\pi}{N}(n+\frac{1}{2})\right) \\ &= \frac{1}{2}\sum_{n=0}^{N-1} 1 + \frac{1}{2}\sum_{n=0}^{N-1}\cos\frac{2k\pi}{N}(n+\frac{1}{2}) = \frac{N}{2},\end{aligned}$$

de donde $\|\sqrt{\frac{2}{N}}\mathbf{C}_k^{(2N)}\| = 1$. □

Usando bases coseno discretas DC-I cualquier $\mathbf{f} = (f[n])_{0\le n<N} \in F_N$ puede ser escrita como

$$f[n] = \sqrt{\tfrac{2}{N}}\sum_{k=0}^{N-1} \hat{f}_I[k]\lambda_k \cos[\frac{k\pi}{N}(n+\frac{1}{2})],\ n = 0, 1, ..., N-1, \tag{3.2.9}$$

donde los coeficientes vienen dados por la fórmula:

$$\hat{f}_I[k] = \langle \mathbf{f}, \sqrt{\tfrac{2}{N}}\lambda_k \mathbf{C}_k^{(2N)} \rangle = \lambda_k\sqrt{\tfrac{2}{N}}\sum_{n=0}^{N-1} f[n]\cos[\frac{k\pi}{N}(n+\frac{1}{2})], \quad 0 \le k < N. \tag{3.2.10}$$

El vector $(\hat{f}_I[k])_{0\le k<N}$ denomina *transformada coseno discreta-I de* $\mathbf{f}$, y se suele denotar por $\mathrm{DCT}_I(\mathbf{f})$.

Al igual que ocurría con la TFD, para obtener DCT_I con (3.2.10), el número de operaciones requeridas es $O(N^2)$, porque hay N coeficientes $\hat{f}_I[k]$ y cada uno necesita $2N$ operaciones. En las secciones siguientes mostraremos un algoritmo rápido que permite compilar DCT_I con un número de operaciones del orden $O(N\log_2 N)$.

La fórmula (3.2.9) sugiere definir una *Transformada Coseno-I Inversa* mediante

$$\text{IDCT}_I(\mathbf{g})[n] = \sqrt{\tfrac{2}{N}} \sum_{k=0}^{N-1} g[k]\lambda_k \cos[\frac{k\pi}{N}(n+\frac{1}{2})], \quad 0 \le n < N, \tag{3.2.11}$$

de modo que se tiene la *fórmula de inversión*:

$$\mathbf{f} = \text{IDCT}_I\big[\text{DCT}_I(\mathbf{f})\big], \quad \forall\, \mathbf{f} \in F_N. \tag{3.2.12}$$

Ambas fórmulas (3.2.10) y (3.2.11) se pueden escribir en forma matricial con una misma matriz:

$$\mathbf{C}_I = \sqrt{\tfrac{2}{N}} \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \cdots & \frac{1}{\sqrt{2}} \\ \cos\frac{\pi}{2N} & \cos\frac{3\pi}{2N} & \cdots & \cos\frac{(2N-1)\pi}{2N} \\ \vdots & \vdots & \vdots & \vdots \\ \cos\frac{(N-1)\pi}{2N} & \cos\frac{3(N-1)\pi}{2N} & \cdots & \cos\frac{(N-1)\pi(2N-1)}{2} \end{bmatrix} \tag{3.2.13}$$

Así, mirando $\mathbf{f}$ y $\mathbf{g}$ como vectores columna, tenemos:

$$\text{DCT}_I(\mathbf{f}) = \mathbf{C}_I\,\mathbf{f}, \quad \text{y} \quad \text{IDCT}_I(\mathbf{g}) = \mathbf{C}_I^{\text{t}}\,\mathbf{g},$$

donde $\mathbf{C}_I^{\text{t}}$ es la matriz traspuesta de $\mathbf{C}_I$. Además, (3.2.12) nos dice que $\mathbf{C}_I^{\text{t}}\mathbf{C}_I = I$, y por lo tanto $\mathbf{C}_I$ es una matriz ortogonal de tamaño $N \times N$.

2.2. Base coseno discreta IV (DC-IV). Un procedimento parecido al usado en la construción de la Base Coseno IV en $L^2[0,1]$, puede ser usado aquí para obtener la Base Coseno Discreta IV (DC-IV) para señales de tamaño N. Para ello, dada una señal $\mathbf{f} = (f[n])_{n=0}^{N-1} \in F_N$ consideramos su extensión $\tilde{\mathbf{f}}$ de tamaño $4N$ tal que $\tilde{\mathbf{f}}$ sea simétrica con respecto a $-\frac{1}{2}$ y antisimétrica con respecto a $N-\frac{1}{2}$ y $-N-\frac{1}{2}$, es decir

$$\tilde{f}[n] = \begin{cases} f[n] \quad si \quad 0 \le n \le N-1 \\ f[-1-n] \quad si \quad -N \le n \le -1 \\ -f[2N-1-n] \quad si \quad N \le n \le 2N-1 \\ -f[2N+n] \quad si \quad -2N \le n < -N \end{cases} \tag{3.2.14}$$

Ver Figura 4. El Teorema 3.2.2, con $2N$ en lugar de N, nos da el siguiente resultado.

Teorema 3.2.15. *El siguiente sistema es una base ortonormal en F_{4N}:*

$$\left\{ u_k^{(4N)} := \left(\frac{1}{\sqrt{4N}} e^{\frac{k\pi i}{2N}(n+\frac{1}{2})} \right)_{n=-2N}^{2N-1} \right\}_{k=-2N}^{2N-1} \tag{3.2.16}$$

Por tanto, razonando con partes reales e imaginarias, como en (3.2.4) y (3.2.5) (con $2N$ en lugar de N) podemos escribir $\tilde{\mathbf{f}} \in F_{4N}$ como

$$\tilde{f}[n] = \sum_{k=0}^{2N-1} a_k.\mathbf{C}_k^{(4N)}[n] + \sum_{k=1}^{2N} b_k.\mathbf{S}_k^{(4N)}[n], \quad -2N \le n < 2N. \tag{3.2.17}$$

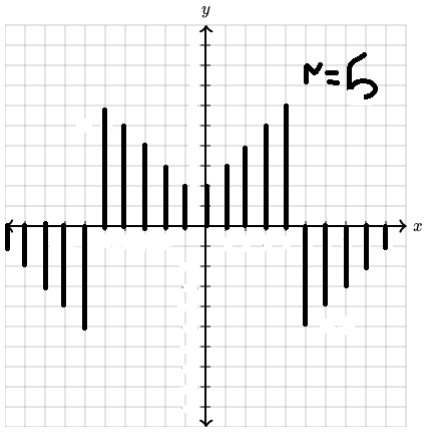


FIGURA 4. Extensión de una señal con $N = 5$ simétrica en $-\frac{1}{2}$ y antisimétrica en $\pm N - \frac{1}{2}$.

Como antes, $b_k = 0$ por ser $\tilde{\mathbf{f}}$ simétrica y $\mathbf{S}_k^{(4N)}$ antisimétricas con respecto a $-\frac{1}{2}$. Además, también tenemos $a_{2k} = 0$ por ser $\tilde{\mathbf{f}}$ antisimétrica y $\mathbf{C}_{2k}^{(4N)}$ simétrica respecto a $N - \frac{1}{2}$. En efecto,

$$\mathbf{C}_{2k}^{(4N)}[2N-1-n] = \cos\big(\frac{\pi k}{N}(2N-n-\tfrac{1}{2})\big) = \cos\big(\frac{\pi k}{N}(n+\tfrac{1}{2})\big) = \mathbf{C}_{2k}^{(4N)}[n], \quad 0 \le n < N.$$

Por tanto, restringiendo (3.2.17) a $0 \le n < N-1$ obtenemos

$$f[n] = \sum_{k=0}^{N-1} a_{2k+1}\mathbf{C}_{2k+1}^{(4N)}[n], \quad 0 \le n < N.$$

Veamos que estos vectores forman una base ortonormal de F_N.

TEOREMA 3.2.18 (Base Coseno IV Discreta, DC-IV). *El sistema*

$$\left\{\sqrt{\tfrac{2}{N}}\mathbf{C}_{2k+1}^{(4N)} = \sqrt{\tfrac{2}{N}}\left(\cos\frac{\pi}{N}(k+\frac{1}{2})(n+\frac{1}{2})\right)_{n=0}^{N-1}\right\}$$

es una base ortonormal en F_N.

DEMOSTRACIÓN. Para $0 \le l < k \le N-1$, tenemos

- si $k \ne l$

$$\begin{aligned}\langle \mathbf{C}_{2k+1}^{(4N)}, \mathbf{C}_{2l=1}^{(4N)}\rangle &= \sum_{n=0}^{N-1}\cos\left[\frac{\pi}{N}(k+\frac{1}{2})(n+\frac{1}{2})\right]\cos\left[\frac{\pi}{N}(l+\frac{1}{2})(n+\frac{1}{2})\right] \\ &= \frac{1}{2}\sum_{n=0}^{N-1}\left[\cos\frac{\pi}{N}(k+l+1)(n+\frac{1}{2}) + \cos\frac{\pi}{N}(k-l)(n+\frac{1}{2})\right] \\ &= 0,\end{aligned}$$

porque para cada una de las sumas se puede aplicar (3.2.8).

- si $k = l$ $(0 \le k \le N-1)$, tenemos:

$$\begin{aligned}\langle \mathbf{C}_{2k+1}^{(4N)}, \mathbf{C}_{2k+1}^{(4N)} \rangle &= \sum_{n=0}^{N-1} \cos^2 \frac{\pi}{N}(k+\frac{1}{2})(n+\frac{1}{2}) \\ &= \frac{1}{2}\sum_{n=0}^{N-1}\left[1+\cos\frac{\pi}{N}(2k+1)(n+\frac{1}{2})\right] = \frac{N}{2}.\end{aligned}$$

□

El Teorema 3.2.18 permite escribir cualquier $\mathbf{f} = (f[n])_{0\le n<N} \in F_N$ como

$$f[n] = \sqrt{\tfrac{2}{N}}\sum_{k=0}^{N-1} \hat{f}_{IV}[k]\cos\big(\frac{\pi}{N}(k+\frac{1}{2})(n+\frac{1}{2})\big), \quad 0 \le n < N, \tag{3.2.19}$$

donde los coeficientes $\hat{f}_{IV}[k]$ vienen dados por la fórmula:

(3.2.20)

$$\hat{f}_{IV}[k] = \langle f, \sqrt{\tfrac{2}{N}}\mathbf{C}_{2k+1}^{(4N)}\rangle = \sqrt{\tfrac{2}{N}}\sum_{n=0}^{N-1} f[n]\cos\big[\frac{\pi}{N}(k+\frac{1}{2})(n+\frac{1}{2})\big], \quad 0 \le k < N.$$

A esta señal $\hat{\mathbf{f}}_{IV}$ la denominamos *transformada coseno discreta IV* de $\mathbf{f}$, que a veces denotaremos por $\mathrm{DCT}_{IV}(\mathbf{f})$. Como antes, la transformación se puede escribir en forma matricial, usando

$$\begin{bmatrix} \hat{f}_{IV}(0) \\ \hat{f}_{IV}(1) \\ \vdots \\ \hat{f}_{IV}(N-1) \end{bmatrix} = \sqrt{\frac{2}{N}} \begin{bmatrix} \cdots & \cos\frac{\pi}{2N}(n+\frac{1}{2}) & \cdots \\ \cdots & \cos\frac{3\pi}{2N}(n+\frac{1}{2}) & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & \cos\frac{(2N-1)\pi}{N}(n+\frac{1}{2}) & \cdots \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ \vdots \\ f(N-1) \end{bmatrix} \tag{3.2.21}$$

La matriz asociada se suele denotar $\mathbf{C}_{IV}$, y como antes es una matriz ortogonal de tamaño $N \times N$.

3. Transformada Coseno Discreta: Algoritmos Rápidos

En esta sección describimos algoritmos rápidos para el cálculo de las transformadas coseno, que son similares a los usados para la FFT.

