Tratamiento Digital de Señales

Práctica 2

Muestreo de señales en tiempo continuo

Actividades Preliminares

Las actividades preliminares se deberán entregar en un folio manuscrito al comienzo de la práctica de laboratorio.

Actividad 1. Muestreo de sinusoides

Dada la señal continua, $x(t)=\cos(2\pi 200t) + \cos(2\pi 600t)$. Si se muestrea con *1 KHz*, escriba la expresión analítica de $x[n]$, y de $X(e^{j\omega})$. Dibuje además $|X(e^{j\omega})|$.

Actividad 2. Interpolación y diezmado

Dibuje el diagrama de bloques que permite transformar una secuencia, $x[n]$, que corresponde a muestras equiespaciadas tomadas a 8KHz de una señal continua, en otra secuencia, $y[n]$, que corresponde a muestras equiespaciadas de la misma señal tomadas a 12 KHz.

Introducción a la práctica

El muestreo de una señal en tiempo continuo $x_c(t)$ es el proceso mediante el que se obtiene una señal discreta $x_d(n)$ que verifica:

$$x[n] = x_c(nT_S), \quad n \in Z$$

donde T_s es el periodo de muestreo, medido en segundos. $F_s = \frac{1}{T_s}$ es la frecuencia de muestreo medida en *Hz* o muestras/segundo. Por tanto, la secuencia de números $x_d(n)$ son las medidas del valor de la señal $x_c(t)$ en los instantes temporales nT_s.

Los teoremas de muestreo establecen los valores adecuados de T_s para que la secuencia de medidas represente fielmente a la señal $x_c(t)$. En el caso particular en que la señal $x_c(t)$ es de banda limitada, con ancho de banda F_0, el periodo de muestreo mínimo debe ser $T_s = \frac{1}{2F_0}$.

En el entorno MATLAB el muestreo de una señal $x_c(t)$ definida en el intervalo temporal $[T_1\ T_2]$, y con un periodo de muestreo T_s se puede realizar con las siguientes líneas de código

```
t=T1:Ts:T2;
x=xc(t);
```

El vector t contiene los instantes de muestreo (rejilla temporal). En la segunda línea se calculan los valores de $x_c(t)$ en los distintos instantes de muestreo.

Una alternativa al código anterior es basarse en el número de muestras que se desea que tenga la secuencia y utilizar la transformación *t=nT*:

```
n=0:255;
x=xc(n*T);
```

Ejercicios de la práctica

1. Muestreo de sinusoides

Considere el siguiente conjunto de sinusoides:

$$x(t) = A\cos(2\pi F_0 t),\ F_0 = 200,\ 850,\ 1850,\ 3800\ \ Hz,\ \ A = 1$$

1.1. Muestree las señales $x(t)$ con una frecuencia de muestreo $F_s = 8000\,Hz$. Dibuje (con plot(. . .), dividiendo la pantalla en cuatro mediante subplot(2,2,x), x=1, . .., 4) un segmento de duración $T = 10\ ms$.

Una solución para este sub apartado podría ser:

```
% Datos
Frecuencias=[200 850 1850 3800]; A=1;
Tv=10e-3;
Fs=8000; Ts=1/Fs;

% Rejilla temporal para el muestreo
t=0:Ts:Tv;

% Cálculo y dibujo de las sinusoides
n=1;
for F0=Frecuencias
   subplot(2,2,n)
   plot(t,A*cos(2*pi*F0*t))
   title(['cos(2\pi' num2str(F0,4) 't)']), xlabel('t'), grid
   n=n+1;
end
subplot
```

Observe como con una estructura de repetición for . . . end calculamos y pintamos las cuatro señales. También observe que se cierran los cálculos con el comando subplot para deshacer la partición de la ventana gráfica.

1.2 Repita el apartado 1.1 pero con una frecuencia de muestreo $F_s = 6000\,Hz$ y el siguiente conjunto de señales:

$$x(t) = A\cos(2\pi F_0 t),\ F_0 = 200,\ 850,\ 1850,\ 5800\ \ Hz,\ \ A = 1$$

1.3 Repita el apartado 1.1 pero con una frecuencia de muestreo $F_s = 6000\,Hz$ y el siguiente conjunto de señales:

$$x(t) = A\sin(2\pi F_0 t),\ F_0 = 200,\ 850,\ 1850,\ 5800\ \ Hz,\ \ A = 1$$

Debe analizar con detenimiento los resultados gráficos, e intentar comprenderlos con ayuda de la teoría de muestreo. Concretamente:

a.- ¿Por qué aparece un efecto parecido a una modulación de amplitud?
b.- ¿Por qué hay un cambio de signo de una de las señales en los apartados 1.2 y 1.3?

