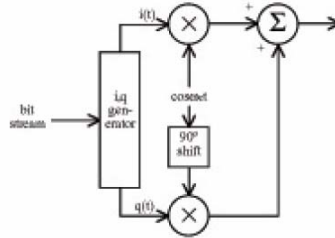


# MAS 160/510 – Boletín de problemas n° 9

No se aceptan trabajos entregados fuera de plazo

## 1. Modulación digital



El sistema cuaternario de creación de claves con cambio de fase de la figura se utiliza para transmitir datos a través de un canal (con ruidos). El generador i,q. de este sistema es:

Par de bits	i	q
(0,0)	1	0
(0,1)	0	1
(1,0)	-1	0
(1,1)	0	-1

La frecuencia de modulación de este sistema,  $\omega_0$ , es  $2\pi \frac{1}{32}$  y cada símbolo sólo debe tomarse para un periodo de la frecuencia de modulación.

- (a) Escriba una función de MATLAB **f32qpsk(x)** que se reciba como una matriz de bits y devuelva la señal modulada generada por el sistema anterior. Cree la matriz con un cero si es de longitud impar. Calcule la salida de la siguiente matriz de bits

*[0 1 1 1 1 0 1 0 0 1 1 1 0 0 1 0]*

- (b) Con la **dft**, determine la ubicación del plano complejo de los puntos de constelación del sistema.
- (c) Determine un método para recibir la señal final del sistema, esto es, cree una función de MATLAB **f32qpsk(x)** que recoja la señal modulada y que devuelva una matriz de bits. Para ello, deberá decidir qué segmento de muestra de los 32 existentes corresponde a cada símbolo. Pista: Una forma de conseguirlo es calcular el punto de constelación más próximo en el plano complejo para la frecuencia correspondiente. Demuestre este método para la matriz de bits del apartado (a).
- (d) Con la función **rand**, agregue ruido al sistema en forma de  $\pm\alpha$ . ¿Qué tasa de error (porcentaje de símbolos incorrectos en el número total de símbolos) del sistema se transmite en la matriz de bits del apartado (a) con un error  $\alpha = 0,1$ ,  $\alpha = 0,5$  y  $\alpha = 5,0$ ?

## 2. Diseño de filtros

Diseñe un filtro digital de fase lineal FIR que se aproxime a la respuesta en frecuencia ideal:

$$H_d(\omega) = \begin{cases} 1, & \text{para } |\omega| \leq \frac{\pi}{6} \\ 0, & \text{para } \frac{\pi}{6} < |\omega| \leq \pi \end{cases}$$

- Determine los coeficientes del filtro de 25 pasos basado en el método de ventana con ventana rectangular.
- Determine y represente la magnitud y la respuesta en fase del filtro.
- Repita los apartados (a) y (b) utilizando la ventana de Hamming.
- Repita ahora los apartados (a) y (b) utilizando la ventana de Barlett.

No olvide NOMBRAR TODOS los ejes correctamente (le serán de utilidad los comandos **xlabel**, **ylabel** y **title** de MATLAB). Para encontrar la respuesta en frecuencia puede buscar muestras de la transformada de Fourier con el comando **fft** o con la función **freqz**. Incluya todas las representaciones y el código.

## 3. Transformada discreta de Fourier y filtrado

Asuma que tiene las siguientes señales continuas en el tiempo  $y_1(t) = \sin 2\pi 10t$  y  $y_2(t) = \sin 2\pi 20t$  que se han muestreado a  $f_s = 50\text{Hz}$  para obtener  $y_1[n]$  e  $y_2[n]$  y para transmitir a través de una canal. Desafortunadamente, el canal no está bien diseñado y en el extremo de recepción la señal que se recibe es  $y[n] = y_1[n] + y_2[n]$ . Su trabajo consistirá en intentar recuperar  $y_1[n]$  a partir de la señal recibida  $y[n]$ . En este ejercicio le ayudaremos a diseñar un filtro selectivo de frecuencias para filtrar la señal extraña e intentar recuperar muestras de la señal  $y_1[n]$ .

En primer lugar, obtendremos muestras de 1 minuto de las señales continuas en el tiempo. Para ello las muestrearemos a 50 Hz:

```
f1=10;
f2=20;
Fs = 50;

n=[0:1/Fs:1]

y1 = sin(2*pi*f1*n);
y2 = sin(2*pi*f2*n);
```

Para analizar el contenido de la frecuencia de una señal, podemos utilizar el comando **fft** de MATLAB. Analicemos 256 puntos de la transformada discreta de Fourier de  $y_1[n]$  escribiendo lo siguiente en la línea de comandos:

```
>> Y1 = fft(y1,256);
```

Con esto recuperaremos 512 muestras de la TDF analizada en el rango  $0 - 2\pi$ . Para visualizar la magnitud de la transformada de Fourier en  $0 - \pi$ , introduzca los siguientes comandos:

```
>> omega = [0:1:127]/128;
>> plot(omega,abs(Y1(1:128)))
```

Así, representaremos la mitad (128) de los puntos de muestra y escalaremos el eje  $\omega$  para que  $\pi$  corresponda a 1.

- (a) Represente la magnitud de las transformadas discretas de Fourier de  $y_1[n]$ ,  $y_2[n]$ ,  $y[n] = y_1[n] + y_2[n]$  calculando para ello 256 puntos. Represente sus resultados con  $\omega$  en el rango  $0 - \pi$  (o  $0 - 1$  si se divide entre  $\pi$ ).
- (b) Al encontrar los picos de la magnitud de la transformada de Fourier, calcule el componente principal de frecuencia de cada señal. ¿Ha obtenido los resultados que esperaba? Brevemente, argumente su respuesta teniendo en cuenta la frecuencia de las señales continuas en el tiempo y la velocidad a la que se muestrean.

A continuación, vamos a filtrar  $y[n]$  con un filtro de paso bajo para intentar recuperar  $y_1[n]$ . En concreto, vamos a utilizar un filtro de paso bajo Butterworth de cuarto orden para filtrar  $y[n]$ . Por suerte, MATLAB proporciona un método rápido para obtener los coeficientes del filtro de un Butterworth de paso bajo. Si escribe

```
>> help butter
```

Obtendrá lo siguiente:

```
BUTTER Butterworth digital and analog filter design.  
[B,A] = BUTTER(N,Wn) designs an N'th order lowpass digital  
Butterworth filter and returns the filter coefficients in length  
N+1 vectors B and A. The cut-off frequency Wn must be  
0.0 < Wn < 1.0, with 1.0 corresponding to half the sample rate.
```

Un filtro de paso bajo «pasarà» frecuencias hasta la frecuencia de corte ( $\omega_n$ ) y atenuará aquellas frecuencias que estén por debajo.

- (c) Asumiendo que queremos colocar la frecuencia de corte a 15 Hz, ¿qué valor se debería utilizar para  $\omega_n$ ?
- (d) Con el valor encontrado para  $\omega_n$ , obtenga los coeficientes del filtro Butterworth de cuarto orden. Utilizando el comando **filter** y los coeficientes, filtre la señal  $y[n]$ . Sea  $z[n]$  la salida del filtro. Represente  $z[n]$ . En el mismo gráfico, represente  $y_1[n]$ . ