SISTEMAS LINEALES
SEGUNDO CURSO GSC/G44

SESIÓN DE LABORATORIO 4 – CURSO ACADÉMICO 2014/2015

Transformación de señales al dominio de la frecuencia y filtrado con Matlab

1. Introducción

Los objetivos de esta sesión de laboratorio son los siguientes:

- Aprender a calcular la transformada de Fourier discreta (*"Discrete Fourier Transform"*, DFT) de una señal con el algoritmo FFT (*"Fast Fourier Transform"*) de Matlab.

- Aprender a calcular los coeficientes de un filtro digital con las funciones existentes en Matlab.

- Aprender a filtrar una señal con las funciones existentes en Matlab.

Cálculo de la FFT de una señal con Matlab

La transformada de Fourier de una señal discreta (DTFT) es una señal continua periódica de período 2π. A la hora de plantear el cálculo de la DTFT computacionalmente hay que tener en cuenta que en Matlab sólo es posible trabajar de forma discreta. Por ello, aunque la transformada es continua sólo se podrán obtener muestras discretas de la misma que, no obstante, pueden constituir una buena aproximación, si se toman suficientes. A esta transformada se le denomina transformada de Fourier discreta (*"Discrete Fourier Transform"*, DFT).

FFT es la abreviatura de *"Fast Fourier Transform"* y es un algoritmo eficiente para calcular la transformada de Fourier discreta (DFT) y su inversa. El rango de frecuencias cubierto por el análisis de la transformada de Fourier discreta depende de la cantidad de muestras recogidas y de la frecuencia de muestreo. Habitualmente se toman 512, 1024, 2048 o 4096 muestras.

En Matlab podemos usar la función `fft` para calcular la DFT, donde sólo hay que especificar la señal a la que se quiere hacer la transformada y el número de puntos. El comando `fft` de Matlab calcula la transformada para valores de 0 hasta fs, siendo fs la frecuencia de muestreo *("sampling frecuency"*). Se recomienda el uso del comando `fftshift` para desplazar y reordenar los valores obtenidos y poder ver la transformada desde –fs/2 hasta fs/2.

Filtrado digital con Matlab

Un filtro digital es un sistema lineal e invariante en el tiempo (LIT) que puede ser descrito por una ecuación diferencial de la siguiente forma:

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k]$$

donde *y[n]* es la señal a la salida del filtro, *x[n]* es la señal a la entrada al filtro y a_k y b_k son los coeficientes del filtro.

La función `filter` de Matlab filtra una señal de entrada a través de un sistema definido por una ecuación diferencial, que a su vez viene especificado a partir de los coeficientes a_k y b_k.

De entre los diferentes tipos de filtros digitales existentes, en esta práctica vamos a utilizar los denominados filtros FIR *("Finite-Impulse Response"*) que se caracterizan porque los coeficientes a_k son todos nulos excepto el a_o que vale 1. Nótese que, por tanto, el orden del filtro es M y su respuesta al impulso es:

$$h[n] = \begin{cases} b_n, & n = 0, 1, ..., M \\ 0 & resto \end{cases}$$

Los coeficientes b_k se pueden obtener usando la función `fir1` de Matlab de la siguiente forma:

```
B = fir1(M, Wn,'type');
```

donde `B` es el vector de coeficientes b_k, `M` es el orden del filtro, `Wn` es la frecuencia de corte y en `'type'` se especifica el tipo de filtro que quiere implementarse. Si en el tipo de filtro se especifica `'low'` se generan coeficientes para un filtro paso bajo, para un filtro paso alto se usa la opción `'high'`, un filtro paso banda con la opción `'bandpass'`, etc. Usando el comando `help fir1` de Matlab se pueden leer más detalles sobre esta función.

2. Transformación de las señales al dominio de la frecuencia

Ejercicio 2.1: Cómputo de la transformada de Fourier discreta de la señal 1 mediante la FFT

- **Cargue un archivo de audio**. Cargue el archivo **audio1.wav** usando el comando `wavread` de la siguiente forma:

```
[x1, fs1] = wavread('audio1.wav');
```

donde la variable `x1` contiene la señal de audio y `fs1` es la frecuencia de muestreo.

