Transformada Rápida de Fourier

Grupo 2 - Laboratorio 4A, 1C2024

Mateo Eljatib, Mora Viola

Índice

1. **Introducción**
    i. Background
    ii. Transformada de Fourier - Repaso
2. **Conceptos importantes**
    i. Aliasing/Muestreo
    ii. Transformada Discreta de Fourier
3. **Transformada Rápida de Fourier (FFT)**
    i. Qué es?
    ii. Para qué sirve?
    iii. Cómo se usa?
4. **Ejemplos**

J.B. Fourier (1768 - 1830)

Introducción - Background

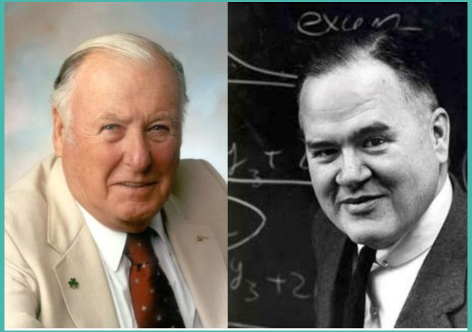
The Fast Fourier Transform (FFT): Most Ingenious Algorithm Ever?

The Remarkable Story Behind The Most Important Algorithm Of All Time

1805

1965

C.F.Gauss(1777 - 1855)



J.W. Cooley (1926-2016) y
J. Tukey (1915-2000)

Introducción - Repaso TF

Proceso físico

Dominio de frecuencias $\boxed{H(f)}$

$$-\infty < f < \infty$$

$$H(f) = \int_{-\infty}^{\infty} h(t) e^{2\pi i f t} dt$$

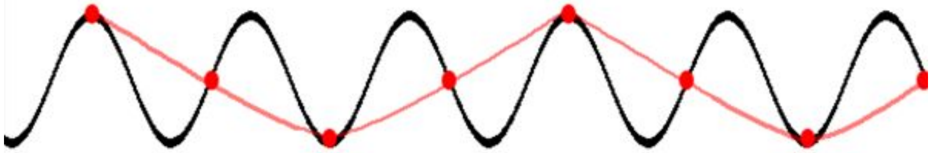
$$h(t) = \int_{-\infty}^{\infty} H(f) e^{-2\pi i f t} df$$

Ec. 1

Dominio temporal $\boxed{h(t)}$

Transformar = Cambiar de representación

Muestreo y Aliasing



Δ : Intervalo de muestreo

$f_c = \frac{1}{2\Delta}$: Frecuencia crítica de Nyquist

$|\sigma|$: Límite de ancho de banda

Teorema de sampleo: $h(t), \Delta \longrightarrow |\sigma| < f_c$ Si $H(f) = 0$ para todo $|f| > f_c$

Entonces $h(t)$ queda completamente determinada por h_n

$$h(t) = \Delta \sum_{n=-\infty}^{+\infty} h_n \frac{\sin[2\pi f_c(t - n\Delta)]}{\pi(t - n\Delta)}$$