3.1. Algoritmos Rápidos para DCT_I. Fijamos $N = 2^l$. Para $\mathbf{f} = (f[n])_{0\le n<N}$ en F_N, denotamos sus transformadas coseno por

$$\mathrm{DCT}_I^N f[k] = \lambda_k\sqrt{\tfrac{2}{N}}\sum_{n=0}^{N-1} f[n]\cos[\frac{k\pi}{N}(n+\frac{1}{2})] \tag{3.3.1}$$

$$\mathrm{DCT}_{IV}^N f[k] = \sqrt{\tfrac{2}{N}}\sum_{n=0}^{N-1} f[n]\cos\left[\frac{\pi}{N}(k+\frac{1}{2})(n+\frac{1}{2})\right] \tag{3.3.2}$$

donde $\lambda_0 = 1/\sqrt{2}$ y $\lambda_k = 1$ si $1 \le k < N$. Estas son las fórmulas para la transformada coseno que usa el programa informático Matlab. Como hicimos con la FFT,

separamos los sumandos

$$\mathrm{DCT}_I^N f[k] = \lambda_k \sqrt{\tfrac{2}{N}} \Big[\sum_{0\le n<N/2} \ldots + \sum_{N/2\le n<N} \ldots \Big].$$

Cambiando variables $n = N-1-m$, el segundo sumando podemos escribirlo como

$$\sum_{0\le m<N/2} f[N-1-m] \cos[\frac{k\pi}{N}(N-m-\frac{1}{2})]$$

y como $\cos[\frac{k\pi}{N}(N-m-\frac{1}{2})] = (-1)^k \cos[\frac{k\pi}{N}(m+\frac{1}{2})]$, tenemos

$$\mathrm{DCT}_I^N f[k] = \lambda_k \sqrt{\tfrac{2}{N}} \sum_{0\le n<N/2} \big(f[n] + (-1)^k f[N-1-n]\big) \cos[\frac{k\pi}{N}(n+\frac{1}{2})].$$

Si definimos las señales de $F_{N/2}$

$$f_p[n] = f[n] + f[N-1-n], \quad \text{y} \quad f_i[n] = f[n] - f[N-1-n], \quad 0 \le n < N/2,$$

entonces vemos que, para $\quad 0 \le k < N/2$,

$$\begin{aligned} \mathrm{DCT}_I^N f[2k] &= \tfrac{1}{\sqrt{2}} \mathrm{DCT}_I^{N/2} f_p[k] \\ \mathrm{DCT}_I^N f[2k+1] &= \tfrac{1}{\sqrt{2}} \mathrm{DCT}_{IV}^{N/2} f_i[k]. \end{aligned}$$

Por tanto, si $A_I(N)$ y $A_{IV}(N)$ denotan respectivamente el número de operaciones para calcular DCT_I^N y DCT_{IV}^N, entonces tendremos

$$A_I(N) = A_I(N/2) + N/2 + A_{IV}(N/2) + N/2 \tag{3.3.3}$$

Por tanto, si conociéramos un algoritmo rápido para DCT_{IV}^N, digamos con

$$A_{IV}(N) \le \alpha N \log_2 N + \beta N, \tag{3.3.4}$$

entonces iterando en (3.3.3) tendríamos

$$A_I(N) = \sum_{j=1}^{l} \big[A_{IV}(N/2^j) + 2N/2^j\big] \le \alpha N \log_2 N + (\beta+2)N.$$

Por último, hay varios algoritmos que permiten probar (3.3.4), incluso con el valor $\alpha = \kappa/2$, donde κ es la constante para la FFT; ver Teorema 2.4.3. Por ejemplo, con cálculos similares no es difícil ver que, para $0 \le k < N/2$,

$$\mathrm{DCT}_{IV}^N f[2k] = \Re\Big[e^{-\frac{i\pi k}{N}} \hat{g}[k]\Big], \tag{3.3.5}$$

$$\mathrm{DCT}_{IV}^N f[N-2k-1] = -\Im\Big[e^{-\frac{i\pi k}{N}} \hat{g}[k]\Big] \tag{3.3.6}$$

donde $g[n] = e^{-i(n+\frac{1}{4})\frac{\pi}{N}} \big(f[2n] + i f[N-2n-1]\big)$, $0 \le n < N/2$, es una señal en $F_{N/2}$ (los detalles se pueden consultar en [**3**, §8.3.4]). Por tanto necesitaremos $\kappa\frac{N}{2}\log_2 N$ operaciones complejas para calcular $\hat{g}$ (con FFT), más $5N/2$ operaciones complejas para construir g y calcular (3.3.5) y (3.3.6).

Capítulo 4

Codificación y cuantización

La compresión es necesaria para almacenar y transmitir señales utilizando el menor espacio posible de memoria. Los algoritmos de compresión suelen constar de varios pasos. Por un lado, los denominados pasos de *compresión con pérdida* realizan procesos irreversibles de aproximación o redondeo en la señal, introduciendo una inevitable pérdida de información.

Por otro lado, los pasos de *compresión sin pérdida* buscan la forma de "codificar" la información, asignando a cada símbolo un código binario que minimice el número de bits necesarios para almacenar la señal. Este proceso es reversible, y si se realiza de forma eficiente puede implicar compresiones en torno al 15 % (como en los algoritmos zip, png, etc...).

En este capítulo explicaremos las técnicas habituales de codificación en la compresión sin pérdida, así como algunas técnicas de redondeo (o cuantización) de la compresión con pérdida. Para ello seguiremos como referencia la sección §10.2 del libro de Mallat [**3**], así como las notas de curso de E. Hernández [**1**].

1. Codificación y Entropía de Shannon

1.1. Alfabetos y códigos binarios. Suponer que tenemos una fuente de información cuyas "palabras" se construyen a partir de un alfabeto finito de símbolos $\mathcal{A} = \{x_1, x_2, ..., x_K\}$. Por ejemplo, los textos literarios, libros, etc... se representan mediante sucesiones consecutivas de letras $\{a, b, \ldots, z\}$ (y algunos símbolos básicos de puntuación). Por otro lado, las imágenes se pueden representar como sucesiones consecutivas de los dígitos $\{0, 1, ..., 255\}$, que representan los tonos de gris de cada píxel.

Fijado un alfabeto $\mathcal{A}$, llamemos X a la variable aleatoria que extrae un símbolo del alfabeto al azar. En un alfabeto dado, puede ocurrir que ciertos símbolos sean más frecuentes que otros, ver por ejemplo la Figura 1.

Para almacenar "palabras" en un ordenador, es necesario reemplazar los símbolos x_k por números binarios ω_k, es decir sucesiones finitas de los bits 0 y 1. El objetivo es buscar un "código", es decir una asignación

$$x_k \mapsto \omega_k \ ,$$

que minimice el número medio de bits por letra $\bar{L}$. Por supuesto, la asignación debe hacerse de manera que a cada palabra $x_{k_1} \ldots x_{k_N}$ le corresponda una *única* sucesión de bits $\omega_{k_1} \ldots \omega_{k_N}$, de modo que a partir de estos podamos reconstruir sin ambigüedad la palabra original. Por ejemplo, los símbolos $\{0, 1, 2, ..., 255\}$ pueden codificarse cada uno de ellos con un número binario de 8 dígitos, que es su representación en base 2. En este caso $\bar{L} = 8$, pero ¿puede conseguirse un código mejor?

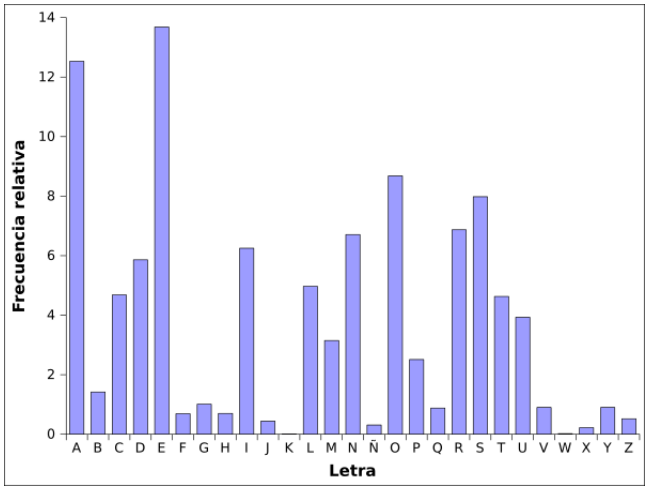


FIGURA 1. Frecuencia de las letras del alfabeto español. *Fuente:* wikipedia

Para formular este problema matemáticamente necesitamos conocer la distribución de probabilidades de cada símbolo

$$p_k = \text{Prob}\,(X = x_k), \quad k = 1, \ldots, K.$$

El objetivo es encontrar un código $\mathcal{C}\colon x_k \mapsto \omega_k$ que minimice

$$\bar{L} = \sum_{k=1}^{K} p_k\, \ell_k, \quad \text{donde } \ell_k = \text{Long}\ \omega_k.$$

EJEMPLO 4.1.1. En el alfabeto AOES solo hay cuatro símbolos. Podríamos usar la siguiente codificación

$$\mathcal{C}_1\colon \quad A = 00, \quad O = 01, \quad E = 10, \quad S = 11,$$

donde todos los números binarios tienen longitud 2. La palabra "aoesiana" AASAEEAO se codificaría

$$0000110010100001,$$

usando un total de 16 bits, es decir 2 bits/letra. Asignemos ahora el código

$$\mathcal{C}_2\colon \quad A = 0, \quad E = 10, \quad S = 110, \quad O = 111.$$

La misma palabra se codificaría ahora como

$$00110010100111,$$

con sólo 14 bits, y un número medio de $\frac{14}{8} = 1,75$ bits/letra. El segundo código parece entonces mejor que el primero, y esto será así si la distribución de letras en el alfabeto AOES cumple

$$p_A = \tfrac{1}{2}, \quad p_E = \tfrac{1}{4}, \quad p_O = p_S = \tfrac{1}{8}, \tag{4.1.2}$$

como ocurre con la palabra anterior. En este caso palabras como SSSSSSSS serían más largas con el código $\mathcal{C}_2$, pero son muy poco frecuentes.

¿Existe un código mejor que el código $\mathcal{C}_2$? Por ejemplo podríamos definir

$$\mathcal{C}_3\colon \quad A = 0, \quad E = 1, \quad O = 10, \quad S = 11, \tag{4.1.3}$$

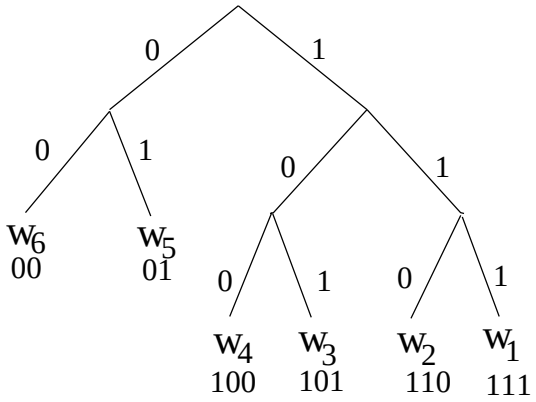


FIGURA 2. Construcción en árbol de un código binario con la condición del prefijo para un alfabeto de 6 símbolos. *Fuente:* Mallat, Fig 10.1

y en este caso la palabra anterior se codifica con sólo 10 bits, y una media de 1'25 bits/letra. Pero este código es ambiguo: la sucesión 101 podría corresponder a EAE o bien a OE, y no podemos descifrar de manera única la palabra. En los códigos $\mathcal{C}_1$ y $\mathcal{C}_2$ no ocurría esto, y las palabras podían recuperarse de manera única sin información adicional.

Para garantizar que toda sucesión $\omega_{k_1}\omega_{k_2}\cdots\omega_{k_N}$ se puede descodificar de manera única, es necesario imponer la siguiente condición:

DEFINICIÓN 4.1.4. **Condición del Prefijo.** Decimos que $\{\omega_1, \ldots, \omega_K\}$ cumple la *condición del prefijo* si ningún número ω_j puede ser el "comienzo" (prefijo) de otro número de la familia ω_k, para $k \neq j$.

Los códigos $\mathcal{C}_1$ y $\mathcal{C}_2$ del ejemplo anterior cumplen la condición del prefijo, pero el código $\mathcal{C}_3$ no la cumple porque ω_2 es un prefijo de los dígitos ω_3 y ω_4.

Una forma práctica de construir códigos con la condición del prefijo es mediante *árboles binarios* de K hojas, cada una de las cuales corresponde a los símbolos $x_1, ..., x_K$. El código binario ω_k asociado a x_k se obtiene seleccionando el camino que une la raíz con la hoja x_k del árbol y escogiendo por orden los dígitos 0 ó 1 según el camino gire a izquierda o derecha; ver Figura 2.

Obsérvese que la longitud l_k de la palabra binaria ω_k es la *profundidad* de la rama correspondiente del árbol binario. Para minimizar $\bar{L}$ es necesario encontrar un *árbol binario óptimo* donde los símbolos más probables correspondan a hojas de los primeros niveles, y los símbolos menos probables a hojas de los últimos niveles.