2. Sistema interpolador

En el fichero tds.mat dispone de la señal correspondiente a la frase "**tratamiento digital de señales**", con una frecuencia de muestreo de 8000 Hz. Sin embargo, suponga que tenemos un reproductor que sólo admite una frecuencia de muestreo de 24000 Hz e intente reproducir la señal con los comandos:

```
p = audioplayer(int16(tds), 24000);
play(p);
```

Al oír la señal se hace evidente la necesidad de interpolar nuestra grabación por un factor L=3 (24000/8000=3).

Para realizar esta interpolación se sigue el esquema correspondiente al sistema interpolador, es decir primero la expansión y luego el filtrado.

Nota: en vez de poner cada sub apartado en una celda, inserte el comando pause al final del código de cada sub apartado. De esta forma cuando ejecute la celda del apartado 2 podrá avanzar por los sub apartados presionando la tecla INTRO.

Expansor

Expanda la señal tds por un factor L=3. Una forma de realizar una expansión en MATLAB es:

```
tdsL = zeros(L*length(tds), 1);

tdsL(1:L:length(tdsL)) = tds;
```

Dibuje, con stem(. . .), un segmento de 30 puntos de tdsL a partir del índice n=3750, y compruebe que hay "L-1 ceros entre muestras". Escuche la señal tdsL y aprecie el ruido "metálico y de alta frecuencia" debido a la expansión.

El comando MATLAB para oir tdsL sería:

```
p=audioplayer(int16(tdsL), 24000);

play(p)
```

Filtro interpolador

Para completar la interpolación se filtra la señal expandida mediante un filtro paso bajo ideal con frecuencia de corte $\omega_c = \frac{\pi}{L}$ y amplitud L. La respuesta impulsiva de este filtro es

$$h_i(n) = \begin{cases} L\dfrac{sen(\omega_c n)}{\pi n} & , \ n \neq 0 \\ 1 \, , \ n = 0 \end{cases}$$

Este filtro es ideal y por tanto no realizable.

Para construir una aproximación de este filtro se va a utilizar el truncando la respuesta impulsiva entre -100 y 100. Con MATLAB sería:

```
n = -100:100;

h = L*sin(pi/L*n)./(pi*n);

h(isnan(h)) = 1;
```

Explique la operación del tercer comando MATLAB.

Dibuje en primer lugar con plot(. . .) toda la respuesta impulsiva con el eje horizontal entre -100 y 100, y después sólo 41 puntos en torno al máximo con stem(. . .). Observe como h(Lk)=0.

Filtrado de la salida del expansor

Filtre la señal expandida tdsL con el comando

```
tdsLi = filter(h, 1, tdsL).
```

Escuche la señal tdsLi y compruebe que se ha suprimido el ruido “metálico y de alta frecuencia”.

"Corte y pegue" las siguientes líneas de código en su *script*:

```
% Visualización
n=3750:3779;
plot(n,tdsLi(n+N),'b')
hold on
stem(n,tdsLi(n+N),'g')
stem(n,tdsL(n),'r')
hold off
grid, xlabel('n')
legend('tds(t)','tdsLi(n)','tdsL','Location','NorthWest')
```

Estas líneas de código representan un segmento de tdsL, tdsLi y la señal original $tds(t)$.

(+N en la segunda y cuarta líneas del código compensa el retardo de N muestras que introduce el filtro interpolador, ¿cuánto vale ese retardo?). Se aprecia claramente que ambas señales coinciden en los índices kL y que el filtro interpolador "mueve" los ceros que introduce el expansor hasta la posición correcta sobre la gráfica de $tds(t)$.

