FUNCIONES y TIPS para MATLAB ⇒ usando FUNCIONES del toolbox_master

SERIE EXPONENCIAL DE FOURIER

Nº de coefic.

⇒ OBTENER coefic. CN ⇒ [Cn = SEF(t, ft, T0, N)]

⇒ Reconstruir función ⇒ [f = ISEF(t, Cn, T0)]

⇒ POTENCIA (tiempo) ⇒ [p_t = POTENCIA(ft, T0, dt)]; ENERGÍA (tiempo) ⇒ [e = ENERGÍA(ft, dt)]

⇒ POTENCIA (frec) ⇒ [p_f = POTENCIA_SEF(Cn)]

TRANSFORMADA DE FOURIER CONTINUA (TFTC)

⇒ OBTENER ESPECTRO Fw y w ⇒ [[Fw, w] = TFTC(t, ft, wmax, dw)]

TIP: si sabemos que la señal tiene ALTAS frecuencias, NO PONER un dw MUY CHICO

⇒ OBTENER ENERGÍA del espectro ⇒ [E_w = ENERGÍA_TFTC(Xw, dw)]

⇒ OBTENER ENERGÍA en un PORCENTAJE ⇒

[[i_porcent, i_idx] = ENERGIA_TFTC_INT(Fw, dw, E_tiempo, porcentaje)]

TIP: para hallar la F_C, la E de ese INTERVALO y el GRAFICO hasta ese INTERVALO:

```
Fc = w(i_idx(2))./(2*pi)   % w es el que devuelve TFTC, i_idx(2) es el VALOR de Wc en el VECTOR w
interv_w = w(i_idx(1):i_idx(2));
interv_Fw = Fw(i_idx(1):i_idx(2));
E_interv = ENERGIA_TFTC(interv_Fw, dw)
plot(interv_w, abs(interv_Fw));
```

DENSIDAD ESPECTRAL

⇒ OBTENER DEE (Sxx) ⇒ [[Rxx, tau] = xcorr(x, x)
[Sxx, w] = TFTC(tau*dt, Rxx*dt, wmax, dw)]

⇒ DEE de la salida (Syy) también puede obtenerse como: [Syy = (abs(Hw).^2).*(abs(Xw).^2)]

ESTOCÁSTICOS EN FRECUENCIA

1 FILA — n/dt COLUMNAS — DEP

⇒ Generar RUIDO BLANCO ⇒ [xr = wgn(1, (n/dt), S0)]

SEÑAL

⇒ ADICIONAR RUIDO BLANCO A SEÑAL ⇒ [xr = awgn(x, 1, 'measured')]

⇒ Calcular POTENCIA con Ryy(0) ⇒ [pot = max(Ryy)*dt]

⇒ Hallar H(w) DEL SIST. con el RUIDO ⇒ [[Ryx, tau] = xcorr(y, xr) % CORR. CRUZADA RUIDO, SALIDA.
[Gyx, w] = TFTC(tau*dt, Ryx*dt, wmax, dw)
Hw = Gyx./Gxx → es la DEP del RUIDO (la xcorr de xr)]

MUESTREO ⇒ NO INCORPORAMOS NUEVAS FUNCIONES, pero van algunos TIPS.

TIPS:

⇒ TOMAR dt tal que 1/dt de una $f_S \geq 2 f_{max}$ ⇒ $dt \simeq T_S$

⇒ ∴ conviene tomar $\left[dt \simeq \frac{1}{\text{VALORES MAYORES a } f_S}\right]$ ⇒ $\left[dt \simeq \text{VALORES MENORES a } T_S\right]$

P/ MUY ALTAS frecuencias:

⇒ Agrandar los INTERVALOS de tiempo si los valores del espectro de MODULO DAN MAS BAJOS
(las señales deben estar definidas en un INTERVALO que las represente en su TOTALIDAD)
(los agrandamos para no PERDERNOS con INTERVALOS de VALORES MUY CHICOS por ser ALTA frec)

⇒ RECORDAR p/ SEÑALES de MUY ALTA f tomar dw del ORDEN de 1, 10, 100, etc

TFTD ⇒ SE APLICA A SEÑALES DE T. DISCRETO, pero el esp. es CONTINUO
⇒ YA NO SE USA Wmax NI dw

⇒ OBTENER ESPECTRO Xw y w ⇒ `[Xw, w] = TFTD(n, Xn, ciclos)` (ciclos: opcional, si no se pone solo toma 1 ciclo)

⇒ OBTENER dw (pues no la definimos) ⇒ `dw = w(2) - w(1)` ⇒ SI USAMOS LA FUNCIÓN ENERGIA_TFTC debemos ponerle ese dw

⇒ OBTENER ENERGÍA DEL ESPECTRO ⇒ `E_w = 1/(2*pi) * sum((abs(Xw)).^2) * dw`

DFT y FFT ⇒ p/ OBTENER ESPECTROS DISCRETOS

PARA No (PERIODO DISCRETO ⇒ CANT. DE MUESTRAS por ciclo) PEQUEÑO uso DFT:

```
[Xw, w] = DFT(n, Xn)
```

PARA No GRANDE uso FFT:

```
Xk = fft(Xn)/N0
```

⇒ En este caso N_0 = length(Xn), siendo Xn discreto

TIP: N0 IDEALMENTE debe ser $= 2^m$, pero si es cercano NO PASA NADA

⇒ GRABAR AUDIO Y GUARDARLO

```
Fs = 11000; nBits = 16; nChannels = 1;
ID = -1;
recobj = audiorecorder(Fs, nBits, nChannels, ID);
disp('start speaking')
recordblocking(recobj, segundos);
disp('End of recording');
x = getaudiodata(recobj); % GUARDADO
```

(líneas 1–6: GRABACIÓN)

⇒ FILTRAR (dominio Ω):

• TOMAR num y den según coeficientes de

$$H(\Omega) = \frac{\sum_{k=0}^{N} b_k e^{-jk\Omega}}{\sum_{k=0}^{N} a_k \cdot e^{-jk\Omega}}$$

SIENDO N el RETARDO (ej: x[n-N]) — NO CONFUNDIR con N0

ej: N lo podemos definir así:

```
N = round(delay * Fs); % delay en s
```

si $H(\Omega) = e^{-j\Omega N} / 1 + \alpha \cdot e^{-j\Omega N}$

```
num = [zeros(1,N) 1]; den = [1, zeros(1,N-1), α];
y_filt = filter(num, den, x)
audiowrite('audiofiltrado.wav', y_filt, Fs); % GUARDA el audio
```

⇒ filter es para FILTROS DIGITALES!

