Tarea 2 - Análisis de señales

Camilo Carvajal Reyes

11 de octubre de 2022

1. Definiciones de la Densidad Espectral de Potencia (PSD)

Proposición : Identidad Útil

$$\sum_{k=1}^{N}\sum_{l=1}^{N} f(k-l) = \sum_{\tau=-N+1}^{N-1} (N-|\tau|)f(\tau)$$

Demostración. Probaremos esto por inducción. Para $N=2$ basta notar que

$$\sum_{k=1}^{N}\sum_{l=1}^{N} f(k-l) = 2f(0)+f(-1)+f(1) = \sum_{\tau=-1}^{1}(2-|\tau|)f(\tau) = \sum_{\tau=-N+1}^{N-1}(N-|\tau|)f(\tau)$$

Luego para el paso inductivo usamos que

$$\sum_{k=1}^{N+1}\sum_{l=1}^{N+1} f(k-l) = \sum_{\tau=-N+1}^{N-1}(N-|\tau|)f(\tau)+\sum_{k=1}^{N} f(k-N-1)+\sum_{l=1}^{N} f(N+1-l)+f(0) = \sum_{\tau=-N}^{N}(N+1-|\tau|)f(\tau)$$

Donde en la primera igualdad usamos la hipótesis inductiva (para N), con lo cual concluimos para $N+1$. □

Proposición : Definiciones equivalentes de PSD

$$\sum_{k\in\mathbb{Z}} c(k)e^{-j\omega k} = \lim_{N\to\infty}\mathbb{E}(\frac{1}{N}|\sum_{k=1}^{N} x_k e^{-j\omega k}|^2)$$

Esto nos da la equivalencia de ambas como definiciones de Densidad Espectral de Potencia.

Demostración. En efecto,

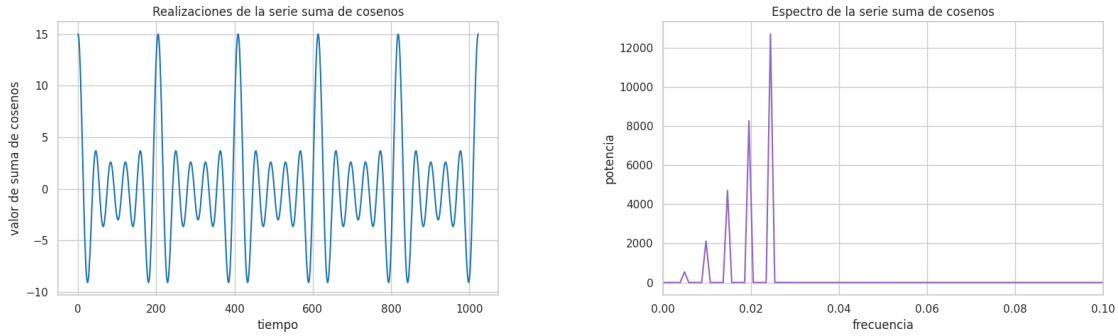
$$\begin{aligned}
\lim_{N\to\infty}\mathbb{E}(\frac{1}{N}|\sum_{k=1}^{N} x_k e^{-j\omega k}|^2) &= \lim_{N\to\infty}\frac{1}{N}\sum_{k=1}^{N}\sum_{l=1}^{N}\mathbb{E}(x_k x_l^*)e^{-j\omega k+j\omega l} \\
\text{(gracias a la identidad útil)} &= \lim_{N\to\infty}\frac{1}{N}\sum_{\tau-N+1}^{N-1}(N-|\tau|)c(\tau)e^{-j\omega\tau} \quad (\text{con } f(n)=c(n)e^{-j\omega n}) \\
&= \lim_{N\to\infty}(\sum_{\tau=-N+1}^{N-1} c(\tau)e^{-j\omega\tau} - \frac{1}{N}\sum_{\tau-N+1}^{N-1}|\tau|c(\tau)e^{-j\omega\tau}) = \sum_{k\in\mathbb{Z}} c(k)e^{-j\omega k}
\end{aligned}$$

Para la última igualdad hemos asumido que $\frac{1}{N}\sum_{\tau-N}^{N}|\tau||r(\tau)|\to 0$ cuando $N\to\infty$, que corresponde a asumir que la sucesión $(c(k))_{k\in\mathbb{N}}$ decae con suficiente rapidez. Se concluye la equivalencia deseada. □

2. Estimación espectral

2.1. Periodograma

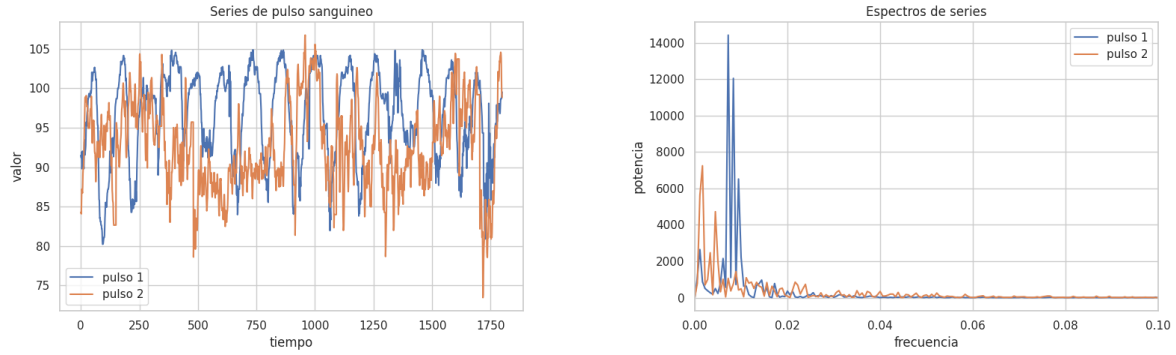
La estimación espectral tiene como objetivo la estimación de la función $S(x)$, que tiene como definición las expresiones de la proposición 1.0.2. A continuación analizaremos el método del periodograma, también conocido como el método de Welch, que correspondo a uno no-paramétrico. En palabras simples, el método toma el promedio de periodogramas a través de "ventanas de tiempo" para realizar la estimación. En efecto, el algoritmo divide la señal en distintos bloques, lo cual a la larga reduce el efecto del ruido asociado a las distintas señales.



(a) Serie $\sum_{n=1}^{5} n\cos(n \cdot 2 \cdot 10 \cdot \pi t)$ (b) Densidad Espectral

Figura 1: Serie y densidad espectral de una seuma de cosenos

En la figura 1 observamos una serie de suma de cosenos $(\sum_{n=1}^{5} n\cos(n \cdot 2 \cdot 10 \cdot \pi t))$ con su correspondiente estimación espectral usando el método del periodograma de la biblioteca *scipy.signal*. Como es de esperar, vemos los cinco peaks crecientes que representan los cosenos en cuestión. Por otro lado, en la figura 2 se observan dos series de pulsos sanguíneos y sus estimaciones espectrales.



(a) Serie (b) Densidad Espectral

Figura 2: Serie y densidad espectral para series de pulso sanguíneo

Al observar el espectro estimado, podemos intuir que el primer pulso (azul) tiene oscilaciones más pronunciadas, ya que tiene frecuencias con alta potencia. Esto se confirma al observar las series originales al lado izquierdo. En contraste, el segundo pulso tiene solo algunos peaks más leves en frecuencias bajas, lo cual se explica en la apariencia errática de su serie. Un cierto grado de conocimiento médico nos puede dar explicaciones del por qué de estas formas observadas.

3. Fast Fourier Transform

3.1. Complejidad Teórica

La transformada de Fourier discreta convierte una serie de tiempo donde las observaciones están equiespaciadas a una secuencia del mismo largo equiespaciada en el dominio de las frecuencias, correspondiente a la transformada de Fourier a tiempo discreto.

En su caso general dada una serie de tiempo real $x_1, \ldots, x_N$, la transformada de Fourier discreta está dada por $X_k = \sum_{n=1}^{N-1} x_n e^{-j\frac{2\pi}{N}kn}$ para $k \in \{1, \ldots, N\}$. Esto se traduce en el siguiente pseudocódigo.

Código 1:
Pseudocódigo de DFT

```
1 X_0, ..., X_N ← DFT(x_1, ..., x_N)
2     for k = 0 to N do
3         X_k = Σ_{n=1}^{N-1} x_n e^{-j 2π/N kn}
4     end for
5 end if
```

La complejidad de esto se calcula como sigue: tenemos un ciclo de largo N. Por otro lado, cada una de las sumas X_k corresponden a N evaluaciones. En conclusión, el algoritmo toma un tiempo del orden de N^2. Para series de tiempo largas, esto es muchísimo tiempo. Es por esto que Cooley y Tukey se proponen el siguiente algoritmo (llamado Fast Fourier Transform) para acelerar el cálculo de la transformada:

Código 2:
Pseudocódigo de FFT

```
1  X_0, ..., X_N ← FFT(x_1, ..., x_N)
2      if N = 1 then
3          X_0 ← x_0
4      else
5          X^par ← FFT({x_n ∈ x : n es par})
6          X^impar ← FFT({x_n ∈ x : n es impar})
7          for k = 0 to N/21 do
8              X_0, ..., X_{N/2 1} ← X_k^par + e^{-2πjk/N} X_k^impar
9              X_{N/2}, ..., X_N ← X_k^par + e^{-2πj(k+N/2)/N} X_k^impar
10         end for
11     end if
```

Notemos que gracias a la recursión, tenemos evaluaciones de costo N, sin embargo la profundidad es de largo $\log(N)$. Con esto, tenemos una complejidad en $O(N\log(N))$. Justifiquemos ahora por qué podemos emplear tal recursión. Primero notar que gracias a la estructura de la transformada, $X_{k+iN} = X_N \, \forall i \in \mathbb{Z}$. Entonces, en vez de hacer calculos redundantes, Cooley-Tukey divide la serie de manera iterativa para reducir el computo. Mas precisamente, aprovechamos que

$$X_k = \sum_{n=1}^{N-1} x_n e^{-j\frac{2\pi}{N}kn} = \sum_{m=1}^{\frac{N}{2}-1} x_{2m} e^{-j\frac{2\pi}{(N/2)}km} + e^{-i2/N} \sum_{m=1}^{\frac{N}{2}-1} x_{2m+1} e^{-j\frac{2\pi}{(N/2)}km}$$

Esto es lo que aprovecha el pseudocódigo y se aplica a las sub-series que generan las mismas divisiones.