1.2. La entropía de Shannon. Sea $\mathcal{A} = \{x_1, \ldots, x_K\}$ un alfabeto cuya variable aleatoria X tiene probabilidades

$$p_k = \text{Prob}\,(X = x_k), \quad k = 1, \ldots, K. \tag{4.1.5}$$

Consideramos un código

$$\mathcal{C}: \quad x_k \mapsto \omega_k, \quad \text{con} \quad \ell_k := \text{Long } \omega_k,$$

y estudiamos la función *número medio de bits/letra*:

$$R_X(\mathcal{C}) = \bar{L} = \sum_{k=1}^{K} p_k \, \ell_k.$$

¿Cuál es el valor más pequeño que podría tomar esta función? El teorema de Shannon que veremos abajo muestra que una cota inferior viene dada por la siguiente *función de entropía*

Definición 4.1.6. Dado un par $(\mathcal{A}, X)$ con distribución de probabilidad (4.1.5), se define su *función de entropía* como

$$\mathcal{H}(X) = \sum_{k=1}^{K} p_k \log_2 \frac{1}{p_k}, \tag{4.1.7}$$

con el convenio de que un sumando es cero si $p_k = 0$.

Ejemplo 4.1.8. En el ejemplo AOES tenemos

$$p_A = \tfrac{1}{2}, \quad p_E = \tfrac{1}{4}, \quad p_O = p_S = \tfrac{1}{8}.$$

Entonces

$$\begin{aligned} \mathcal{H}(X) &= \frac{1}{2}\log_2 2 + \frac{1}{4}\log_2 4 + \frac{1}{8}\log_2 8 + \frac{1}{8}\log_2 8 \\ &= \frac{1}{2} + \frac{1}{2} + \frac{3}{8} + \frac{3}{8} = \frac{4+4+3+3}{8} = \frac{14}{8} = 1'75 \end{aligned}$$

Por tanto el código $\mathcal{C}_2$ cumple que $R_X(\mathcal{C}_2) = \mathcal{H}(X)$, y como veremos después, por el Teorema de Shannon 4.1.11, será un código minimal.

La entropía $\mathcal{H}(X)$ mide la "incertidumbre" al seleccionar símbolos de la fuente de información X. En el caso extremo

$$p_1 = 1, \quad p_k = 0, \quad k = 2, 3, ..., K,$$

no hay incertidumbre porque siempre se producirá x_1, y tenemos $\mathcal{H}(X) = 1 \log_2 1 = 0$. En el otro extremo, si

$$p_1 = p_2 = \ldots = p_K = \frac{1}{K}, \quad k = 1, 2, ..., K,$$

entonces la incertidumbre es máxima porque todos los símbolos tienen la misma probabilidad de aparecer, y se tiene

$$\mathcal{H}(X) = \sum_{k=1}^{K} \frac{1}{K} \log_2 K = \log_2 K.$$

El siguiente lema nos garantiza que estos dos casos son en efecto extremales.

Lema 4.1.9. *Para todo par $(\mathcal{A}, X)$ con K símbolos se tiene*

$$0 \leq \mathcal{H}(X) \leq \log_2 K.$$

DEMOSTRACIÓN. Es claro que $0 \leq \mathcal{H}(X)$ ya que $0 \leq p_k \leq 1$. Tenemos pues que maximizar $\mathcal{H}(X) = \sum_{k=1}^{K} p_k \log_2 \frac{1}{p_k}$ con la condición $0 \leq p_k \leq 1$ y $\sum_{k=1}^{K} p_k = 1$. Usando la teoría de los multiplicadores de Lagrange escribimos

$$F(p_1,p_2,...,p_K,\lambda) = \sum_{k=1}^{K} p_k \log_2 \frac{1}{p_k} + \lambda(\sum_{k=1}^{K} p_k - 1).$$

Para buscar los extremos con las restriciones $p_k \geq 0$, $\sum_{k=1}^{K} p_k = 1$, se resuelve el sistema:

$$\begin{cases} \frac{\partial F}{\partial p_k} = \log_2 \frac{1}{p_k} - \frac{1}{\ln 2} + \lambda = 0, \ \ k = 0, 1, ..., K \\ \frac{\partial F}{\partial \lambda} = \sum_{k=1}^{K} p_k - 1 = 0 \end{cases} \tag{4.1.10}$$

De las K primeras ecuaciones se deduce

$$\log_2 \frac{1}{p_k} = \frac{1}{\ln 2} - \lambda \Rightarrow \frac{1}{p_k} = 2^{\frac{1}{\ln 2} - \lambda} = 2^{-\lambda} e$$

por tanto $p_1 = \ldots = p_K = 2^{\lambda}/e$. La restrición $\sum_{k=1}^{K} p_k = 1$ entonces implica que

$$p_1 = \ldots = p_K = 1/K.$$

Este valor da un máximo, porque

$$\frac{\partial^2 F}{\partial p_k \partial p_j}\Big|_{p_k = \frac{1}{K}} = \delta_{jk}(-\frac{K}{\ln 2}),$$

que produce una matriz hessiana negativa. □

A continuación probamos el principal resultado de la sección.

TEOREMA 4.1.11 (Shannon). *Para todo par $(\mathcal{A}, X)$ con probabilidades $\{p_1, \ldots, p_K\}$ y para todo código $\mathcal{C}$ con longitudes $\{\ell_1, \ldots, \ell_K\}$, se tiene*

$$R_X(\mathcal{C}) = \sum_{k=1}^{K} p_k \ell_k \geq \mathcal{H}(X). \tag{4.1.12}$$

Además, siempre existe un código $\mathcal{C}$ con la regla del prefijo tal que

$$R_X(\mathcal{C}) < \mathcal{H}(X) + 1. \tag{4.1.13}$$

Para la demostración del teorema de Shannon necesitamos el siguiente resultado.

LEMA 4.1.14 (Kraft). *Si un código binario $\{\omega_1, \ldots, \omega_K\}$ está formado por dígitos de longitudes $\{\ell_1, \ldots, \ell_K\}$ y cumple la condición del prefijo, entonces*

$$\sum_{j=1}^{L} 2^{-\ell_j} \leq 1. \tag{4.1.15}$$

Recíprocamente, si unos números naturales $\ell_1, \ell_2, ..., \ell_K$ cumplen la desigualdad (4.1.15)*, entonces existe un código binario con la condición del prefijo formado por dígitos de longitudes $\ell_1, \ell_2, ..., \ell_K$.*

DEMOSTRACIÓN. Sea $\mathcal{C}$ un código binario con la condición del prefijo; ver Definición 4.1.4. Sea $\mathcal{T}$ el árbol binario del que proviene, cuyos vértices $\partial\mathcal{T}$ etiquetamos con $\{\omega_j\}_{j=1}^K$, y cuya profundidad es $M = \text{máx}\{\ell_1, ..., \ell_K\}$.

Colgando de cada vértice ω_j, definimos un nuevo árbol $\mathcal{T}_j$ de raíz ω_j y profundidad $M - \ell_j$. Por la condición del prefijo, los subárboles $\mathcal{T}_1, ..., \mathcal{T}_K$ no tienen nodos en cómun, y la unión de todos ellos $\mathcal{T}^* = \mathcal{T} \cup \mathcal{T}_1 \cup \ldots \cup \mathcal{T}_K$ sigue siendo un árbol de profundidad M. Por tanto,

$$\sum_{j=1}^{N} \text{card}(\partial\mathcal{T}_j) \leq \text{card}(\partial\mathcal{T}^*) \leq 2^M.$$

Pero cada subárbol $\mathcal{T}_j$ tiene profundidad $M - \ell_j$ y por tanto $\text{card}(\partial\mathcal{T}_j) = 2^{M-\ell_j}$. Insertándolo en la fórmula anterior resulta:

$$\sum_{j=1}^{K} 2^{M-\ell_j} \leq 2^M \Longrightarrow 2^M \sum_{j=1}^{K} 2^{-\ell_j} \leq 2^M \Longrightarrow \sum_{j=1}^{K} 2^{-\ell_j} \leq 1.$$

Probemos ahora el recíproco. Admitamos que se cumple (4.1.15) y ordenamos las longitudes $\ell_1 \leq \ell_2 \leq ... \leq \ell_K =: M$. Considero el árbol binario completo $\mathcal{T}^*$ de profundidad M . Sean S_1 los $2^{M-\ell_1}$ primeros vértices del nivel M de $\mathcal{T}^*$; sean S_2 los $2^{M-\ell_2}$ vértices siguientes y así sucesivamente. Como

$$\sum_{j=1}^{K} 2^{M-\ell_j} \leq 2^M,$$

los conjuntos $S_1, S_2, ..., S_K$ (que son disjuntos por construcción) tienen juntos menos de 2^M elementos y por tanto pueden elegirse con el algoritmo descrito.

Obsérvese que asociado a cada conjunto S_j se construir hacia arriba el subárbol $\mathcal{T}_j$ de $\mathcal{T}^*$ tal que $\partial\mathcal{T}_j = S_j$ y cuya profundidad es $M - \ell_j$. Llamamos v_j al nodo raíz del subárbol T_j. Entonces el árbol $\mathcal{T} = \mathcal{T}^* \setminus (\mathcal{T}_1 \cup \ldots \cup \mathcal{T}_K)$ tiene como vértices los puntos $v_1, \ldots, v_K$, con profundidades $\ell_1, \ldots, \ell_K$. Por tanto $\mathcal{T}$ genera un código binario $\{\omega_1, \ldots, \omega_K\}$ con la propiedad del prefijo y con Long $\omega_j = \ell_j$. □

Ahora podemos probar el Teorema de Shannon.

DEMOSTRACIÓN. Tenemos que minimizar la función

$$R_X(\ell_1, ..., \ell_K) = \sum_{j=1}^{K} p_j \ell_j$$

en las variables naturales $\ell_1, ..., \ell_K$ sujeto a la condición $\sum_{j=1}^K 2^{-\ell_j} \leq 1$ que da el Lema de Kraft. Vamos a comenzar resolviendo un problema un poco más general. Se trata de minimizar

$$\tilde{R}_X(x_1, ..., x_K) = \sum_{j=1}^{K} p_j x_j$$

en las variables reales $x_1, x_2, ..., x_K > 0$, sujeto a la condición $\sum_{j=1}^{K} 2^{-x_j} = 1 - \varepsilon$, con $0 \leq \varepsilon \leq 1$. Usamos el método del multiplicador de Lagrange. Sea

$$F(x_1, ..., x_K, \lambda) \;\; = \;\; \sum_{j=1}^{K} p_j x_j + \lambda(\sum_{j=1}^{K} 2^{-x_j} - 1 + \varepsilon).$$

Los extremos de $\tilde{R}_X$ deben ser solución de

$$\begin{cases} \frac{\partial F}{\partial x_j} = p_j - (\ln 2)\lambda 2^{-x_j} = 0, \quad j = 0, 1, ..., K \\ \frac{\partial F}{\partial \lambda} = \sum_{k=1}^{K} 2^{-x_j} - 1 + \varepsilon = 0 \end{cases} \tag{4.1.16}$$

De las primeras K ecuaciones se deduce

$$p_j = (\ln 2)\lambda 2^{-x_j} \Longrightarrow 2^{-x_j} = \frac{p_j}{\lambda \ln 2} \tag{4.1.17}$$

Poniendo este resultado en la última de las ecuaciones de (4.1.16) se deduce

$$1 - \varepsilon = \sum_{j=1}^{K} 2^{-x_j} = \sum_{j=1}^{K} \frac{p_j}{\lambda \ln 2} = \frac{1}{\lambda \ln 2} \Longrightarrow \lambda = \frac{1}{(1-\varepsilon)\ln 2}$$

Substituyendo en (4.1.17) se obtiene $2^{-x_j} = (1 - \varepsilon)p_j$ y por tanto

$$-x_j = \log_2[(1-\varepsilon)p_j] \Longrightarrow x_j = \log_2 \frac{1}{(1-\varepsilon)p_j}, \quad j = 1, 2, ..., K.$$

Esta elección corresponde a un mínimo porque $\frac{\partial^2 F}{\partial x_i \partial x_j} = p_j \ln 2 \delta_{ij}$ que da una matriz definida positiva.

