Compresión de imágenes

Fernando Martínez
fernando.martinez@upc.edu

Departament de Matemàtiques • Universitat Politècnica de Catalunya

15 de abril de 2025

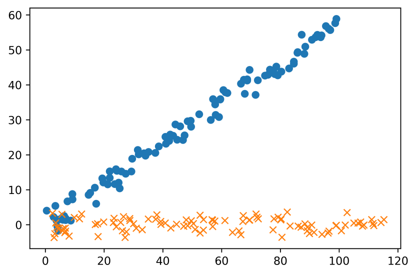
Transformaciones (ejemplos I)

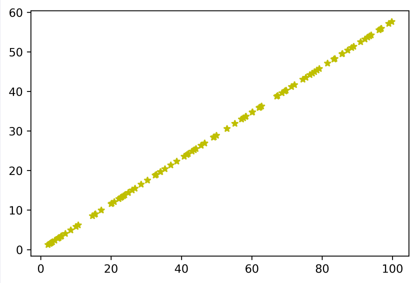
Ejemplo

- Multiplicar *XCVI* por *XII*: *XCVI=96*, *XII=12*, *96·12=1152*, *1152=MCLII*
- Burrows-Wheeler

Ejemplo

Nube de puntos$\rightarrow$ Rotación α $\rightarrow$ cuantización $\rightarrow$ Rotación $-\alpha$.

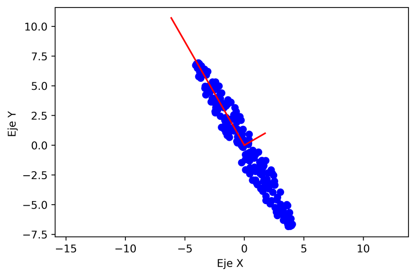


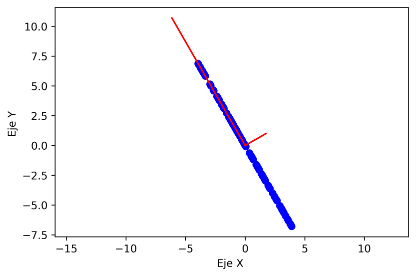


Transformaciones (ejemplos II)

Ejemplo

Nube de puntos$\rightarrow$ Tranformación $\rightarrow$ cuantización $\rightarrow$ Transformación inversa.





Transformaciones lineales invertibles:

$$\vec{y} = A\vec{x} \qquad y_i = \sum_j a_{ij} x_j$$

$$\vec{x} = A^{-1}\vec{y} \equiv B\vec{y} \qquad x_i = \sum_j b_{ij} y_j$$

Transformaciones lineales ortogonales: $A^{-1} = A^T$,

$$y_i = \sum_j a_{ij} x_j \qquad x_i = \sum_j a_{ji} y_j$$

Las transformaciones ortogonales conservan la norma (energía)

$$\|\vec{y}\|^2 = \vec{y}^T \cdot \vec{y} = (A\vec{x})^T \cdot A\vec{x} = \vec{x}^T A^T \cdot A\vec{x} = \vec{x}^T \cdot \vec{x} = \|\vec{x}\|^2$$

Transformaciones lineales ortogonales: Ejemplo 1

Ejemplo

$$G_\alpha = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}, \qquad G^{-1} = G^T$$

$$G_{\frac{\pi}{4}} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix}, \qquad \vec{x} = \begin{pmatrix} a \\ a \end{pmatrix}, \qquad \vec{y} = G_{\frac{\pi}{4}} \vec{x} = \begin{pmatrix} \sqrt{2}a \\ 0 \end{pmatrix}$$

Si $\vec{x}$ tiene la energía repartida equitativamente entre las componentes, la transformación la concentra en la primera componente: modificar la segunda componente transformada no afecta significativamente a la *energía* y, por lo tanto, el error sería pequeño:

$$\vec{x} = \begin{pmatrix} a-\epsilon \\ a+\epsilon \end{pmatrix}, \quad \vec{y} = G_{\frac{\pi}{4}} \vec{x} = \begin{pmatrix} \sqrt{2}a \\ \sqrt{2}\epsilon \end{pmatrix}.$$

Transformaciones lineales ortogonales: Ejemplo 2 (I)

Ejemplo

$$W = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix}, \qquad W^{-1} = W^T = W$$

