TRANSFORMADA DISCRETA DE FOURIER (DFT).

En esta práctica se estudiarán las características de esta transformada, tan importante en el área de procesado de señal. Se volverá a hacer hincapié en el significado de esta transformada así como los problemas que tiene así como posibles formas de solucionarlos. Las instrucciones de MATLAB para determinar la DFT y su inversa son, respectivamente, **fft y ifft**.

Seguidamente se va a demostrar el efecto del "solape temporal" que tiene el muestreo en frecuencia, que es dual al efecto del solape frecuencial al muestrear una señal temporal. Para ello se utilizará la señal $(a)^n \cdot u(n)$. Determina en primer lugar la transformada de Fourier de esta señal. A continuación muestréala para $w_k = \frac{2 \cdot \pi \cdot k}{N}$ (tanto a como N son parámetros del programa). Determina seguidamente la transformada inversa de estos coeficientes mediante el comando ifft. En este punto tienes que tener en cuenta la precisión finita de Matlab ya que la parte imaginaria de la ifft no se llega a anular (comprueba el orden de magnitud de dicha parte imaginaria) por lo que sólo hay que tener en cuenta la parte real. Representa, para diferentes valores de a y N, la señal obtenida y la original. ¿Qué ocurre para valores de a próximos a 1?.

El siguiente punto a estudiar es la resolución de una DFT. Hay que distinguir aquí dos tipos de resolución: la física y la "operacional". Si se tiene una señal de longitud L y se está trabajando con un periodo de muestreo T la mínima frecuencia que se podrá discernir será aquella cuyo periodo sea, precisamente, L·T. Sin embargo, en una DFT la resolución es igual a la frecuencia de muestreo dividida por el número de puntos. Tenemos entonces:

$$\Delta f\big|_{Física} = \frac{f_m}{L} \qquad\qquad \Delta f\big|_{DFT} = \frac{f_m}{N}$$

Veamos la diferencia entre ellas mediante un sencillo ejemplo. Ejecuta el siguiente *script:*

```
clear
close all
clc
f1=90;
f2=100;
f3=240;
f4=360;
fm=input('Frecuencia de muestreo (Hz) ');
tt=input('Duración del muestreo en ms ');
N=input('Longitud de la DFT aumentada ');
t=(fm*tt)*0.001;
x1=cos(2*pi*(f1/fm)*(0:t-1))+cos(2*pi*(f2/fm)*(0:t-1));
x2=x1+cos(2*pi*(f3/fm)*(0:t-1))+cos(2*pi*(f4/fm)*(0:t-1));
y=fft(x2);
t1=0:(fm/t):fm-(fm/t);
plot(t1,abs(y),'*-r');
yy=fft(x2,N);
hold on
t2=0:(fm/N):fm-(fm/N);
plot(t2,abs(yy),'+-k')
```

```
axis([0 fm/2 0 max([abs(y) abs(yy)])]);
grid
```

En este programa tomaremos 1KHz como frecuencia de muestreo con un intervalo de muestreo de 25 ms. En primer lugar consideraremos que el orden de la DFT es igual al de la longitud de la señal. ¿Son discernibles todas las componentes sinusoidales de la señal?. Fijando la longitud de la señal (esos 25 ms) aumentaremos el orden de la DFT, ¿podemos discernir las sinusoides de 90 y 100 Hz?.
Ahora aumentaremos el tiempo de muestreo a 100 ms. Ejecuta ahora el programa considerando la DFT del mismo orden que la longitud de la señal, ¿qué ocurre ahora?. Aumenta el orden de la DFT, ¿qué ocurre?.

Hemos visto los problemas que nos encontramos cuando la resolución física no es lo suficientemente pequeña para poder determinar una sinusoide. Sin embargo podemos tener otro problema: ¿qué ocurre cuando esta resolución física es suficiente pero la frecuencia de las componentes no es un múltiplo de la frecuencia unidad de la DFT $\Delta f = \frac{f_m}{N}$?. Representa gráficamente una sinusoide de frecuencia 10 Hz muestreada a 1000 Hz durante un periodo de 250 ms, fíjate en los puntos iniciales y finales de dicha onda. Determina ahora la DFT de esa señal con N=L, ¿qué ocurre?.