Por tanto, para $\varepsilon \in [0, 1]$ fijo, el mínimo de $\tilde{R}_X$ es $\sum_{j=1}^{K} p_j \log_2 \frac{1}{(1-\varepsilon)p_j}$. De todos estos valores de ε, el menor corresponde a $\varepsilon = 0$, y por tanto

$$\tilde{R}_X(x_1, ..., x_K) \geq \sum_{j=1}^{K} p_j \log_2 \frac{1}{p_j}, \quad \forall\, x_1, x_2, ..., x_K > 0 \;\; : \;\; \sum_{j=1}^{K} 2^{-x_j} \leq 1.$$

Esto resuelve el problema cuando tenemos valores reales. Para el caso entero

$$\min_{\sum 2^{-\ell_j} \leq 1} R_X(\ell_1, ..., \ell_K) \geq \min_{\sum 2^{-x_j} \leq 1} \tilde{R}_X(x_1, ..., x_K),$$

porque en la derecha se toma el mínimo sobre un conjunto más grande que en la izquierda. Por tanto

$$R_X(\ell_1, ..., \ell_K) \geq \sum_{j=1}^{K} p_j \log_2 \frac{1}{p_j} = \mathcal{H}(X),$$

lo que demuestra (4.1.12). Para probar que siempre existe un código $\mathcal{C}$ con la propiedad del prefijo que satisface $R_X(\mathcal{C}) < \mathcal{H}(X) + 1$, elegimos los números

$$\ell_j = \left\lceil \log_2 \frac{1}{p_j} \right\rceil, \quad j = 1, 2, ..., K, \tag{4.1.18}$$

donde $\lceil x \rceil$ denota la parte entera por arriba de x, es decir el menor entero mayor o igual que x. Observar que

$$\sum_{j=1}^{K} 2^{-\ell_j} = \sum_{j=1}^{K} 2^{-\left\lceil \log_2 \frac{1}{p_j} \right\rceil} \leq \sum_{j=1}^{K} 2^{-\log_2 \frac{1}{p_j}} = \sum_{j=1}^{K} p_j = 1$$

Entonces el lema de Kraft garantiza que existe un código binario $\{\omega_1, \ldots, \omega_K\}$ con la condición del prefijo. Entonces el código $\mathcal{C}\colon x_k \mapsto \omega_k$ cumple

$$\begin{aligned} R_X(\mathcal{C}) &= \sum_{j=1}^{K} p_j \left\lceil \log_2 \frac{1}{p_j} \right\rceil < \sum_{j=1}^{K} p_j (\log_2 \frac{1}{p_j} + 1) \\ &= \sum_{j=1}^{K} p_j \log_2 \frac{1}{p_j} + \sum_{J=1}^{K} p_j = \mathcal{H}(X) + 1, \end{aligned}$$

porque la parte entera por arriba cumple $\lceil x \rceil < x + 1$. Esto demuestra (4.1.13). □

2. El Código de Huffman

Dado un alfabeto $\mathcal{A}$, denotamos por $\mathfrak{C}(\mathcal{A})$ la familia de todos los códigos $\mathcal{C}\colon x_k \mapsto \omega_k$ con la condición del prefijo. Decimos que $\mathcal{C}_0$ es un *código minimal* para $(\mathcal{A}, X)$ si

$$R_X(\mathcal{C}_0) = \min_{\mathcal{C} \in \mathfrak{C}(\mathcal{A})} R_X(\mathcal{C}) = R_X^{\min}.$$

El teorema de Shannon 4.1.11 nos dice que, si $\mathcal{C}_0$ es un código minimal, entonces

$$\mathcal{H}(X) \leq R_X(\mathcal{C}_0) < \mathcal{H}(X) + 1.$$

El algoritmo utilizado en la demostración, ver (4.1.18), en el caso en que todos los $p_j = 2^{-\kappa_j}$ para ciertos enteros κ_j, nos permite elegir $\ell_j = \kappa_j$ y obtener un código $\mathcal{C}$ con $R_X(\mathcal{C}) = \mathcal{H}(X)$ que por tanto será minimal. Pero en general, $R_X(\mathcal{C})$ no coincidirá con $\mathcal{H}(X)$ porque $\log_2 \frac{1}{p_j}$ no es necesariamente un número entero.

Por tanto se plantea la pregunta de si dado un par $(\mathcal{A}, X)$ existe algún algoritmo constructivo para obtener códigos minimales. O de forma equivalente, si dada una familia de probabilidad $P = \{p_1, \ldots, p_K\}$ existe un K-árbol binario $\mathcal{T}$, es decir con K hojas en la frontera $\{h_1, \ldots, h_K\}$, tal que se minimice la expresión

$$\mathcal{R}_P(\mathcal{T}) = \sum_{j=1}^{K} p_k \ell_k, \quad \text{donde } \ell_k = \text{profundidad}\,(h_k).$$

A tales árboles $\mathcal{T}$ los llamaremos *K-árboles minimales* o *árboles óptimos para P*. Obsérvese que hay a lo sumo una cantidad finita de K-árboles binarios, por lo que siempre existirá alguno que sea minimal.

El problema de encontrar el código óptimo para unas probabilidades dadas fue propuesto por R. Fano a sus estudiantes de un curso de doctorado en el MIT. La oferta era la siguiente: El curso se superaba o bien resolviendo este problema o bien superando el examen final. David Huffman trabajó en el problema hasta una semana antes del examen, cuando decidió abandonarlo y ponerse a estudiar sus notas para el examen final. Unos días después, cuando recogía papeles de su mesa para tirarlos a la papelera le surge la idea que le llevó a resolver el problema.

Es claro que los símbolos con mayor probabilidad deben estar más cerca de la raíz del árbol. La idea de Huffman es comenzar a construir el árbol por sus últimas hojas usando las probabilidades más pequeñas. Comenzamos haciendo algunas observaciones sobre un árbol óptimo $\mathcal{T}$:

1. Cada hoja de profundidad máxima de un árbol óptimo tiene al menos otra hoja hermana en su mismo nivel; si no fuera así podríamos sustituir el árbol por otro con $\mathcal{R}_P$ menor, simplemente quitando esta hoja.
2. Sean h_1 y h_2 dos hojas de un árbol óptimo $\mathcal{T}$ con longitudes $\ell_1 > \ell_2$ respectivamente. Entonces para sus probabilidades se ha de tener $p_1 \leq p_2$. Esto es así, porque si se cumpliera $p_1 > p_2$, a partir del árbol $\mathcal{T}$ podriamos encontrar un árbol $\widetilde{\mathcal{T}}$ intercambiando las etiquetas h_1 y h_2. Entonces

$$\begin{aligned}
\mathcal{R}_P(\mathcal{T}) &= \sum_{j=3}^{K} p_j\ell_j + p_1\ell_1 + p_2\ell_2 \\
&= \sum_{j=3}^{K} p_j\ell_j + p_1\ell_2 + p_2\ell_1 + (p_1\ell_1 + p_2\ell_2 - p_1\ell_2 - p_2\ell_1) \\
&= \mathcal{R}_P(\widetilde{\mathcal{T}}) + p_1(\ell_1 - \ell_2) + p_2(\ell_2 - \ell_1) \\
&= \mathcal{R}_P(\widetilde{\mathcal{T}}) + (p_1 - p_2)(\ell_1 - \ell_2) \; > \; \mathcal{R}_P(\widetilde{\mathcal{T}}),
\end{aligned}$$

   y el árbol $\mathcal{T}$ no sería óptimo.
3. Por tanto, en el último nivel de un árbol óptimo siempre debería haber al menos dos hojas hermanas h_1 y h_2, correspodientes a las probabilidades menores p_1 y p_2.

La clave del algoritmo de Huffman reside en el siguiente lema:

Lema 4.2.1. *Sea $P = \{p_1, p_2, ..., p_K\}$ y supongamos que $p_1 \leq p_2 \leq ... \leq p_K$. Suponer que $\mathcal{T}^*$ es un $(K-1)$-árbol óptimo para el conjunto $P^* = \{p_1+p_2, p_3, ..., p_K\}$. Si la hoja que tiene la probabilidad $p_1 + p_2$ la separamos en dos subhojas de un nivel más cada una con probabilidades p_1 y p_2 entonces se obtiene un K-árbol $\mathcal{T}$ que es óptimo para P.*

Demostración. Observar que $\mathcal{T}$ tiene dos hojas más que $\mathcal{T}^*$ con un nivel de profundidad más, por lo que

$$\mathcal{R}_P(\mathcal{T}) = \mathcal{R}_{P^*}(\mathcal{T}^*) + p_1 + p_2.$$

Sea $\mathcal{S}$ un K-árbol óptimo para P, para el cual podemos suponer que p_1 y p_2 son vecinas en el último nivel (observación 3). Eliminamos estas dos hojas de $\mathcal{S}$ y las unimos en una sola de nivel superior con propabilidad p_1+p_2. Se obtiene así un $(K-1)$-árbol $\mathcal{S}^*$ para las probabilidades P^*. Como $\mathcal{T}^*$ era óptimo para estas probabilidades se tiene que

$$\mathcal{R}_{P^*}(\mathcal{T}^*) \leq \mathcal{R}_{P^*}(\mathcal{S}^*) = \mathcal{R}_P(\mathcal{S}) - p_1 - p_2.$$

Por tanto,

$$\mathcal{R}_P(\mathcal{T}) = \mathcal{R}_{P^*}(\mathcal{T}^*) + p_1 + p_2 \leq \mathcal{R}_P(\mathcal{S}) - p_1 - p_2 + p_1 + p_2 = \mathcal{R}_P(\mathcal{S}).$$

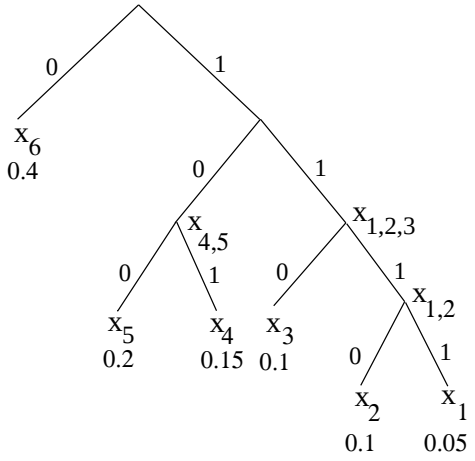


Figure 11.3: Prefix tree grown with the Huffman algorithm for a set of $K = 6$ symbols x_k whose probabilities p_k are indicated at the leaves of the tree.

FIGURA 3. Construcción de un código de Huffman con $K = 6$. *Fuente:* Mallat, Fig 10.3

Como $\mathcal{S}$ era óptimo para P, también tenemos $\mathcal{R}_P(\mathcal{S}) \leq \mathcal{R}_P(\mathcal{T})$, y por tanto

$$\mathcal{R}_P(\mathcal{T}) = \mathcal{R}_P(\mathcal{S}),$$

lo cual implica que $\mathcal{T}$ es un K-árbol óptimo. □

El Lema 4.2.1 da un algoritmo para obtener el código óptimo. Suponer que $P = \{p_1 \leq p_2 \leq \ldots \leq p_K\}$. Construimos un 1-árbol colocando p_1 y p_2 como nodos finales, y al padre común de ambos le asignamos la probabilidad $p_{1,2} = p_1 + p_2$. Reordenamos el conjunto $P^* = \{p_{1,2}, p_3, \ldots, p_K\}$ y construimos otro 1-árbol con las dos menores p_a y p_b. Si $p_{1,2}$ es una de ellas, entonces unimos los árboles anteriores en un 2-árbol por el nodo común, asignando al vértice la probabilidad $p_{a,b} = p_a + p_b$. Si no, dejamos los 1-árboles disjuntos en el mismo nivel. A continuación iteramos el proceso con el $(K-2)$-conjunto de probabilidades restante $P^{**} = \{p_{a,b}, \ldots\}$, y así sucesivamente.

EJEMPLO 4.2.2. Se trata de construir un código de Huffman con $K = 6$ asociado a las probabilidades

$$p_1 = 0,05 \quad p_2 = 0,1 \quad p_3 = 0,1 \quad p_4 = 0,15 \quad p_5 = 0,2 \quad p_6 = 0,4$$

Procediendo como se describe arriba se llega al 6-árbol de la Figura 3. El código binario minimal por tanto será

$$\{\omega_1 = 1111, \quad \omega_2 = 1110, \quad \omega_3 = 110, \quad \omega_4 = 101, \quad \omega_5 = 100, \quad \omega_6 = 0\}$$

La longitud media por símbolo de este código es

$$\bar{L} = 4 \cdot 0'05 + 4 \cdot 0'1 + 3 \cdot 0'1 + 3 \cdot 0'15 + 3 \cdot 0'2 + 1 \cdot 0'4 = 2'35.$$

Por otro lado, la entropía de P es

$$\mathcal{H} = 0'05 \log_2 \tfrac{1}{0'05} + \ldots + 0'4 \log_2 \tfrac{1}{0'4} = 2'284.$$

Por tanto en este caso la longitud media minimal no coincide con la entropía.