Si:

$$\vec{x} = \begin{pmatrix} a, & a, & a, & a, \end{pmatrix}^T, \qquad \vec{y} = W\vec{x} = \begin{pmatrix} 2a, & 0, & 0, & 0 \end{pmatrix}^T$$
$$\vec{x} = \begin{pmatrix} a, & a, & -a, & -a, \end{pmatrix}^T, \quad \vec{y} = W\vec{x} = \begin{pmatrix} 0, & 2a, & 0, & 0 \end{pmatrix}^T$$
$$\vec{x} = \begin{pmatrix} a, & -a, & -a, & a, \end{pmatrix}^T, \quad \vec{y} = W\vec{x} = \begin{pmatrix} 0, & 0, & 2a, & 0 \end{pmatrix}^T$$
$$\vec{x} = \begin{pmatrix} a, & -a, & a, & -a, \end{pmatrix}^T, \quad \vec{y} = W\vec{x} = \begin{pmatrix} 0, & 0, & 0, & 2a \end{pmatrix}^T$$

Si la energía está repartida equitativamente entre las componentes, la transformación la concentra en la primera componente. Si hay un salto brusco a *mitad* concentra la energía en la segunda componente...

Transformaciones lineales ortogonales: Ejemplo 2 (II)

Ejemplo (continuación)

Si

$$\vec{x} = \begin{pmatrix} 5 \\ 9 \\ 10 \\ 8 \end{pmatrix} = 8 \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} - 1 \begin{pmatrix} 1 \\ 1 \\ -1 \\ -1 \end{pmatrix} - \frac{3}{2} \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix} - \frac{1}{2} \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix}.$$

$$\vec{y} = W\vec{x} = \begin{pmatrix} 16 \\ -2 \\ -3 \\ -1 \end{pmatrix} = 16 \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} - 2 \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} - 3 \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} - 1 \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

Transformaciones lineales ortogonales: Ejemplo 2 (III)

Ejemplo (continuación)

Si eliminamos la última componente

$$\vec{y'} = \begin{pmatrix} 16 \\ -2 \\ -3 \\ 0 \end{pmatrix} = 16 \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} - 2 \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} - 3 \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

entonces

$$\vec{x'} = W^{-1}\vec{y'} = W^T\vec{y'} = W\vec{y'} = \begin{pmatrix} 5{,}5 \\ 8{,}5 \\ 10{,}5 \\ 7{,}5 \end{pmatrix}.$$

Comparando la *energía*: $\|\vec{x}\| = \|\vec{y}\| = 16{,}43$, $\|\vec{x'}\| = \|\vec{y'}\| = 16{,}40$.

Transformaciones lineales

En el caso de imágenes se acostumbra a trabajar con bloques que son matrices

$$Y_{kl} = \sum_i \sum_j a_{klij} X_{ij}$$

Las transformaciones lineales con las que se trabaja son separables: se aplica primero una transformación a las filas y después otra (usualmente la misma) a las columnas:

$$Y_{kl} = \sum_i \sum_j A_{ki} B_{lj} X_{ij} = \sum_i \sum_j A_{ki} A_{lj} X_{ij} = \sum_i \sum_j A_{ki} X_{ij} A_{lj},$$

matricialmente:

$$Y = AXA^T, \quad X = A^T Y A \qquad (A^{-1} = A^T \text{ortogonal}).$$

Discrete Walsh-Hadamard Transform (DWHT)

Se definen las matrices auxiliares $\tilde{H}_1 = (1)$, $\tilde{H}_{2N} = \begin{pmatrix} \tilde{H}_N & \tilde{H}_N \\ \tilde{H}_N & -\tilde{H}_N \end{pmatrix}$

$$\tilde{H}_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \ \tilde{H}_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix},$$

$$\tilde{H}_8 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix}.$$

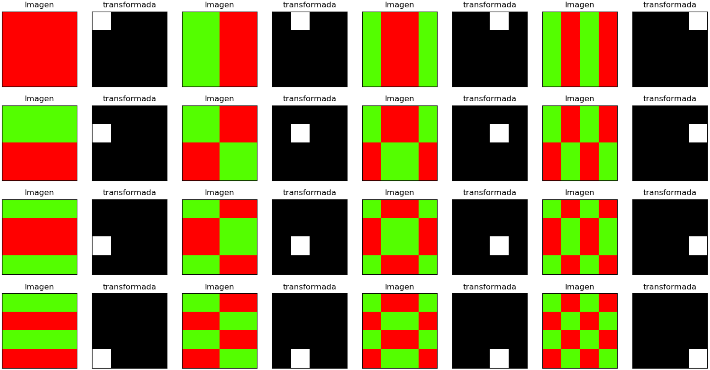
Discrete Walsh-Hadamard Transform (DWHT)