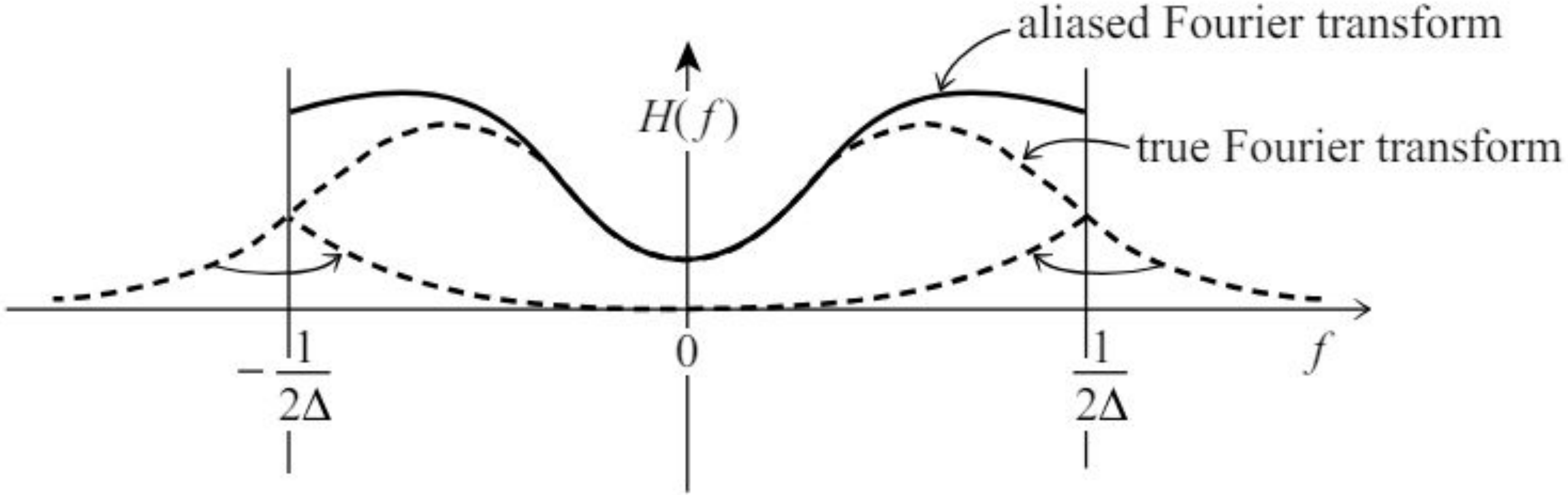
Ec. 2

Determinar Δ para captar todo el contenido de la señal

Aliasing para samplear funciones continuas donde NO se cumple que $|\sigma| < f_c$

Aliasing

“Falsa traducción”



Transformada Discreta de Fourier

$$h_k \equiv h(t_k), \qquad t_k \equiv k\Delta, \qquad k = 0, 1, 2, \ldots, N-1$$

- Integral $\longrightarrow$ Sumatoria
- Tiempos discretizados
- n tiene en cuenta el aliasing y el criterio de Nyquist
- N valores de muestreo consecutivos

$$H(f) = \int_{-\infty}^{\infty} h(t) e^{2\pi i f t} dt$$

Ec. 3

$$\boxed{f_n \equiv \frac{n}{N\Delta},}$$

$$\boxed{n = -\frac{N}{2}, \ldots, \frac{N}{2}}$$

$$H_n \equiv \sum_{k=0}^{N-1} h_k \, e^{2\pi i k n / N}$$

Ec. 4

Motivación FFT

$$W \equiv e^{2\pi i/N} \quad \Longrightarrow \quad H_n = \sum_{k=0}^{N-1} W^{nk} h_k$$

Ec. 5

Tanto la matriz W^{nk} como el vector h_k son de orden N

La transformada es de $O(N^2)$

¿Cómo se puede reducir el orden del proceso?

Fast Fourier Transform (FFT)

$$(\frac{N}{2})^2 + (\frac{N}{2})^2 = \frac{N^2}{2}$$

Ec. 6

Algoritmo para computar en $O(Nlog_2(N))$

Cada transformada cuesta $O(N^2)$ operaciones. Pero si hagos en vez de una para N elementos, dos para N/2 elementos me cuesta la mitad ...