3. Cuantización

Supongamos que la fuente de información es una variable aleatoria continua X, que toma valores numéricos reales, en principio arbitrarios, en un intervalo $[a, b]$ (o incluso en todo $\mathbb{R}$). Puesto que los valores de X deben codificarse con un número finito de dígitos binarios es necesario aproximar X por un "cuantizador" $\widetilde{X} = Q(X)$ que sólo toma valores sobre un conjunto finito $\{x_1, \ldots, x_K\}$.

Para hacer la cuantización se divide $[a, b]$ en K intervalos $\{(y_{k-1}, y_k]\}_{1\le k\le K}$, en principio de longitud variable, y se aproxima cada $x \in (y_{k-1}, y_k]$ por un valor x_k fijo, es decir

$$Q(x) = x_k, \quad \forall\, x \in (y_{k-1}, y_k].$$

A los intervalos $(y_{k-1}, y_k]$ se les llama *intervalos de cuantización.*

Ejemplo 4.3.1. *Redondeo al entero más cercano.* En este caso el cuantizador y los intervalos de cuantización se eligen como

$$Q(x) = k, \quad \text{si} \quad x \in (k - \tfrac{1}{2}, k + \tfrac{1}{2}], \quad k \in \mathbb{Z}.$$

3.1. Cuantizadores y distorsión. Sea $p(x)$ la distribuición de probabilidad de la fuente X mirada como variable aleatoria. Decimos que un cuantizador Q tiene *alta resolución* si $p(x)$ es constante en los intervalos de cuantización. En ese caso,

$$p(x) = \frac{p_k}{y_k - y_{k-1}} \quad \forall\, x \in (y_{k-1}, y_k], \quad k = 1, 2, ..., K, \tag{4.3.2}$$

donde $p_k = \int_{y_{k-1}}^{y_k} p(x)\, dx$ cumple $\sum_{k=1}^{K} p_k = 1$. En la práctica, si los intervalos son suficientemente pequeños, $p(x)$ será aproximadamente constante en ellos, y podemos considerar el cuantizador como de alta resolución.

La proposición siguiente nos dice cómo debe elegirse el valor x_k del cuantizador para minimizar el error cuadrático medio o *distorsión*

$$D = \int_a^b (x - Q(x))^2 p(x) dx. \tag{4.3.3}$$

Proposición 4.3.4. *Para un cuantizador de alta resolución, el error cuadrático medio* (4.3.3) *se minimiza cuando*

$$x_k = \frac{y_k + y_{k-1}}{2}, \quad k = 1, \ldots, K,$$

y en ese caso la distorsión vale

$$D = \frac{1}{12} \sum_{k=1}^{K} p_k (y_k - y_{k-1})^2. \tag{4.3.5}$$

Demostración. Para un cuantizador de alta resolución con $Q(x) = x_k$,

$$D = \sum_{k=1}^{K} \int_{y_{k-1}}^{y_k} (x - x_k)^2 \frac{p_k}{y_k - y_{k-1}} dx = \sum_{k=1}^{K} \frac{p_k}{y_k - y_{k-1}} \int_{y_{k-1}}^{y_k} (x - x_k)^2 dx \tag{4.3.6}$$

Basta pues minimizar cada integral

$$F_k(x_k) = \int_{y_{k-1}}^{y_k} (x - x_k)^2 dx.$$

Los extremos se alcanzarán cuando $F_k'(x_k) = 0$, es decir

$$\int_{y_{k-1}}^{y_k} -2(x - x_k)dx = -[(x - x_k)^2]_{y_{k-1}}^{y_k} = 0.$$

Operando tenemos

$$\begin{aligned} &-(y_k - x_k)^2 + (y_{k-1} - x_k)^2 = 0 \\ &-y_k^2 + 2y_k x_k - x_k^2 + y_{k-1}^2 - 2y_{k-1}x_k + x_k^2 = 0 \\ &y_{k-1}^2 - y_k^2 + 2x_k(y_k - y_{k-1}) = 0 \end{aligned}$$

y por tanto

$$x_k = \frac{y_k + y_{k-1}}{2}.$$

Como $F_k''(x_k) = 2(y_k - y_{k-1}) > 0$, el valor obtenido es un mínimo. Por tanto, la distorsión mínima queda

$$\begin{aligned} D &= \sum_{k=1}^{K} \int_{y_{k-1}}^{y_k} (x - \frac{y_k + y_{k-1}}{2})^2 \frac{p_k}{y_k - y_{k-1}} dx \\ &= \sum_{k=1}^{K} \frac{p_k}{y_k - y_{k-1}} \times \left[\frac{(x - \frac{y_k + y_{k-1}}{2})^3}{3} \right]_{y_{k-1}}^{y_k} \\ &= \frac{1}{3} \sum_{k=1}^{K} \frac{p_k}{y_k - y_{k-1}} [(y_k - \frac{y_k + y_{k-1}}{2})^3 - (y_{k-1} - \frac{y_k + y_{k-1}}{2})^3] \\ &= \frac{1}{3} \sum_{k=1}^{K} \frac{p_k}{y_k - y_{k-1}} [(\frac{y_k - y_{k-1}}{2})^3 - (\frac{y_{k-1} - y_k}{2})^3] \\ &= \frac{1}{3} \sum_{k=1}^{K} \frac{p_k}{y_k - y_{k-1}} \times \frac{2(y_k - y_{k-1})^3}{2^3} = \frac{1}{12} \sum_{k=1}^{K} p_k (y_k - y_{k-1})^2. \end{aligned}$$

□

A partir de ahora supondremos que los valores x_k de los cuantizadores son los dados por la Proposición 4.3.4. Además, diremos que Q es un *cuantizador uniforme* cuando todos los intervalos de cuantización tienen la misma longitud

$$\triangle = y_k - y_{k-1}, \quad \forall\, k = 1, 2, ..., K.$$

Para cuantizadores uniformes la distorsión viene dada por

$$D = \frac{1}{12} \sum_{k=1}^{K} p_k \triangle^2 = \frac{\triangle^2}{12}, \tag{4.3.7}$$

y por tanto no depende de la distribución de probabilidad.

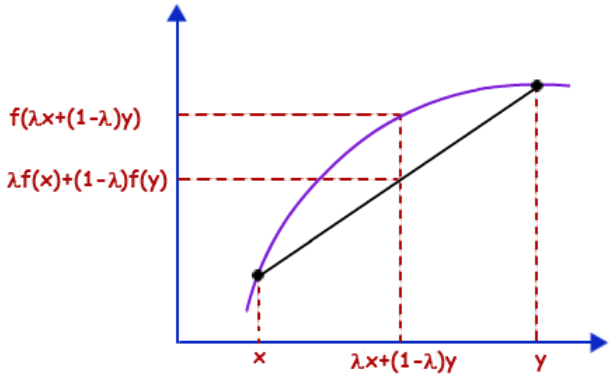


FIGURA 4. Función cóncava

3.2. Cuantizadores que minimizan la entropía. En la práctica, fijada una magnitud D para la distorsión, buscamos un cuantizador $\widetilde{X} = Q(X)$ que nos permita codificar datos minimizando el número de bits/letra. El Teorema de Shannon nos sugiere buscar el cuantizador $\widetilde{X} = Q(X)$ que tenga menor entropía

$$\mathcal{H}(\widetilde{X}) = \sum_{k=1}^{K} p_k \log_2 \frac{1}{p_k}.$$

Para ello definimos un nuevo parámetro, la *entropía diferencial de* X, como

$$\mathcal{H}_d(X) = \int_a^b p(x) \log_2 \frac{1}{p(x)}\, dx.$$

Si $p(x)$ es constante en intervalos, como en (4.3.2), tenemos

$$\mathcal{H}_d(X) = \sum_{k=1}^{K} \frac{p_k}{y_k - y_{k-1}} (\log_2 \frac{y_k - y_{k-1}}{p_k})(y_k - y_{k-1}) = \sum_{k=1}^{K} p_k \log_2 \frac{y_k - y_{k-1}}{p_k}. \tag{4.3.8}$$

PROPOSICIÓN 4.3.9. *Para todo cuantizador* $\widetilde{X} = Q(X)$ *de alta resolución se tiene*

$$\mathcal{H}(\widetilde{X}) \geq \mathcal{H}_d(X) - \frac{1}{2} \log_2(12D), \tag{4.3.10}$$

y la igualdad en (4.3.10) *se cumple si y solo si* Q *es uniforme.*

DEMOSTRACIÓN. Usando (4.3.8) tenemos

$$\begin{aligned} \mathcal{H}_d(X) &= \sum_{k=1}^{K} p_k \log_2 \frac{1}{p_k} + \sum_{k=1}^{K} p_k \log_2 (y_k - y_{k-1}) \\ &= \mathcal{H}(\widetilde{X}) + \frac{1}{2} \sum_{k=1}^{K} p_k \log_2 (y_k - y_{k-1})^2. \end{aligned}$$

Recordemos que una función $\varphi(x)$ es cóncava si para todo $\{a_1, a_2, ..., a_K\}$ y para todo $p_k \geq 0$ con $\sum_{k=1}^K p_k = 1$ se tiene que

$$\sum_{k=1}^{K} p_k \varphi(a_k) \leq \varphi(\sum_{k=1}^{K} p_k a_k), \tag{4.3.11}$$

ver Figura 4. Además, si φ es estritamente cóncava, solo hay igualdad en (4.3.11) cuando $a_k = a$ para todo $k = 1, 2, ..., K$. Como $\varphi(x) = \log_2 x$ es estritamente cóncava se tiene

$$\frac{1}{2}\sum_{k=1}^{K} p_k \log_2(y_k - y_{k-1})^2 \leq \frac{1}{2}\log_2(\sum_{k=1}^{K} p_k (y_k - y_{k-1})^2) = \frac{1}{2}\log_2(12D), \tag{4.3.12}$$

usando la fórmula (4.3.5). Por tanto

$$\mathcal{H}_d(X) \leq \mathcal{H}(\widetilde{X}) + \frac{1}{2}\log_2(12D),$$

de donde se obtiene (4.3.10). Además la igualdad sólo ocurre si $y_k - y_{k-1}$ son todos iguales, es decir, Q es un cuantizador uniforme. □

Corolario 4.3.13. *Fijada una distorsión D, el cuantizador $\widetilde{X} = Q(X)$ de menor entropía es el cuantizador uniforme con $\triangle = \sqrt{12D}$. Además se cumple*

$$\mathcal{H}(\widetilde{X}) = \mathcal{H}_d(X) - \frac{1}{2}\log_2(12D) = \mathcal{H}_d(X) + \log_2 \frac{1}{\triangle}. \tag{4.3.14}$$

Usando el Teorema de Shannon, si codificamos $\widetilde{X}$ con un código minimal $\mathcal{C}_0$ (como el de Huffman), el número de bits/símbolo $R_{\widetilde{X}} = R_{\widetilde{X}}(\mathcal{C}_0)$ cumplirá

$$R_{\widetilde{X}} \geq \mathcal{H}(\widetilde{X}) = \mathcal{H}_d(X) + \log_2 \frac{1}{\triangle}, \qquad \text{y} \quad R_{\widetilde{X}} < \mathcal{H}(\widetilde{X}) + 1. \tag{4.3.15}$$

Alternativamente, usando (4.3.14), tenemos que la distorsión de $\widetilde{X}$ cumple

$$D = \frac{1}{12} 2^{2\mathcal{H}_d(X)} 2^{-2\mathcal{H}(\widetilde{X})} \geq \frac{1}{12} 2^{2\mathcal{H}_d(X)} 2^{-2R_{\widetilde{X}}}, \quad \text{y} \quad D < \frac{1}{12} 2^{2\mathcal{H}_d(X)} 2^{-2(R_{\widetilde{X}}-1)}. \tag{4.3.16}$$

Ejemplo 4.3.17. *Cuantización de X con valores en $[a, b]$.* En la práctica, primero se escoge un entero K suficientemente grande de modo que la función de probabilidad $p(x)$ sea (esencialmente) constante sobre intervalos de longitud $\triangle = (b - a)/K$. Después se define una "función de redondeo"

$$Q_\triangle(x) := a + (k + \tfrac{1}{2})\triangle, \quad \text{si} \quad x \in \big[a + k\triangle, a + (k + 1)\triangle\big), \quad 0 \leq k < K.$$

Entonces $\widetilde{X} = Q_\triangle(X)$ es un cuantizador uniforme de alta definición, con distorsión $\triangle^2/12$ y que codificado con el algoritmo de Huffman tiene un número de bits/símbolo $R_{\widetilde{X}}$ que esencialmente vale $\mathcal{H}_d(X) + \log_2 \frac{1}{\triangle}$.