Se multiplican por $\frac{1}{\sqrt{N}}$ (así conseguiremos $HH^T = I$) y se reordenan las filas según el número de cambios de signo:

$$H_2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix},\ H_4 = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix},$$

$$H_8 = \frac{1}{\sqrt{8}}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{pmatrix}.$$

Discrete Walsh-Hadamard Transform (DWHT)



$$N = 4$$

Discrete Walsh-Hadamard Transform (DWHT)

Las *imágenes* I_i son *base*: cualquier imagen I se puede escribir como combinación lineal de los elementos de la base $I = \sum_i \lambda_i I_i$.

La transformación es lineal: $A \cdot I = \displaystyle\sum_i \lambda_i A \cdot I_i$.

$A \cdot I_i$ es una matriz cuyos elementos son todos 0 excepto el que ocupa la *posición* i que vale 1. Por lo tanto:

$$A \cdot I = \begin{pmatrix} \lambda_1 & \lambda_2 & ... \\ ... & ... & ... \\ ... & ... & \lambda_r \end{pmatrix}$$

λ_1 recibe el nombre de componente principal, es la que tiene mayor *influencia* en la imagen y acostumbra a tener el mayor valor (es la media de la imagen), a medida que nos *alejamos* de λ_1 menos *influencia* en la imagen y podemos prescindir de ellas.

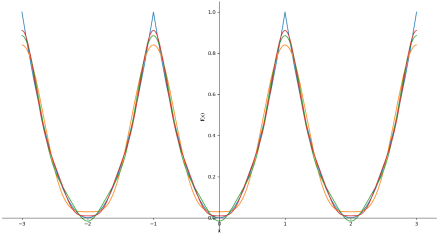
Serie trigonométrica de Fourier

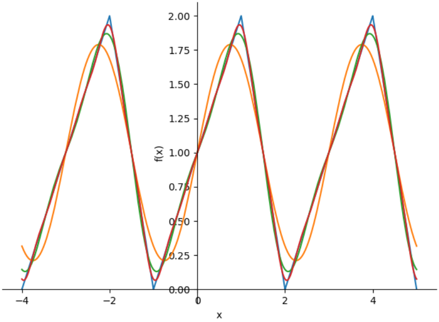
Dada una función suficientemente regular periódica de periodo T, $f(t) = f(t + kT),\ k \in \mathbb{Z}$, podemos representarla por su serie trigonométrica de Fourier:

$$f(t) = \frac{a_0}{2} + \sum_{n=1} a_n \cos(\frac{2\pi}{T} n\, t) + \sum_{n=1} b_n \sin(\frac{2\pi}{T} n\, t)$$

$$a_n = \frac{2}{T} \int_0^T f(t) \cos(\frac{2\pi}{T} n\, t) dt$$
$$b_n = \frac{2}{T} \int_0^T f(t) \sin(\frac{2\pi}{T} n\, t) dt$$

Si T es el periodo, $\nu \equiv \frac{1}{T}$ es la frecuencia y $\omega \equiv \frac{2\pi}{T} = 2\pi\nu$ es la frecuencia angular.


1.0
0.8
0.6
0.4
0.2
0.0
f(x)
−3
−2
−1
0
1
2
3
x


2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
f(x)
−4
−2
0
2
4
x

Serie trigonométrica de Fourier

1. $\int_0^T \cos(n\,\omega\,t)\cos(k\,\omega\,t)dt = \int_0^T \sin(n\,\omega\,t)\sin(k\,\omega\,t)dt = \frac{T}{2}\delta_{kn}$
2. $\int_0^T \cos(n\,\omega\,t)\sin(k\,\omega\,t)dt = 0$

Una función f se puede escribir como $f(t) = f_P(t) + f_I(t)$ siendo $f_P(t) = \frac{f(t)+f(-t)}{2}$ una función par y $f_I(t) = \frac{f(t)-f(-t)}{2}$ una función impar.