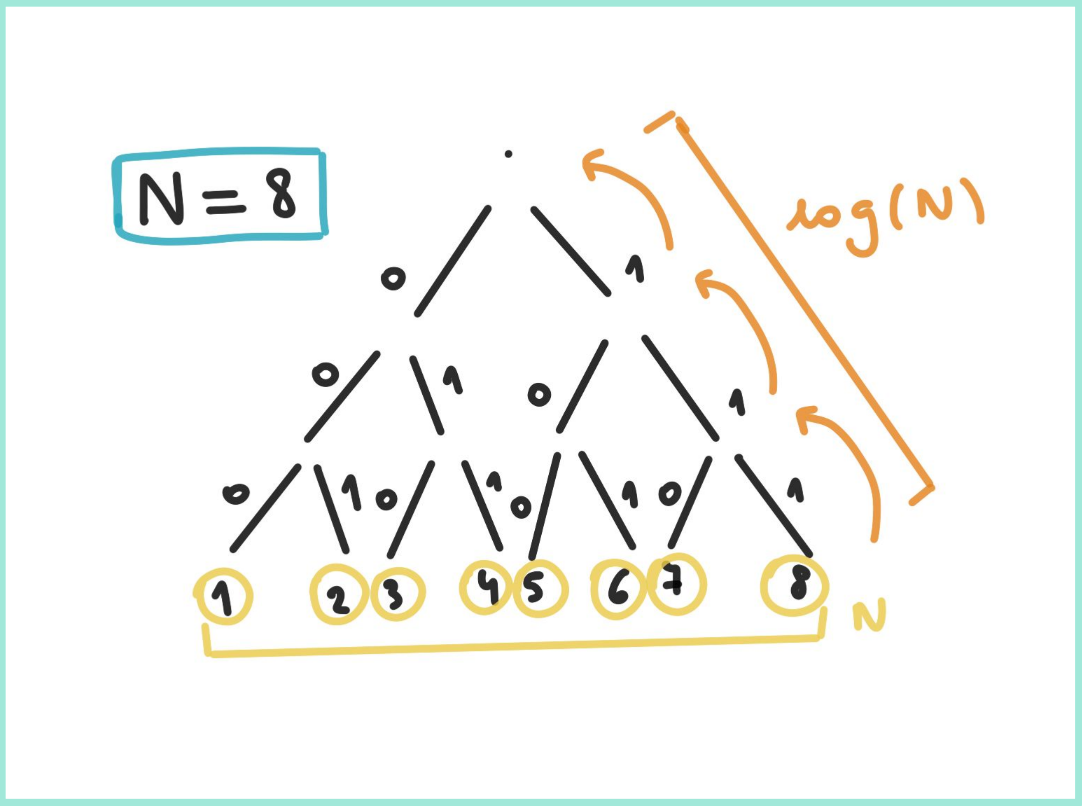
Danielson-Lanczos Lemma

Ec. 7 $$\boxed{F_k = F_k^e + W^k F_k^o}$$

Transformada de los elementos impares (odds)

Transformada de los elementos pares (even)

Ec. 8 $$F_k = \sum_{j=0}^{N-1} e^{2\pi ijk/N} f_j$$



1. **Reordenamiento de bits en reversa**

(No agrega espacio, es intercambiar)

$$F_k^{eoeeoeo\cdots oee} = f_n \qquad \text{for some } n$$

2. **Ejecutar un loop** $log_2(N)$ **veces para calcular N transformadas.**

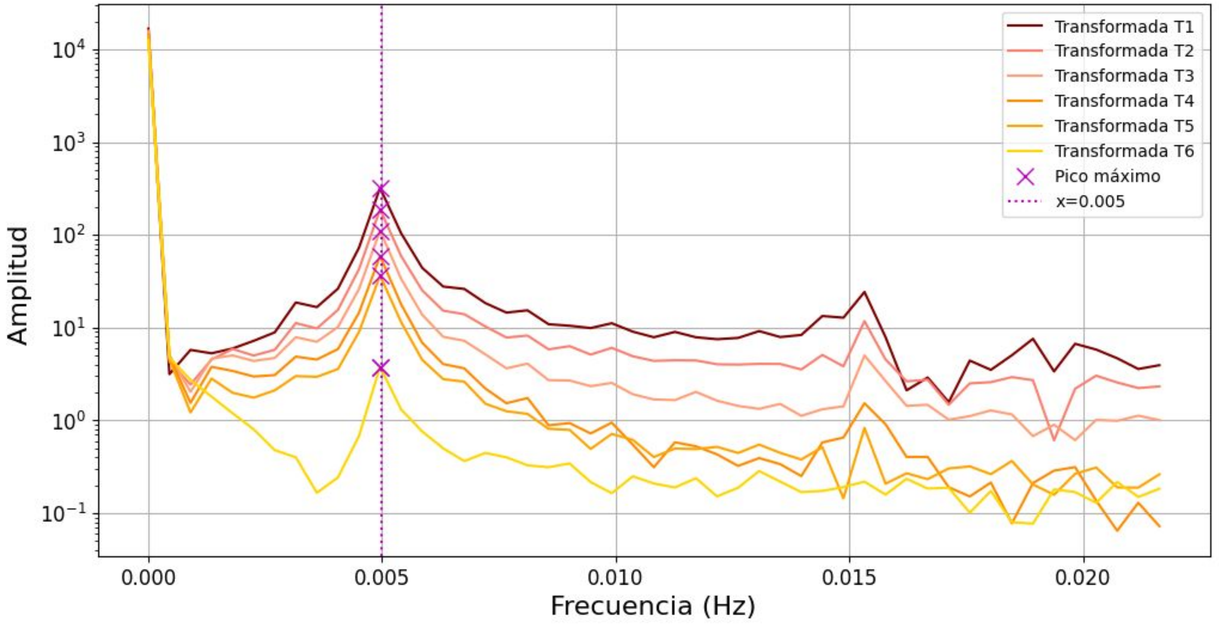
IDEA: N potencia entera de 2

Otros algoritmos de FFT

- *Sandey - Turkey* : Itera sobre los inputs
- base 4 - FFTs, base-8 FFTs
- FFT con N   potencia de 2

Ejemplo: Práctica Difusividad

- Frecuencia de muestreo de 3 segundos, frecuencia de la señal cuadrada de 0,005 Hz de forma que se cumpla el criterio de Nyquist



Código

```
amplitudes = np.array([T1-np.mean(T1), T2-np.mean(T2), T3-np.mean(T3), T4-np.mean(T4), T5-np.mean(T5), T6-np.mean(T6)])
tiempo = t

transformadas = np.fft.fft(amplitudes, axis=1)
largo = len(tiempo)
d_tiempo = np.mean(np.diff(tiempo))


frecuencia = np.fft.fftfreq(largo, d_tiempo)
frecuencia_max = 1 / d_tiempo
frecuencia = np.linspace(0, frecuencia_max, largo)
```

¡Muchas Gracias!