Ejemplo 4.3.18. *Cuantización de X limitada a R bits/símbolo.* A veces el ordenador, o el canal de transmisión, no permiten usar más de R bits/símbolo. En ese caso, podemos usar el mismo cuantizador Q_Δ que en el ejemplo anterior, pero escogiendo $\triangle = (b - a)/K$ de modo que se cumpla

$$R_{\widetilde{X}} < 1 + \mathcal{H}_d(X) + \log_2 \frac{1}{\triangle} \leq R.$$

Es decir, bastaría con escoger K de modo que $\Delta = (b-a)/K \geq 2^{\mathcal{H}_d(X)-R+1}$ (y suficientemente pequeño para tener alta definición).

3.3. Cuantización de vectores. En algunas aplicaciones (como al cuantizar señales discretas) las fuentes de información serán vectoriales, es decir

$$\mathbf{X} = \sum_{0\leq n<N} X[n]\mathbf{e}_n,$$

donde cada $X[n]$ es una variable aleatoria escalar (que supondremos de media cero) y $\{\mathbf{e}_n\}_{0\leq n<N}$ una base ortonormal. Queremos ahora minimizar el *número medio de bits/símbolo* para *todas* las fuentes de información. Es decir, encontrar cuantizadores de alta resolución $\widetilde{\mathbf{X}} = (\widetilde{X}[n])_{0\leq n<N}$, de modo que minimicen la *entropía media*

$$\overline{\mathcal{H}}(\widetilde{\mathbf{X}}) := \frac{1}{N}\sum_{0\leq n<N} \mathcal{H}(\widetilde{X}[n]).$$

Por la Proposición 4.3.9 los cuantizadores $\widetilde{X}[n]$ pueden suponerse uniformes con tamaños de los intervalos $\triangle_n$. El objetivo ahora es minimizar entre todos los tamaños posibles de los intervalos.

Definimos la *distorsión total* de un cuantizador vectorial $\widetilde{\mathbf{X}} = (\widetilde{X}[n])_{0\leq n<N}$ como

$$D_{\widetilde{\mathbf{X}}} = \mathbb{E}\big[\|\mathbf{X}-\widetilde{\mathbf{X}}\|^2\big] = \sum_{0\leq n<N} \mathbb{E}\big[|X[n]-\widetilde{X}[n]|^2\big] = \sum_{0\leq n<N} D_{\widetilde{X}[n]},$$

y la *entropía diferencial media* de $\mathbf{X}$ como

$$\overline{\mathcal{H}}_d(\mathbf{X}) = \frac{1}{N}\sum_{n=0}^{N-1} \mathcal{H}_d(X[n]).$$

El siguiente resultado generaliza la Proposición 4.3.9.

Teorema 4.3.19. *Entre todos los vectores $\widetilde{\mathbf{X}} = (\widetilde{X}[n])_{0\leq n<N}$ de cuantizadores de alta resolución, con distorsión total fija D, la entropía media $\overline{\mathcal{H}}$ es mínima si son todos uniformes con*

$$\triangle_n^2 = \frac{12D}{N}, \quad 0\leq n<N. \tag{4.3.20}$$

En este caso se tiene

$$D = \frac{N}{12}\, 2^{2\overline{\mathcal{H}}_d(\mathbf{X})} 2^{-2\overline{\mathcal{H}}}. \tag{4.3.21}$$

Demostración. Por la Proposición 4.3.9 sabemos que los $\widetilde{X}[n]$ pueden suponerse uniformes. Además, por (4.3.14)

$$\mathcal{H}(\widetilde{X}[n]) = \mathcal{H}_d(X[n]) - \frac{1}{2}\log_2(12D_{\widetilde{X}[n]}).$$

Tomando promedios,

$$\overline{\mathcal{H}}(\widetilde{\mathbf{X}}) = \overline{\mathcal{H}}_d(\mathbf{X}) - \frac{1}{N}\sum_{n=0}^{N-1}\frac{1}{2}\log_2(12D_{\widetilde{X}[n]}) \tag{4.3.22}$$

Basta pues maximizar la función

$$F(x_1,\dots,x_N)=\frac{1}{N}\sum_{n=1}^{N}\log_2(x_n),$$

con la restricción $\sum_{n=1}^{N}x_n=D$. Usando de nuevo la concavidad de $\varphi(x)=\log_2 x$ y la desigualdad (4.3.11) (con $p_k=1/N$ y $a_k=x_k$) se tiene

$$\frac{1}{N}\sum_{n=1}^{N}\log_2(x_n)\le\log_2(\frac{1}{N}\sum_{n=1}^{N}x_n)=\log_2(D/N),$$

con igualdad si y solo si todos los x_n son iguales. Por tanto debemos tener

$$D_{\widetilde{X}[n]}=D/N,\quad\forall\, n=0,\dots,N-1.$$

Por otro lado, de (4.3.7) sabemos que

$$\triangle_n^2=12D_{\widetilde{X}[n]}=12D/N,$$

lo que prueba (4.3.20). Además, sustituyendo en (4.3.22),

$$\overline{\mathcal{H}}(\widetilde{\mathbf{X}})=\overline{\mathcal{H}}_d(\mathbf{X})-\frac{1}{2}\log_2(12D/N),\tag{4.3.23}$$

de donde se obtiene (4.3.21) despejando la D. □

3.4. Cuantizadores vectoriales ponderados. En algunas aplicaciones, interesa utilizar una función de distorsión que varíe de forma distinta en cada componente $X[n]$. Por ejemplo, cuando las variables aleatorias $X[n]$ tienen distintas desviaciones típicas. Otro ejemplo se da al cuantizar la TFD de una señal de audio o de una imagen, pues la percepción humana es más sensible a los cambios en bajas frecuencias (n pequeños) que en las altas (n grandes).

En estos casos es conveniente considerar una función de *distorsión ponderada*

$$D^w_{\widetilde{\mathbf{X}}}:=\sum_{0\le n<N}\mathbb{E}\Big[\Big|\tfrac{X[n]-\widetilde{X}[n]}{w_n}\Big|^2\Big]=\sum_{0\le n<N}\frac{D_n}{w_n^2},\quad\text{donde}\quad D_n=D_{\widetilde{X}[n]},$$

y donde $w=(w_n)_{0\le n<N}$ es un vector de pesos positivos prefijado. ¿Cuál será el cuantizador vectorial óptimo $\widetilde{\mathbf{X}}$ entre los que tienen $D^w_{\widetilde{\mathbf{X}}}=D$ fijo?

Para determinarlo basta aplicar la construcción de la sección anterior a

$$\mathbf{Y}=(Y[n])_{0\le n<N},\quad\text{con}\quad Y[n]:=X[n]/w_n.$$

El Teorema 4.3.19 nos da cuantizadores uniformes $\widetilde{Y}[n]=Q_\Delta(Y[n])$, con intervalos de tamaño común $\Delta=\sqrt{12D_{\widetilde{\mathbf{Y}}}/N}$. Ahora simplemente definimos

$$\widetilde{X}[n]:=w_n\widetilde{Y}[n]=w_nQ_\Delta(X[n]/w_n)=Q_{w_n\Delta}(X[n]),\quad 0\le n<N,\tag{4.3.24}$$

como cuantizadores uniformes de los $X[n]$. Obsérvese que $D_{\widetilde{\mathbf{Y}}}=D^w_{\widetilde{\mathbf{X}}}=D$. Los intervalos asociados a $\widetilde{X}[n]$ tienen ahora tamaños distintos

$$\Delta_n=w_n\Delta=w_n\sqrt{12D/N}.$$

Como las funciones de probabilidad cumplen

$$p_{Y[n]}(x) = w_n p_{X[n]}(w_n x),$$

es fácil ver que $\mathcal{H}(\widetilde{Y}[n]) = \mathcal{H}(\widetilde{Y}[n]) + \log_2(1/w_n)$ (y lo mismo para $\mathcal{H}_d$), de donde se sigue (llamando $\bar{w} = \frac{1}{N}\sum_{0\le n<N} \log_2(1/w_n)$)

$$\begin{aligned} \overline{\mathcal{H}}(\widetilde{\mathbf{X}}) &= \overline{\mathcal{H}}(\widetilde{\mathbf{Y}}) + \bar{w} = \overline{\mathcal{H}}_d(\mathbf{Y}) + \bar{w} - \frac{1}{2}\log_2(12D/N) \\ &= \overline{\mathcal{H}}_d(\mathbf{X}) - \frac{1}{2}\log_2(12D/N). \end{aligned}$$

Corolario 4.3.25. *Fijada una constante D, el cuantizador $\widetilde{\mathbf{X}} = (\widetilde{X}[n])_{0\le n<N}$ definido por*

$$\widetilde{X}[n] = Q_{w_n\Delta}(X[n]), \quad 0 \le n < N, \quad \text{con } \Delta = \sqrt{12D/N}, \tag{4.3.26}$$

minimiza la entropía media de entre todos los cuantizadores que tienen $D^w_{\widetilde{\mathbf{X}}} = D$.

En las aplicaciones a la codificación de la DCT, y en particular en el algoritmo JPEG, se utilizan estos cuantizadores ponderados, escogiendo la familia de pesos $(w_n)_{0\le n<N}$ según un formato universal adecuado (y según la tasa de compresión que se busque). Ver Capítulo 5, sección 3.

Capítulo 5

La codificación en el formato JPEG

JPEG, en inglés *Joint Photographic Experts Group*, es un algoritmo que sirve para comprimir imágenes, ya sean de colores o en escalas de gris. Fue desarrollado originalmente en los años 80 y la primera versión comenzó a ser usada extensamente en 1992.

Aunque a finales de los 90 se hicieron esfuerzos por mejorar JPG80 incorporando técnicas matemáticas más sofisticadas (como las bases de wavelets en lugar de las bases coseno), lo cierto es que por problemas de patentes y de compatibilidad de software, JPG80 sigue siendo el más extendido.

En este capítulo explicaremos con detalle el proceso para el tratamiento y la compresión de una imagen en el formato JPG80. Para ello seguimos como referencia el libro de Mallat [**3**, §10.5], con la adaptación dada en los apuntes [**1**].

1. Preparación de la imagen digital

1.1. El sistema RGB para representar colores. Una imagen digital se representa con una matriz de datos

$$\mathbf{f} = \big(f[m,n]\big)_{0\le m<M, 0\le n<N}$$

donde cada entrada $f[m,n]$ corresponde a la intensidad de color del píxel situado en la posición $[m,n]$. En la práctica M, N son potencias de 2, que suelen variar entre 256 y 2048 (o más en las cámaras más modernas).

En las fotografías en blanco y negro, $f[m,n]$ es un número entero en $\{0,\dots,255\}$, y su valor mide la intensidad de gris del píxel correspondiente (donde 0 es más oscuro y 255 más claro). Este número se almacena con 1 *byte* = 8 bits.

En las fotografías en color, $f[m,n]$ en un vector de $\mathbb{R}^3$, donde cada componente indica la intensidad de un color primario en la escala de 0 a 255 (donde 0 indica que el color no está presente, y 255 su intensidad máxima). En el llamado *sistema RGB* los colores primarios son rojo, verde y azul (Red, Green, Blue, por sus siglas en inglés). La combinación adecuada de estos tres colores permite obtener cualquier color de la paleta, y representar las fotografías con gran fidelidad.

La Figura 1 muestra la paleta de colores en un cubo 3D, donde los vértices corresponden a los colores primerios y sus combinaciones. Los colores en la diagonal del cubo, representados por $(\lambda,\ \lambda,\ \lambda)$ con $0 \le \lambda \le 255$, corresponden a la escala de gris, donde $\lambda = 0$ es negro y $\lambda = 255$ blanco.

1.2. Transformación del espacio de color. Para manipular la imagen, es habitual que el formato RGB sea transformado en otro formato denotado `Y Cb Cr`, similar al usado en los sistemas televisivos como PAL o NTSC.

- `Y` se denomina luminancia o brillo
- `Cb` y `Cr` se denominan saturación, o crominancia, de azules y rojos.