Si

$$f(t) = \frac{a_0}{2} + \sum_{n=1} a_n \cos(\frac{2\pi}{T} n\,t) + \sum_{n=1} b_n \sin(\frac{2\pi}{T} n\,t)$$

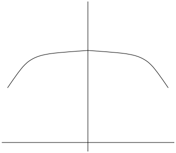
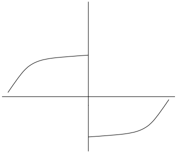
entonces

$$f_P(t) = \frac{a_0}{2} + \sum_{n=1} a_n \cos(\frac{2\pi}{T} n\,t) \qquad f_I(t) = \sum_{n=1} b_n \sin(\frac{2\pi}{T} n\,t)$$

Identidad de Parseval

$$\frac{2}{T}\int_0^T |f(t)|^2\,\mathrm{d}t = \frac{a_0^2}{2} + \sum_{n=1}^{\infty}(a_n^2 + b_n^2)$$

Dada una función definida en el intervalo $[0, T/2]$ la podemos extender a toda la recta real

- de forma simétrica ($b_n = 0$), 
- de forma antisimétrica ($a_n = 0$), 
- de cualquier otra forma.

Discrete Cosine Transform (DCT)

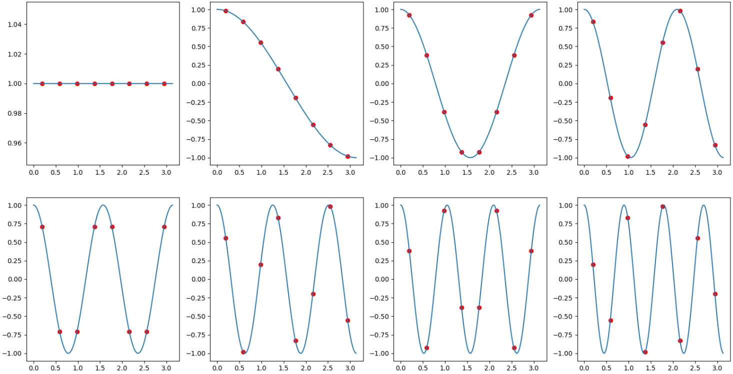
Teorema de Nyquist

Teorema

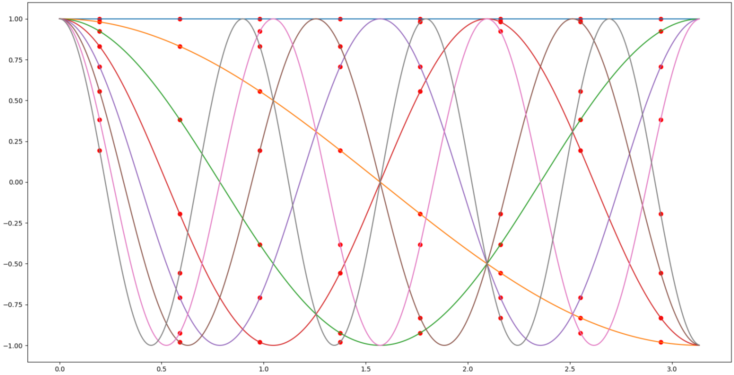
Es posible reconstruir una señal periódica continua si se muestrea a una tasa mayor que el doble de su frecuencia máxima.

En la práctica la frecuencia máxima vendrá dada por el número de muestras (*samples*) N ya que hacemos una extensión simétrica $T = 2N$ (DCT)

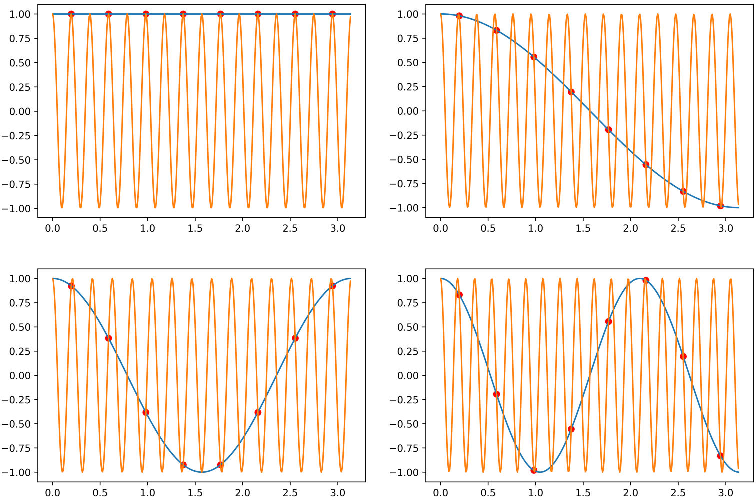
Teorema de Nyquist



Teorema de Nyquist



Teorema de Nyquist



Discrete Cosine Transform (DCT)