⇒ REPRODUCIR AMBOS (SIN que se SOLAPEN)

```
sound(g*x, Fs); % g: GANANCIA
duracionx = lenght(x)/Fs; % durac. en s
pause(duracionx + 1);
sound(g * y_filt, Fs);
```

⇒ señales en el tiempo:

```
ts = 1/Fs;
tx = 0:ts:duracionx-ts;
```

⇒ espectro en frec. pasando N0 a frecuencia

```
N0 = Fs * duracionx; % MUESTRAS por PERIODO (TODO el AUDIO)
Frec = (0:N0-1) * Fs/N0; % Fs/N0 = F0
x_fft = fft(x)/N0; % luego → stem(Frec, abs(x_fft));
```

⇒ LEER AUDIO

```
[x, Fs] = audioread('nombredelaudio.mp3');
```

TRANSFORMADA DE LAPLACE
PARA CONTINUAS

EN LAPLACE VA COMO POTENCIAS POSITIVAS

⇒ num y den según coeficientes en dominio S ⇒ ej: a: $H(S)=\frac{1}{S^2}$; b: $H(S)=\frac{S^2+1}{(S+2)(S^2+S+1)}$

con a: `num = [0 0 1]; den = [1 0 0]` % $H(S)=0S^2+0S+1/S^2+0S+0$

con b: `num = [1 0 1]; den = conv([1 2][1 1 1]);` % uso conv p/no hacer distrib.

⇒ DIAGRAMA DE POLOS Y CEROS

```
figure
zplane(num,den);
```

o sino ⇒ `pzmap(x);`

⇒ POLOS Y CEROS por comando

```
x = tf(num,den);
polos = pole(x)
ceros = zero(x)
```

⇒ tf nos da la TL

TIP: tf NO ME DEVUELVE un VECTOR ∴ para graficar X en función de w debo:

⇒ OBTENER ESPECTRO DE LA TL y GRAFICAR:

2 FORMAS → en dB : definir vector w y luego ⇒

```
xw = freqs(num,den,w);
plot(w,abs(xw));
```

→ SIN UNIDADES :

```
[xw,w] = freqs(num,den);
plot(w,abs(xw));
```

⇒ OBTENER SEÑAL TEMPORAL A PARTIR de la TL (CONTINUA)

- siendo x = tf(num,den) ⇒

```
[xt,t] = impulse(x);
plot(t,xt)
```

- También podemos definir t ⇒

```
t = 0:0,01:10;
xt = impulse(x,t);
plot(t,xt)
```

SI LA TL es H(S):

RESP. IMPULSIONAL h(t):

```
ht = impulse(H,t);
```

RESP. INDICIAL g(t):

```
gt = step(H,t);
```

⇒ OTRA FORMA DE DEFINIR TL sin num y den, pero con la EXPRESIÓN ANALÍTICA

ej: usando b:

```
S = tf('S');
Hs = S^2 +1 /(S+2)(S^2 +S+1);
[ht,t] = impulse(Hs);
```

⇒ TENIENDO num y den de la H(S) de un filtro, obtener SALIDA p/ cierta ENTRADA

```
y = lsim(Num,Den,x,t);
```

⇒ es como filter pero para FILTROS ANALÓGICOS (recordar que LAPLACE es p/CONTINUOS)

lsim me da la traspuesta ∴ si la quiero graficar debo hacer:

```
y_filt = transpose(y);
```

TIPS: . RECORDAR que en LAPLACE trabajamos con SEÑALES CONTINUAS ∴ usamos TFTC

TRANSFORMADA Z ⇒ P/DISCRETAS

EN Z VA COMO POTENCIAS NEGATIVAS

⇒ num y den según COEFICIENTES en dominio Z ⇒ $H(Z)=\frac{b(1)+b(2)Z^{-1}+\ldots+b(n+1)Z^{-n}}{a(1)+a(2)Z^{-1}+\ldots+a(m+1)Z^{-m}}$

ej: $H(Z)= Z+1/(Z-0,5)(Z^2-0,5Z+0,25)$

```
num = [0 0 1 1]; den = conv([1 -0,5],[1 -0,5 0,25]);
```

⇒ OBTENER TZ ⇒ `xz = tf(num,den,1);` 1 es para Z

⇒ OBTENER ESPECTRO DE TZ ⇒ `[xw,w] = freqz(num,den);` o `xw = freqz(num,den,w)` es la Ω NORMALIZADA ∴ w = -pi:0,01:pi

⇒ OBTENER SEÑAL DISCRETA A PARTIR de TZ ⇒

```
[xn,n] = impz(num,den,n);
```

n → MUESTRAS

o sino definir n, luego: `xn = impz(x,n)`

TENIENDO H(Z):

RESP. IMPULSIONAL h[n] ⇒ `[hn,n] = impz(num,den,n)`

RESP. INDICIAL g[n] ⇒ `[gn,n] = stepz(num,den,n)`