3.2. Complejidad empírica

En esta subsección comprobaremos la complejidad teórica que se obtuvo en la parte anterior. Se implementaron en *python* ambos métodos mencionados. Primeramente se computan las transformadas con ambos métodos para las siguientes series: "coseno", que corresponde a una suma de cosenos: $\sum_{n=1}^{5} n\cos(n2\cdot 10\cdot \pi t)$; "coseno ruido" que equivale a lo anterior con un ruido $\sim \mathcal{N}(0, 0.5)$ en cada paso; y las series de pulso cardiaco hr_1 y hr_2 de preguntas anteriores.

Cronometramos la ejecución de ambos algoritmos truncando las series en tamaños multiplos de dos. En la figura 3 vemos los tiempos en cuestión con respecto al largo de las series. Además, ploteamos el promedio para DFT y FFT. Como referencia, ploteamos también funciones M_1N^2 y $M_2N\log(N)$ para M_1 y M_2 constantes.

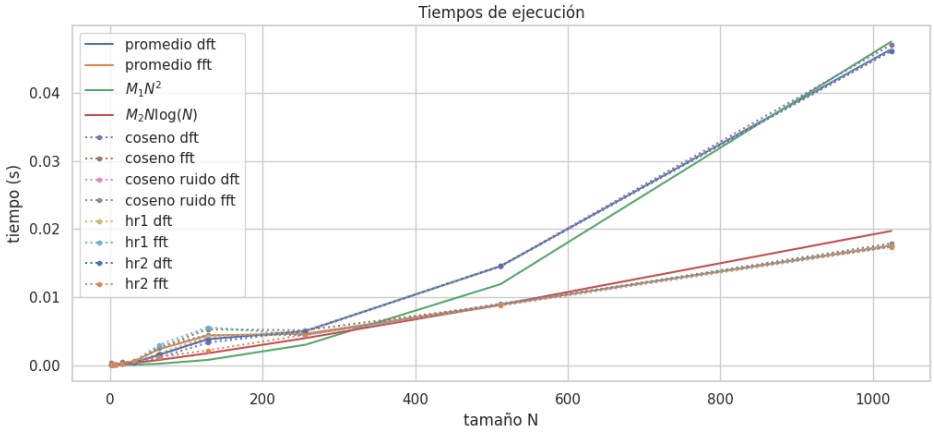


Figura 3: Tiempo de ejecución de DFT Y FFT versus largo de input.

Los resultados están dentro de lo esperado, pues se vislumbra claramente que el algoritmo FFT disminuye el tiempo de computo. Para graficar de mejor forma que las complejidades reportadas corresponden a las teóricas graficamos las mismas funciones de referencia junto con los tiempos de ejecución para "coseno ruido". La figura 4 nos muestra la dominación asintótica de nuestras funciones de referencia. Con esto se tiene que existen en cada caso las constantes M con las cuales podemos afirmar la pertenencia de ambas complejidades a las familias $O(N^2)$ y $O(N\log(N))$ para DFT y FFT respectivamente.

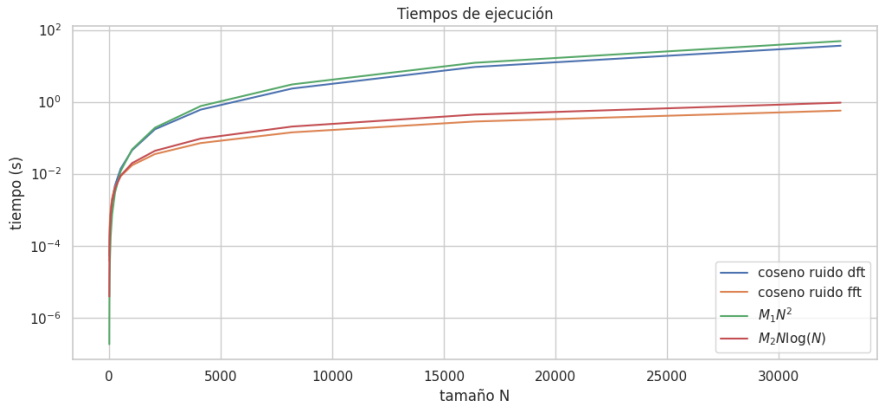


Figura 4: Tiempo de ejecución de DFT Y FFT versus largo de input (en escala logarítmica).