Un vector $\vec{x}$ de N componentes puede representar una señal periódica simétrica de frecuencia máxima $N-1$.

$$\vec{x} = \sum_{n=0}^{N-1} a_n \vec{\Theta}^{(n)}, \Theta_i^{(n)} = c_n \cos\left(\frac{2\pi}{2N} n(i+\frac{1}{2})\right),\ c_0 = \frac{1}{2\sqrt{2}}, c_n = \frac{1}{2}\ n \neq 0$$

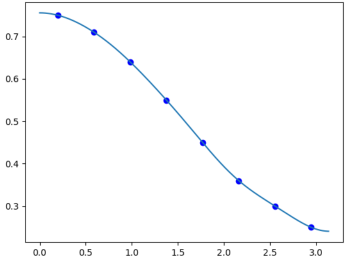
$$\vec{\Theta}^{(0)} = \frac{1}{2\sqrt{2}}[1, ..., 1]$$
$$\vec{\Theta}^{(1)} = \frac{1}{2}\left[\cos\left(\frac{\pi}{2N}\right), \cos\left(\frac{3\pi}{2N}\right), \cos\left(\frac{5\pi}{2N}\right), ..., \cos\left(\frac{\pi}{N}(N-1+\frac{1}{2})\right)\right]$$
$$\vec{\Theta}^{(2)} = \frac{1}{2}\left[\cos\left(\frac{\pi}{N}\right), \cos\left(\frac{3\pi}{N}\right), \cos\left(\frac{5\pi}{N}\right), ..., \cos\left(\frac{\pi}{N}2(N-1+\frac{1}{2})\right)\right]$$
$$...$$
$$\vec{\Theta}^{(N-1)} = \frac{1}{2}\left[\cos\left(\frac{\pi(N-1)}{N}\right), \cos\left(\frac{3\pi(N-1)}{2N}\right), ..., \cos\left(\frac{\pi}{N}(N-1)(N-1+\frac{1}{2})\right)\right]$$

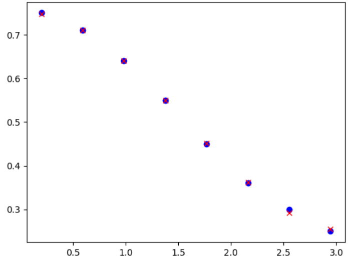
Ejemplo DCT (I)

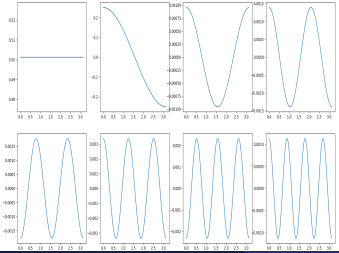
Ejemplo

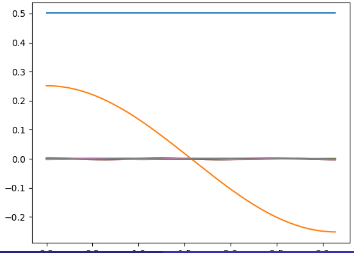
- Dominio espacial:
  $\vec{x} = [0{,}75,\ 0{,}71,\ 0{,}64,\ 0{,}55,\ 0{,}45,\ 0{,}36,\ 0{,}3,\ 0{,}25]$
- Dominio frecuencias:
  $\vec{a} = [1{,}418,\ 0{,}503,\ 0,00191,\ 0{,}00279,\ -0{,}00354,\ 0{,}00672,\ -0{,}00462,\ 0{,}00225]$
- Dominio frecuencias:
  $\vec{a}' = [1{,}418,\ 0{,}503,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0] = [\frac{0{,}501}{2\sqrt{2}},\ \frac{0{,}251}{2},\ 0,\ 0,\ 0,\ 0,\ 0,\ 0]$
- Dominio espacial:
  $\vec{x}' = [0{,}748,\ 0{,}710,\ 0{,}641,\ 0{,}550,\ 0{,}452,\ 0{,}361,\ 0{,}292,\ 0{,}254]$
- Energia perdida: 0.005 %