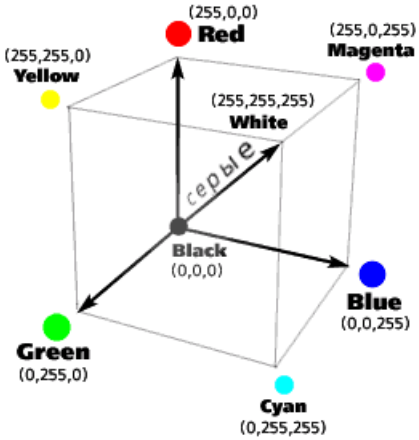


FIGURA 1. Representación espacial de colores. *Fuente:* internet

La ecuación linear para pasar de un sistema a otro es la siguiente :

$$\begin{bmatrix} \texttt{Y} \\ \texttt{Cb} \\ \texttt{Cr} \end{bmatrix} = \begin{bmatrix} 0,299 & 0,587 & 0,114 \\ -0,169 & -0,331 & 0,5 \\ 0,5 & -0,419 & -0,081 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} \tag{5.1.1}$$

Los valores (`Y, Cb, Cr`) obtenidos a partir de los valores (R, G, B) cumplen la relación

$$0 \leq \texttt{Y},\ \ \texttt{Cr},\ \ \texttt{Cb}\ \leq 255,$$

y tras redondearlos se representan con 8 bits cada uno de ellos. En este paso se pierde un poco de información por redondeo, pero esta diferencia es indetectable a ojos humanos. En el proceso de reconstrucción, los componentes (R, G, B) se recuperan invirtiendo la ecuacion linear anterior.

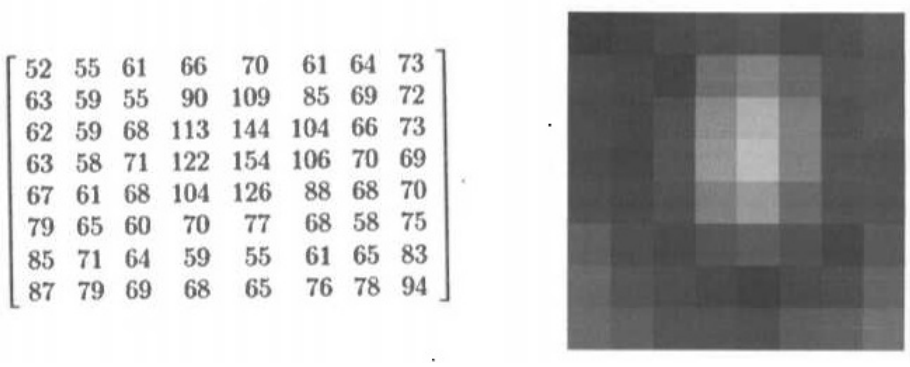
1.3. Reducción de resolución. JPEG suele además aplicar una reducción (downsampling) en los canales `Cr,Cb`. Esto es debido a las cualidades de los receptores en el ojo humano, que son más sensibles al brillo de la imagen `Y`, que a la saturación de colores `Cb` y `Cr`. Los codificadores JPEG aprovechan este hecho para reducir la información de color de la imagen, eliminando una muestra de cada 2 consecutivas en los valores `Cb, Cr` (downsampling). Después se reconstruye por interpolación.

2. Subdivisión en bloques 8×8

A partir de ahora supondremos que la imagen viene dada por una matriz discreta

$$\mathbf{f} = \big(f[n,m]\big)_{0 \leq n < N, 0 \leq m < M}$$

(que puede corresponder a cualquiera de los canales `Y,Cr,Cb` anteriores, o al nivel de gris en una imagen en blanco y negro). Supondremos que M y N son potencias de 2, una restricción habitual para poder aplicar los algoritmos de FFT. Por simplicidad supondremos $M = N = 2^L$, $L \geq 3$.



$$\begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 55 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix}$$

FIGURA 2. Ejemplo de un bloque 8×8 de una imagen y su representación en escala de gris. *Fuente:*[**1**, Fig 2.13]

Para procesar esta señal el primer paso es dividirla en bloques de tamaño 8×8, donde cada uno de ellos será procesado de forma independiente. Esto reduce notablemente la complejidad de los cálculos de FFTs, pero a costa de introducir posibles alteraciones debido al cuadriculado de la imagen. Las bases coseno tratan precisamente de minimizar este efecto. Un ejemplo de bloque 8×8 y su representación en escala de gris viene dado por la figura 2

Para analizar cada bloque 8×8, JPG80 utiliza la transformada coseno discreta en 2 dimensiones, que denotaremos 2D-DCT$_I$. Esta transformada representa la señal $\mathbf{f}$ en términos de una base de cosenos bidimensional, que se obtiene como producto tensorial de las bases coseno-I que vimos en el Teorema 3.2.7. Es decir

$$\mathbf{u}_{k,l} = \frac{\lambda_k \lambda_l}{4} \left(\mathbf{C}_k[n]\mathbf{C}_l[m] = \cos\left[\frac{k\pi}{8}(n+\frac{1}{2})\right] \cos\left[\frac{l\pi}{8}(m+\frac{1}{2})\right] \right)_{n,m=0}^{7}$$

con $0 \leq k,\ l \leq 7$, y $\lambda_0 = 1/\sqrt{2}$ o bien $\lambda_p = 1$ si $1 \leq p \leq 7$. Los 64 coeficientes de la señal transformada, 2D-DCT$_I(\mathbf{f})$, vienen dados por

$$\hat{f}_I[k,l] = \frac{\lambda_k \lambda_l}{4} \sum_{0 \leq n,m < 8} f[n,m] \cos[\tfrac{k\pi}{8}(n+\tfrac{1}{2})] \cos[\tfrac{l\pi}{8}(m+\tfrac{1}{2})], \quad 0 \leq k,\ l \leq 7,$$

y son compilados con un algoritmo rápido (como los vistos en la sección §3.3.1). Con este algoritmo, son necesarias aproximadamente $8^2 \log_2 8^2 = 64 \times 6 = 384$ operaciones para calcular todos los coeficientes $\hat{f}_I[k,l]$, $0 \leq k,l \leq 7$. Para hacer el cálculo para los $\left(2^{L-3}\right)^2$ bloques, el número de operaciones necesarias es:

$$O\left(2^{2(L-3)} 8^2 \log_2 8^2\right) = O\left(2^{2L} \log_2 8^2\right) = O\left(N^2\right).$$

El primer coeficiente de cada bloque, $\hat{f}_I[0,0]$, juega un papel relevante, y suele llamarse *coeficiente DC*. En efecto, su expresión

$$\hat{f}_I[0,0] = \frac{1}{8} \sum_{m=0}^{7} \sum_{n=0}^{7} f[n,m],$$

es el promedio de los niveles de gris de todo el bloque (multiplicado por 8), y suele ser el coeficiente más grande de todos. Al resto de coeficientes $\hat{f}_I[k,l]$ se les suele llamar *coeficientes AC*. Habitualmente, los coeficientes DC no varían mucho de un bloque

$$\begin{bmatrix} -76 & -73 & -67 & -62 & -58 & -67 & -64 & -55 \\ -65 & -69 & -73 & -38 & -19 & -43 & -59 & -56 \\ -66 & -69 & -60 & -15 & 16 & -24 & -62 & -55 \\ -65 & -70 & -57 & -6 & 26 & -22 & -58 & -59 \\ -61 & -67 & -60 & -24 & -2 & -40 & -60 & -58 \\ -49 & -63 & -68 & -58 & -51 & -60 & -70 & -53 \\ -43 & -57 & -64 & -69 & -73 & -67 & -63 & -45 \\ -41 & -49 & -59 & -60 & -63 & -52 & -50 & -34 \end{bmatrix}$$

FIGURA 3. Representación del bloque en la figura 2 después de restar 128 de cada componente. *Fuente:* [**1**, Fig 2.14]

$$\begin{bmatrix} -415.38 & -30.19 & -61.20 & 27.24 & 56.12 & -20.10 & -2.39 & 0.46 \\ 4.47 & -21.86 & -60.76 & 10.25 & 13.15 & -7.09 & -8.54 & 4.88 \\ -46.83 & 7.37 & 77.13 & -24.56 & -28.91 & 9.93 & 5.42 & -5.65 \\ 48.53 & 12.07 & 34.10 & -14.76 & -10.24 & 6.30 & 1.83 & 1.95 \\ 12.12 & -6.55 & -13.20 & -3.95 & -1.87 & 1.75 & -2.79 & 3.14 \\ -7.73 & 2.91 & 2.38 & -5.94 & -2.38 & 0.94 & 4.30 & 1.85 \\ -1.03 & 0.18 & 0.42 & -2.42 & -0.88 & -3.02 & 4.12 & -0.66 \\ -0.17 & 0.14 & -1.07 & -4.19 & -1.17 & -0.10 & 0.50 & 1.68 \end{bmatrix}$$

FIGURA 4. Coeficientes $\hat{f}_I[k,l]$ de los elementos de la figura 3. *Fuente:* [**1**, Fig 2.15]

al bloque siguiente, y JPEG sacará provecho de esto almacenando las diferencias $DC^i - DC^{i-1}$, que por lo general serán próximas a cero.

Por último, JPEG hace una normalización adicional para reducir el tamaño neto de los coeficientes $\hat{f}_I[k,l]$, y por tanto, poderlos codificar luego con menos dígitos. Para ello, considera la señal

$$\big(f[n,m] - 128\big)_{0\le m,n<N}\ ,$$

de modo que al restarle el punto medio del intervalo $[0,255]$, la señal tiene ahora valores en el intervalo $[-128,127]$. Por tanto,

$$|\hat{f}_I[k,l]| \le \frac{1}{4}\,8^2\,128 = 2048 = 2^{11}, \tag{5.2.1}$$

y los números $\hat{f}_I[k,l]$ pueden codificarse con a lo sumo 12 bits.

La Figura 4 muestra los valores de los coeficientes $\left\{\hat{f}_I[k,l]\right\}_{k,l=0}^{7}$ asociados al bloque 8×8 normalizados de la Figura 3. Obsérvese que el coeficiente DC es el mayor en valor absoluto, y que los coeficientes AC más grandes (en valor absoluto) están situados cerca de la esquina superior-izquierda de la matriz.

3. Cuantización de los bloques

Este es el apartado donde se produce la mayor compresión de la señal, a costa de perder la información que se considera menos relevante. El ojo humano es más sensible a los cambios en bajas que en altas frecuencias. Por ello, a la hora de redondear (cuantizar) la matriz $(\hat{f}_I[k,l])_{k,l=0}^{7}$, es conveniente usar una cuantización

$$\begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -3 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

FIGURA 5. Matriz obtenida por la cuantizacion de la Figura 4 con la matriz de cuantización W en (5.3.1). *Fuente:* [**1**, Fig 2.16]

ponderada que distinga mejor en la zona de bajas frecuencias, y redondee más en la zona de altas frecuencias.

Por ello los coeficientes se cuantizan según la fórmula

$$Q_1\Big(\frac{\hat{f}_I[k,l]}{w_{k,l}}\Big),$$

donde $Q_1(x)$ denota el redondeo al entero más cercano, y donde $w_{k,l}$ es una colección adecuada de pesos fijada por el grupo de expertos JPEG. Abajo mostramos una matriz de pesos específica del formato JPG80 Standard:

$$W = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 108 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 194 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 121 & 100 & 103 & 99 \end{bmatrix} \tag{5.3.1}$$

Después de dividir cada coeficiente del bloque de la Figura 4 por el correspondiente elemento de la matriz W en (5.3.1), y redondear el resultado al entero más cercano, el resultado queda como se ve en la Figura 5. Aquí se aprecia que casi dos tercios de los coeficientes se han redondeado a cero.

4. Codificación de cada bloque 8×8

Para codificar los valores de un bloque 8×8, lo primero que hacemos es ordenarlos mediante una lectura en zig-zag como en la Figura 6. Es decir, primero situamos el coeficiente DC y después los coeficientes AC, comenzando por las diagonales más cercanas a la esquina superior izquierda. Para la matriz cuantizada de la Figura 5, este ordenamiento quedaría

$$\begin{aligned} &-26,\ -3,\ 0,\ -3,\ -2,\ -6,\ 2,\ -4,\ 1,\ -3,\ 1,\ 1 \\ &\quad 5,\ 1,\ 2,\ -1,\ 1,\ -1,\ 2,\ 0,\ 0,\ 0,\ 0,\ 0,\ -1,\ -1, \quad EOB, \end{aligned}$$

donde EOB (End of Block, en inglés fin del bloque) significa que el resto de los elementos hasta completar los 64 coeficientes son todos cero. La compresión ya se aprecia aquí pues sólo se necesitan 27 de los 64 coeficientes para codificar el original.

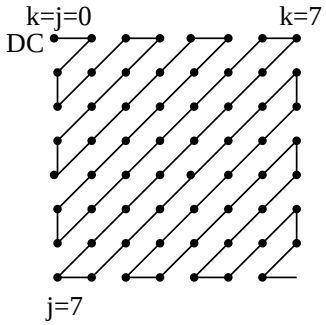


FIGURA 6. Orden de lectura en zig-zag de las matrices cuantizadas. *Fuente:* Mallat, Fig 10.13.