El efecto que aparece se conoce como "goteo espectral" y es debido a que las componentes sinusoidales no se corresponden con múltiplos de la frecuencia fundamental de la DFT. Otra forma de verlo es trasladarse al dominio temporal: la DFT es la representación de la señal x(n) que tiene periodo N: como hay una diferencia entre el punto inicial y final de dicha señal se produce un "salto" en esa señal periódica que la DFT intenta modelizar. Para evitar este problema se plantea el uso de una ventana; de forma indirecta ya la estamos utilizando pues al determinar una DFT "multiplicamos" por la ventana rectangular definida como:

$$w(n) = \begin{cases} 1 \text{ si } 0 \le n \le L-1 \\ 0 \text{ en otro caso.} \end{cases}$$

Una multiplicación en el dominio temporal (que es lo que se hace al enventanar) supone una convolución en el dominio frecuencial por lo que las características frecuenciales de la señal quedan, lógicamente, condicionadas por dicha ventana. Existen otras ventanas que tienden a evitar ese "goteo espectral". MATLAB tiene implementadas varias; las que estudiaremos serán la rectangular (boxcar), triangular (triang) la de Hamming, etc. Visualízalas y utiliza el comando freqz para determinar su respuesta en magnitud y fase. La relación entre la ganancia de los 2 primeros lóbulos da la capacidad de la ventana para reducir el goteo espectral mientras que la amplitud del primer lóbulo define la resolución de la ventana (a menor amplitud mayor resolución).

Una vez vistas sus características frecuenciales enventanaremos la última sinusoide antes de determinar su DFT. ¿qué se obtiene ahora?; ¿qué ocurre con todas laas componentes "ficticias" asociadas al goteo espectral? Por lo visto anteriormente estaríamos dispuestos a enventanar siempre una señal antes de determinar su DFT, pero, ¿no se paga nada a cambio?. La respuesta, lamentablemente ,es sí: perdemos resolución.

Para comprobarlo determina la DFT (con N=L) de la siguiente señal sin enventanar (¡¡realmente aplicamos la rectangular!!) y aplicando la ventana de Hamming:

$$x(n) = \cos\left(\frac{2 \cdot \pi \cdot f_1}{f_m} \cdot n\right) + \cos\left(\frac{2 \cdot \pi \cdot f_2}{f_m} \cdot n\right)$$

con f_1=100 Hz, f_2=110 Hz, f_m=1000 Hz,. n=0,.....99;

Una vez vista la DFT pasaremos a algunas de sus aplicaciones. La primera es la utilización de su algoritmo rápido (FFT), que presenta una carga computacional de $N \cdot \log_2 N$, para implementar operaciones cuyo coste es de N^2, como por ejemplo, la convolución de dos secuencias. Puede parecer que no se gana nada: representa la relación entre la carga computacional de la FFT y la convolución para valores de N que vayan de 1 a 2048. ¿qué te parece?.

Seguidamente comprobaremos la equivalencia entre una convolución circular de dos secuencias aumentadas con ceros y la convolución lineal de esas secuencias. Considera las siguientes señales: X1=[1,2,3]; X2=[-1 1]. Determina su convolución. Seguidamente determina la DFT de 4 puntos de cada una de ellas (has rellenado con ceros cada una de las secuencias), multiplícalas punto a punto y determina la inversa, ¿qué obtienes?. Si la DFT es de orden 5 se mantiene el resultado?; ¿y si es de orden 3?.

El último punto de esta práctica es el desarrollo de un sistema detector de alarmas de un edificio. Tenemos tres alarmas: incendio, robo y detector de llamada telefónica. Cada una de ellas se activa por una frecuencia característica (la frecuencia de muestreo del sistema es de 500 Hz). Estas frecuencias son: 150, 175 y 200 Hz. Desarrolla un programa en Matlab que implemente este detector usando el algoritmo de Goertzel. Los parámetros del sistema serán el término k del armónico a detectar y el orden de la DFT necesario para discernir esas frecuencias. La salida del programa debe ser un vector de 3 componentes indicando cada una de ellas si se activa la alarma (salida=1) o no se activa (salida=0).