Ejemplo DCT (II)









Discrete Cosine Transform (DCT) 1D

Una manera de calcular ***una***[1] DCT-1D de $\vec{x} = (x_0, x_1, ..., x_{N-1})$ es:

$$a_i = \sqrt{\frac{2}{N}} c_i \sum_{k=0}^{N-1} x_k \cos\left(\frac{(2k+1)i\pi}{2N}\right)$$

$$c_0 = \frac{1}{\sqrt{2}}, \qquad c_i = 1,\ i \neq 0$$

La inversa, dadas las amplitudes a_i, es:

$$x_k = \sqrt{\frac{2}{N}} \sum_{i=0}^{N-1} c_i\, a_i \cos\left(\frac{(2k+1)i\pi}{2N}\right)$$

[1] Hay otras definiciones de DCT, dependiendo de cómo se toman los puntos.

Discrete Cosine Transform (DCT) 2D

En el caso 2D, la DCT que se utiliza en JPEG es:

$$\omega_{ij} = \frac{1}{\sqrt{2N}} c_i c_j \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} P_{xy} \cos\left(\frac{(2x+1)i\pi}{2N}\right) \cos\left(\frac{(2y+1)j\pi}{2N}\right)$$

$$c_0 = \frac{1}{\sqrt{2}}, \qquad c_i = 1,\ i \neq 0$$

siendo P una imagen de $N \times N$ píxeles y P_{xy} el valor del píxel correspondiente. La inversa es:

$$P_{xy} = \frac{1}{\sqrt{2N}} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} c_i c_j \omega_{ij} \cos\left(\frac{(2x+1)i\pi}{2N}\right) \cos\left(\frac{(2y+1)j\pi}{2N}\right)$$

Discrete Cosine Transform (DCT) 2D

Matricialmente se puede escribir:

$$\omega = C\,P\,C^T$$

$$C = (C_{ij}) \qquad C_{0j} = \frac{1}{\sqrt{N}}, \quad C_{ij} = \sqrt{\frac{2}{N}} \cos\left(\frac{(2j+1)i\pi}{2N}\right)$$

Lo usual es aplicar la DCT a bloques 8×8 de la imagen. En este caso al calcular $C\,P$ cada elemento de la matriz requiere 8 productos, en total $8 \cdot 8 \times 8$ para el bloque. Multiplicar por C^T necesita los mismos productos. En total $2 \cdot 8^3$ multiplicaciones.

Para una imagen $n \times n$ son necesarios $\frac{n}{8}\frac{n}{8} 2 \cdot 8^3 = 16n^2$ productos. (Aplicar la DCT a toda la imagen: $2n^3$)

Se puede reducir el número de productos escribiendo $C = C_1 C_2 \cdots C_7$ siendo C_1 matrices con pocos elementos no nulos y siendo éstos en su mayoría ± 1.[2]

[2] Ver página 323 de la 4ª edición del Salomon.

Discrete Sine Transform (DST)

Notemos que $\sin(x)$ es una función impar i.e. $f(0) = 0$.

Por lo tanto la DST no tiene coeficiente DC (siempre vale 0).

$$a_i = \sum_{k=0}^{N} x_k \sin\left(\frac{\pi}{N+1}(k+1)(i+1)\right)$$

Discrete Sine Transform (DST)

Ejemplo

$$x = (1, 1, 1, 1, 1, 1, 1, 1)$$
$$DST(x) = (5{,}67,\ 0,\ 1{,}73,\ 0, 0{,}84\ , 0,\ 0{,}36,\ 0)$$
$$DCT(x) = (8, 0, 0, 0, 0, 0, 0, 0)$$

Ejemplo

$$x = (0,\ 0{,}87,\ 1{,}47,\ 1{,}985,\ 2{,}5,\ 1{,}866,\ 0{,}643,\ 0{,}002)$$
$$DST(x) = (8{,}28,\ -0{,}297,\ -2{,}57,\ 0{,}750,\ -0{,}522,\ -0{,}644,\ -0{,}137,\ 0{,}0176)$$
$$DCT(x) = (3{,}30,\ -0{,}0668,\ -2{,}42,\ 0{,}314,\ -0{,}128,\ -0{,}365,\ -0{,}0157,\ 0{,}0247)$$