La codificación de estos coeficientes cuantizados se hace en dos etapas:

1. Convertir los coeficientes cuantizados en una sucesión de símbolos binarios.
2. Asignar un código minimal a estos símbolos (como el código de Huffman).

El coeficiente DC (frecuencia cero) se codifica de manera diferente a los restantes, así que a continuación describimos cómo representar los 63 coeficientes AC restantes. Para ello se cuentan sólo los coeficientes *no nulos*, representando el i-ésimo coeficiente no nulo con la siguiente terna de símbolos

$$S_i \equiv (Z_i, L_i)\,(A_i)\,, \tag{5.4.1}$$

donde

$$\begin{aligned} Z_i &= \text{número de ceros consecutivos que preceden al coeficiente } i \\ L_i &= \text{longitud binaria del coeficiente } i \\ A_i &= \text{valor binario del coeficiente } i \end{aligned}$$

Los números Z_i, L_i se escogen siempre en $[0, 15]$, de modo que pueden representarse con 4 dígitos binarios cada uno. Si tuviéramos más de 15 ceros consecutivos se usaría el símbolo

$$(15, 0),$$

que JPEG interpreta como 16 ceros consecutivos, y que puede ir seguido del siguiente símbolo S_{i+1}. Del mismo modo, el símbolo final EOB se escribe como

$$(0, 0),$$

y se interpreta como fin de la codificación del bloque de 8×8.

Respecto al significado de L_i, recordemos de (5.2.1) que los coeficientes $\hat{f}_I[k, l]$, tras ser cuantizados, toman sus valores en $[-2^{11}, 2^{11}]$. Por tanto, la longitud L_i de un coeficiente en sistema binario es a lo sumo de 12 dígitos, y el valor de esa longitud $L_i \in [1, 12]$ se escribe a su vez con 4 dígitos binarios. La siguiente tabla indica la

longitud binaria L_i a partir del valor A_i del coeficiente cuantizado $Q(\hat{f}_I[k,l])$

$$\begin{array}{cc} L_i & A_i \\ 1 & -1,1 \\ 2 & -3,-2,2,3 \\ 3 & -7,...,-4,4,...,7 \\ 4 & -15,....,-8,8,...,15 \\ 5 & -31,...,-16,16,...,31 \\ 6 & -63,...,-32,32,...,63 \\ 7 & -127,...,-64,64,...,127 \\ \vdots & \cdots \end{array}$$

Por último, el valor numérico de A_i en (5.4.1) corresponde a su escritura en sistema binario, según la tabla anterior. Por ejemplo, cuando $L = 1, 2, 3$ podemos poner

$$(5.4.2) \qquad \begin{array}{rr|rr} 1 \to & 0 & 2 \to & 00 \\ -1 \to & 1 & 3 \to & 01 \end{array} \quad \begin{array}{|rr} -2 \to & 10 \\ -3 \to & 11 \end{array} \quad \text{y} \quad \begin{array}{rr|rr} 4 \to & 000 & -4 \to & 100 \\ 5 \to & 001 & -5 \to & 101 \\ 6 \to & 010 & -6 \to & 110 \\ 7 \to & 011 & -7 \to & 111 \end{array} \quad \cdots$$

Por otro lado, el coeficiente DC del bloque, que se coloca en primer lugar, se representa con una pareja de símbolos

$$S_0 \equiv (L_0)\,(A_0)$$

donde ya no es necesario indicar el número de ceros anteriores. La característica principal es que ahora los valores de L_0 y A_0 corresponden al número $DC^i - DC^{i-1}$, que es la diferencia entre los DCs del bloque actual (i-ésimo) y el bloque anterior (tomando $DC^{-1} = 0$ en el caso $i = 1$). Así, el bloque completo quedaría representado como una sucesión de símbolos

$$S_0\, S_1\, \ldots\, S_K\, EOB.$$

A modo de ejemplo, representamos los símbolos correspondientes a la siguiente matriz

$$\begin{bmatrix} 15 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Supongamos que el coeficiente DC del bloque anterior es $DC^{i-1} = 12$. Entonces, $DC^i - DC^{i-1} = 3$, cuya longitud es $L = 2$ y su amplitud $A = 3$. Así, el primer símbolo del código es (2)(3), o en binario (0010)(01), según (5.4.2). Los coeficientes AC se representan de manera similar, dando lugar a

$$(2)(3),\ (1,2)(-2),\ (0,1)(-1),\ (0,1)(-1),\ (0,1)(-1),\ (2,1)(-1),\ (0,1)(-1),\ (0,0)$$

o las correspondientes expresiones binarias (que no escribimos por simplicidad).

En esta cadena, a los símbolos (Z_i, L_i), inicialmente con 8 dígitos binarios cada uno, se les aplica el algoritmo de Huffman. En nuestro ejemplo, las probabilidades y el código de Huffman vienen dados por

$$\begin{array}{lcl@{\qquad}lcl} (2) & \to & p = 1/8 & (0,1) & \to & 0 \\ (1,2) & \to & p = 1/8 & (0,0) & \to & 100 \\ (0,1) & \to & p = 4/8 & (2,1) & \to & 101 \\ (2,1) & \to & p = 1/8 & (1,2) & \to & 110 \\ (0,0) & \to & p = 1/8 & (2) & \to & 111 \end{array}$$

que ahora tiene un promedio de sólo 2 bits/símbolo. Por otro lado, los símbolos (A_i) se codifican según (5.4.2), quedando

$$01\ 10\ 1\ 1\ 1\ 1\ 1$$

Por tanto, en total se necesitan 25 bits para codificar los 64 píxels, lo que da una tasa de 0,39 bits/píxel, en lugar de los 8 bits/píxel originales. Esto corresponde a una compresión cercana al 95 %.

Nota: En la práctica, el código de Huffman se aplica conjuntamente a los datos de todos los bloques 8×8. A la tasa anterior hay que sumarle los bits que ocupa la tabla de descodificación, o bien utilizar las tablas de Huffman sugeridas por JPEG, y que han sido obtenidas estadísticamente con múltiples imágenes.

Ejemplo 5.4.3. Mostramos ahora la codificación en el caso de la matriz cuantizada de la Figura 5:

$$F = \begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -4 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5.4.4}$$

y supongamos que $DC^i - DC^{i-1} = -4$. La representación simbólica en zig-zag de estos coeficientes cuantizados es

$$\begin{array}{ccccccc} (3)(-4), & (0,2)(-3), & (1,2)(-3), & (0,2)(-2), & (0,3)(-6), & (0,2)(2), & (0,3)(-4) \\ (0,1)(1), & (0,3)(-4), & (0,1)(1), & (0,1)(1), & (0,3)(5), & (0,1)(1), & (0,2)(2) \\ (0,1)(-1), & (0,1)(1), & (0,1)(-1), & (0,2)(2), & (5,1)(-1), & (0,1)(-1), & (0,0) \end{array}$$

En este caso las probabilidades y la codificación de Huffman de los símbolos (Z_i, L_i) queda

$$\begin{array}{ll@{\qquad}ll} (3) \to p = \frac{1}{21} & & (3) \to 11111 \\ (0,2) \to p = \frac{5}{21} & & (1,2) \to 11110 \\ (1,2) \to p = \frac{1}{21} & & (5,1) \to 11101 \\ (0,3) \to p = \frac{4}{21} & & (0,0) \to 11100 \\ (0,1) \to p = \frac{8}{21} & & (0,3) \to 110 \\ (5,1) \to p = \frac{1}{21} & & (0,2) \to 10 \\ (0,0) \to p = \frac{1}{21} & & (0,1) \to 0 \end{array}$$

$$\begin{bmatrix} -416 & -33 & -60 & 32 & 48 & -40 & 0 & 0 \\ 0 & -24 & -56 & 19 & 26 & 0 & 0 & 0 \\ -42 & 13 & 80 & -24 & -40 & 0 & 0 & 0 \\ -56 & 17 & 44 & -29 & 0 & 0 & 0 & 0 \\ 18 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

FIGURA 7. Matriz de la Figura 4 después del paso de cuantización inversa. *Fuente:* [**1**, Fig 2.17]

Es decir, esta parte se codifica con

(5.4.5) 11111 10 11110 10 110 10 110 0 110 0 0 110 0 10 0 0 0 10 11101 0 11100

que corresponde a una tasa de $50/21 = 2'38$ bits/símbolo. Por otro lado, las amplitudes A_i se codifican con

(5.4.6) 100 11 11 10 110 00 100 0 100 0 0 001 1 00 1 0 1 00 1 1

que son 36 bits adicionales. En total tenemos una tasa global de $86/64 = 1,34$ bits/píxel, que corresponde a una compresión superior al 80 %.

5. Reconstrucción de la imagen tras la compresión

Revirtiendo cada uno de los pasos anteriores se vuelte a construir una señal en formato RGB, que corresponde a la imagen comprimida. El primer paso es descodificar las cadenas de bits, como (5.4.5) y (5.4.6), siguiendo las tablas con los códigos de Huffman. De este modo se vuelve a obtener la matriz F en (5.4.4) sin pérdida de información.

El siguiente paso es la cuantización inversa, que se obtiene multiplicando los elementos de la matriz F por los de la matriz de pesos W en (5.3.1). De este modo se obtiene la matriz de la Figura 7, que aunque se asemeja, es distinta de la original en la Figura 4. Por tanto en este paso se produce una pérdida irreversible de información.

El paso final es para calcular la transformada inversa 2D-IDCT$_I$ de la matriz de la Figura 7, redondear al entero más próximo, y sumar 128 a cada coeficiente (si algún coeficiente quedara fuera del rango $[0, 255]$ lo aproximamos por uno de los extremos). El resultado final se muestra en la Figura 8 .

Las diferencias entre cada elemento de la matriz reconstruida en la Figura 8, y la matriz original en la Figura 2, nos da una idea de las diferencias globales entre las imágenes original y comprimida.

La Figura 9 muestra la diferencia entre las dos imágenes en la escala de grises. Se aprecian diferencias visuales en la esquina inferior izquierda de las imágenes. Por último, la Figura 10, extraída del libro [**3**, Fig. 10.14], muestra las imágenes correspondientes a compresiones con JPEG con tasas 0'5 y 0'2 bits/píxel. Como se aprecia en las imágenes, JPEG deja de ser eficiente por debajo de 0'5 bits/píxel.

$$\begin{bmatrix} 60 & 63 & 55 & 58 & 70 & 61 & 58 & 80 \\ 58 & 56 & 56 & 83 & 108 & 88 & 63 & 71 \\ 60 & 52 & 62 & 113 & 150 & 116 & 70 & 67 \\ 66 & 56 & 68 & 122 & 156 & 116 & 69 & 72 \\ 69 & 62 & 65 & 100 & 120 & 86 & 59 & 76 \\ 68 & 68 & 61 & 68 & 78 & 60 & 53 & 78 \\ 74 & 82 & 67 & 54 & 63 & 64 & 65 & 83 \\ 83 & 96 & 77 & 56 & 70 & 83 & 83 & 89 \end{bmatrix}$$

FIGURA 8. Reconstrucción del bloque de la Figura 2 después de aplicar la 2D-IDCT$_I$. *Fuente:* [**1**, fig2.18]

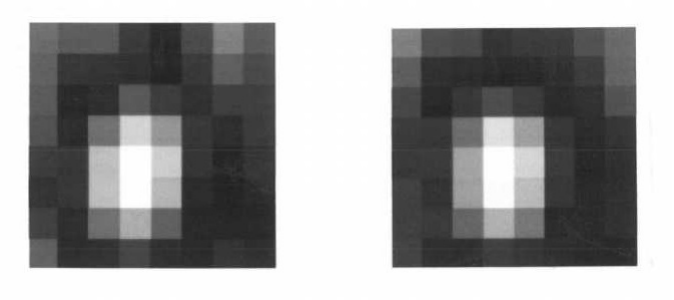


FIGURA 9. Imágenes original y comprimida en escala gris. *Fuente:* [**1**, Fig 2.19]

A Wavelet Tour of Signal Processing
Stéphane Mallat, Academic Press 1999 (2nd edition)

Figure 11.14: Image compression with JPEG.

FIGURA 10. Imágenes comprimidas con JPEG con tasas 0'5 y 0'2 bits/píxel. *Fuente:* [**3**, Fig. 10.14]

Bibliografía

[1] Eugenio Hernández, *Apuntes del curso Ondículas y Tratamiento de Señales*, Universidad Autónoma de Madrid, 2006-07.
[2] G. Folland *Real Analysis*, 2nd Ed. Wiley, 1999.
[3] S. Mallat *A wavelet tour of signal processing*, 3rd Ed. Elsevier 2009.