3. Sistema Diezmador

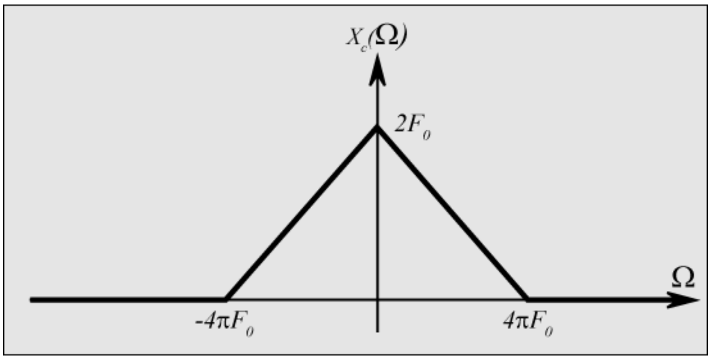
La realización del diezmado de una señal es la tarea que realiza el sistema diezmador que está compuesto por dos subsistemas conectados en cascada: El filtrado y el compresor

3.1 Señal de entrada

Con el fin de comprobar el funcionamiento del sistema diezmador genere la siguiente señal:

$$x_c(t) = \begin{cases} \left(\dfrac{sen(2\pi F_0 t)}{\pi t} \right)^2, & t \neq 0 \\ 4F_0^2, & t = 0 \end{cases}$$

es de banda limitada a $2F_0$ Hz, siendo su espectro el de la siguiente figura:



Considere la secuencia de duración finita (2 segundos):

$$x(n) = x_c(nT_s)\ ,\ \ -1 \le nTs \le 1$$

donde $T_s = \dfrac{1}{12F_0}$. Esta secuencia se puede considerar de banda limitada (aproximadamente ya que estamos muestreando en una ventana de 2 segundos solamente), con $\omega_0 = \dfrac{\pi}{M}$, por lo que su "redundancia" es de $M-1$ muestras de cada M. **¿Cuánto vale M ?**

Suponiendo $F_0 = 1\ KHz$, calcule la señal $x(n)$. Dibuje con el comando stem(. . .).la señal calculada, pero solo en el rango de índices $[-30 \quad 30]$.

Sugerencias: a) calcule una rejilla temporal con los instantes de muestreo. b) evite el resultado NaN que se produce al dividir por cero con el mismo procedimiento que en el apartado 2.2.

3.2 Sistema Diezmador

Programe el sistema diezmador de orden 2 cuya secuencia de entrada es *x[n]*. Dibuje los resultados utilizando el comando stem tanto de la salida del filtro como del compresor. Para comparar los resultados seleccione 40 muestras alrededor del máximo.

Recuerde que el filtro es un paso bajo ideal con frecuencia de corte $\omega_c=\pi/M$ y amplitud 1. Para construir este filtro puede utilizar el mismo código del apartado anterior (Observe que la ganancia del filtro sistema interpolador es L mientras que la del sistema diezmador es 1)

Para la operación de compresión don un factor de 2emplee el siguiente código Matlab:

```
M=2;
xc=xf(1:M:length(xf));
```

Siendo *xf*, la salida del filtro.

Compruebe que el resultado obtenido coincide con el resultado esperado

4 Estudio del expansor y del compresor en el dominio de la frecuencia

Utilice la misma secuencia *x[n]* del apartado anterior

4.1 Cálculo de $X(\omega)$

Calcule y dibuje, en escala lineal, el módulo de $X(\omega)$ en el intervalo $\left[0 \quad 2\pi\right]$. Para el cálculo de $X(\omega)$ emplee:

```
[X w]=freqz(x,1,'whole');
```

Observe el espectro resultante y compruebe que el ancho de banda es $\frac{\pi}{M}$. Mida el máximo de $\left|X(\omega)\right|$ y verifique que **su valor es correcto**.

4.2 Espectro de una señal expandida

Considere la señal

$$x_e(n) = \begin{cases} x\left(\dfrac{n}{L}\right), & n = \dot{L} \\ 0, & resto \end{cases}$$

Calcule y dibuje, en escala lineal, el módulo de $X_e(\omega)$ en el intervalo $\left[0 \quad 2\pi\right]$. Para la operación de expansión emplee:

```
L=3;
xe=zeros(L*length(x),1);
xe(1:L:length(xe))=x;
```

Compruebe que el resultado obtenido coincide con el resultado esperado, teniendo en cuenta la operación en frecuencia del expansor discreto.

4.3 Espectro de una señal comprimida

Considere la señal

$$x_c(n) = x(2n)$$

Calcule y dibuje el módulo de $X_d(\omega)$ en el intervalo $\lfloor 0 \quad 2\pi \rfloor$.

Para la operación de compresión emplee:

```
M=2;
xc=x(1:M:length(x));
```

Compruebe que el resultado obtenido coincide con el resultado esperado, teniendo en cuenta la operación en frecuencia del compresor discreto. ¿Se ha producido “aliasing” al comprimir la señal $x(n)$? ¿Por qué?