JPEG (Joint Photographic Experts Group)

I) Transformación $RGB \to YC_bC_r$, en el caso de imágenes de color:

$$
\begin{aligned}
Y &= 0{,}299\,R + 0{,}587\,G + 0{,}114\,B \\
C_b &= -0{,}1687\,R - 0{,}3313\,G + 0{,}5\,B + 128 \\
C_r &= 0{,}5\,R - 0{,}4187\,G - 0{,}0813\,B + 128
\end{aligned}
$$

$$
\begin{aligned}
R &= Y + 1{,}402\,(C_r - 128) \\
G &= Y - 0{,}71414\,(C_r - 128) - 0{,}34414\,(C_b - 128) \\
B &= Y + 1{,}772\,(C_b - 128)
\end{aligned}
$$

II) Para cada componente, se divide la imagen en bloques 8×8. (Si el número de filas o columnas no es multiplo de 8, se replica la última fila/columna.)

III) A cada bloque se le aplica la DCT, pero previamente se le resta 128 a cada elemento del bloque para que el coeficiente DC se reduzca su valor en $128 \cdot 64 = 8192$ en promedio.

JPEG (Joint Photographic Experts Group)

- IV) Se cuantizan los valores obtenidos, con las matrices de cuantización. Los estándares usan una para la componente Y, luminancia, y otras para las componentes C_b y C_r, crominancias, pero se pueden utilizar otras que se han de incluir.

$$Q_L = \begin{pmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix}$$

$$Q_C = \begin{pmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{pmatrix}$$

- V) Se tratan los coeficientes DC y AC por separado. Todos los DC se codifican juntos (sus diferencias); los 63 AC se codifican juntos.

JPEG (DC)

0:	0										0
1:	-1	1									10
2:	-3	-2	2	3							110
3:	-7	-6	-5	-4	4	5	6	7			1110
4:	-15	-14	...	-9	-8	8	9	10	...	15	11110
5:	-31	-30	-29	...	-17	-16	16	17	...	31	111110
6:	-63	-62	-61	...	-33	-32	32	33	...	63	1111110
7:	-127	-126	-125	...	-65	-64	64	65	...	127	11111110
$\vdots$				$\vdots$							
14:	-16383	-16382	-16381	...	-8193	-8192	8192	8193	...	16383	111111111111110
15:	-32767	-32766	-32765	...	-16385	-16384	16384	16385	...	32767	1111111111111110
16:	32768										1111111111111111

Figura: Código de Huffman para los coeficientes DC y sus diferencias

JPEG (DC)

Ejemplo

DC: 1118, 1114, 1119...$\longmapsto$ 1118, -4, 5

1118 está en la fila 11, columna 1118 de la tabla 1, codificamos

$$\overbrace{111111111110}^{\text{fila 11}} \quad \overbrace{10001011110}^{\text{columna 1118 con 11 bits}}$$

-4 está en la fila 3, columna 3 de la tabla, codificamos

$$\overbrace{1110}^{\text{fila 3}} \quad \overbrace{011}^{\text{columna 3 con 3 bits}}$$

5 está en la fila 3, columna 5 de la tabla, codificamos

$$\overbrace{1110}^{\text{fila 3}} \quad \overbrace{101}^{\text{columna 5 con 3 bits}}$$

JPEG (AC)

Para cada bloque se codifican los 63 coeficientes AC con una combinación de RLE y Huffman.

Los 63 coeficientes AC formarán una lista que contenga unos pocos elementos no nulos, por ejemplo $2, 0, -2, \overbrace{0..,0}^{13 \text{ ceros}}, -1, 0, ...$

Se codifica de la siguiente forma, para cada valor $x \neq 0$:

1. z representa el número de ceros que preceden a x,
2. se busca x en la tabla 1, sea R su fila y C su columna,
3. Con R y z, se usa la tabla 2 para elegir una entrada E
4. E se concatena con el valor de C usando R bits.