- **Modifique la frecuencia de muestreo de la señal de audio**. Remuestree la señal anterior con una frecuencia de muestreo de 16 kHz. Para ello puede usar el comando `resample`:

```
varOut = resample(varIn, P, Q)
```

que genera una variable `varOut` que es la variable `varIn` con un factor de diezmado `P` e interpolación `Q`, con respecto a la variable de entrada `varIn`.

Llame a la señal de audio resultante `voz` y escúchela usando el comando `soundsc`.

- **Cómputo de la transformada de Fourier mediante FFT**. Use los comandos `fft` y `fftshift` para calcular la transformada discreta de Fourier. Use 4096 puntos para la FFT.

- **Dibuje el espectro obtenido**: Genere una variable que recorra las frecuencias a las que se ha calculado la transformada y dibuje mediante `plot` el módulo del espectro de la señal. Para facilitar la visualización es conveniente representar dicho módulo en escala logarítmica `10*log10(variable)`.

Ejercicio 2.2. Cómputo de la transformada de Fourier discreta de la señal 2 mediante FFT

- Repita el ejercicio 2.1 para la señal de audio contenida en el fichero **audio2.wav**. Llame `piano` a la señal resultante del proceso de remuestreo.

Ejercicio 2.3. Desplazamiento de la señal 2 en frecuencia

- **Genere una coseno**. Genere una señal de coseno de la misma longitud que la señal `piano` (o la señal que quiera desplazar) y con frecuencia f0 = fs/4 Hz:

```
t = (1:length(piano))/fs;
f0 = fs/4;
coseno= cos(2*pi*f0*t);
```

- **Multiplique el coseno por la señal**. Multiplique la señal `piano` en el dominio del tiempo por el la señal `coseno`. Llame a la nueva señal desplazada `piano_desp`.

- **Calcule la FFT**. Calcule la FFT de `piano_desp`.

Determine teóricamente cuál es el efecto en el dominio de la frecuencia de multiplicar una señal por un coseno en el dominio del tiempo y compruebe

que se verifica experimentalmente mediante la comparación de los espectros de las señales `piano y piano_desp.`

Ejercicio 2.4. Mezclado de señales

- **Sume las señales**. Sume la señal desplazada en frecuencia `piano_desp` con la señal de audio `voz` y llame a la nueva señal `mezcla`.

- **Calcule la FFT**. Calcule la FFT de la señal `mezcla` y compárela con la del apartado anterior. Escuche la señal `mezcla` con el comando `soundsc`.

3. Filtrado

Ejercicio 3.1. Generación de un filtro paso bajo

- **Calcule los coeficientes del filtro**. Calcule los coeficientes de un filtro lineal FIR paso bajo usando el comando `fir1` de Matlab,

```
B = fir1(N, Wn,'low');
```

  donde `N` es el orden del filtro, `Wn` es la frecuencia de corte y `'low'` se refiere a la creación de un filtro paso bajo. Construya un filtro con `N`=1024 puntos y con una frecuencia de corte `Wn`=0.2.

- **Calcule la respuesta en frecuencia del filtro**: Aplique `fft` y `fftshift` a los coeficientes del filtro para calcular su respuesta en frecuencia.

- **Dibuje el módulo de la respuesta en frecuencia del filtro**: Recuerde que para calcular el módulo hay que usar el comando `abs` y que se visualiza mejor en escala logarítmica.

Ejercicio 3.2. Filtrado de la señal

- **Filtrado de la señal mezclada**. Filtre la señal mezclada anteriormente (`mezcla`) usando el comando `filter`:

```
Y = filter(B, A, X);
```

  donde `Y` es la señal a la salida del filtro, `B` son los coeficientes calculados anteriormente, `A`=1 y `X` es la señal a la entrada del filtro.

  Denomine a la señal filtrada (salida del filtro) como `mezcla_filtrada` y y calcule y represente el módulo de su DFT. Escuche la señal `mezcla_filtrada` con el comando `soundsc`. Dicha señal debería ser similar a la señal `voz`. Explique el motivo.