		R			
Z	1 6	2 7	3 8	4 9	5 10
0	00 111000	01 1111000	100 1111110110	1011 1111111110000010	11010 1111111110000011
1	1100 1111111110000100	11011 1111111110000101	1111001 1111111110000110	111110110 1111111110000111	11111110110 1111111110001000
2	11100 1111111110001010	11111001 1111111110001011	1111110111 1111111110001100	111111110100 1111111110001101	1111111110001001 1111111110001110
3	111010 1111111110010001	111110111 1111111110010010	111111110101 1111111110010011	1111111110001111 1111111110010100	1111111110010000 1111111110010101
4	111011 1111111110011001	1111111000 1111111110011010	1111111110010110 1111111110011011	1111111110010111 1111111110011100	1111111110011000 1111111110011101
5	1111010 1111111110100001	11111110111 1111111110100010	1111111110011110 1111111110100011	1111111110011111 1111111110100100	1111111110100000 1111111110100101
6	1111011 1111111110101001	111111110110 1111111110101010	1111111110100110 1111111110101011	1111111110100111 1111111110101100	1111111110101000 1111111110101101
7	11111010 1111111110110001	111111110111 1111111110110010	1111111110101110 1111111110110011	1111111110101111 1111111110110100	1111111110110000 1111111110110101
8	111111000 1111111110111001	111111111000000 1111111110111010	1111111110110110 1111111110111011	1111111110110111 1111111110111100	1111111110111000 1111111110111101
9	111111001 1111111111000010	1111111110111110 1111111111000011	1111111110111111 1111111111000100	1111111111000000 1111111111000101	1111111111000001 1111111111000110
10	111111010 1111111111001011	1111111111000111 1111111111001100	1111111111001000 1111111111001101	1111111111001001 1111111111001110	1111111111001010 1111111111001111
11	1111111001 1111111111010100	1111111111010000 1111111111010101	1111111111010001 1111111111010110	1111111111010010 1111111111010111	1111111111010011 1111111111011000
12	1111111010 1111111111011101	1111111111011001 1111111111011110	1111111111011010 1111111111011111	1111111111011011 1111111111100000	1111111111011100 1111111111100001
13	11111111000 1111111111100110	1111111111100010 1111111111100111	1111111111100011 1111111111101000	1111111111100100 1111111111101001	1111111111100101 1111111111101010
14	1111111111101011 1111111111110000	1111111111101100 1111111111110001	1111111111101101 1111111111110010	1111111111101110 1111111111110011	1111111111101111 1111111111110100
15	1111111111110101 1111111111111010	1111111111110110 1111111111111011	1111111111110111 1111111111111101	1111111111111000 1111111111111110	1111111111111001 1111111111111111
16	11111111001				

JPEG: ejemplo AC (I)

Ejemplo

$$2, 0, -2, \overbrace{0..,0}^{13\ ceros}, -1, 0, ...$$

- $z = 0$, $x = 2$: $R = 2$, $C = 2$
  $R = 2$, $z = 0 \to 01$
  $C = 2$ con $R = 2$ bits $\to 10$
  `0110`
- $z = 1$, $x = -2$: $R = 2$, $C = 1$
  $R = 2$, $z = 1 \to 11011$
  $C = 1$ con $R = 2$ bits $\to 01$
  `1101101`

JPEG: ejemplo AC (II)

Ejemplo

$$2, 0, -2, \overbrace{0..,0}^{13\ ceros}, -1, 0, ...$$

- $z = 13$, $x = -1$: $R = 1$, $C = 0$
  $R = 1$, $z = 13 \rightarrow 11111111000$
  $C = 0$ con $R = 1$ bits $\rightarrow 0$
  `111111110000`

Como ya no hay más términos no nulos se envía `EOB`: `1010`.
Si hay 16 o más ceros consecutivos se envía `11111111001` para indicar 15 ceros y se procede con lo restante.
Resumiendo: `0110 1101101 111111110000 1010`
27 bits para los 63 coeficientes AC.

JPEG: ejemplo AC (III)

Los 63 coeficientes AC se ordenan en zig-zag para conseguir que el número de ceros consecutivos aumente:

$$\begin{pmatrix} -24 & 2 & -4 & 0 & 0 & 0 & 0 & 0 \\ 3 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \rightarrow 2, 3, -1, 2, -4, 0, 0, 2, \overbrace{0..,0}^{11\, ceros}, -1, 0, ...$$

WebP An image format for the Web

https://developers.google.com/speed/webp?hl=en

WebP is a modern image format that provides superior lossless and lossy compression for images on the web. Using WebP, webmasters and web developers can create smaller, richer images that make the web faster.

WebP lossless images are 26 % smaller in size compared to PNGs. WebP lossy images are 25-34 % smaller than comparable JPEG images at equivalent SSIM quality index.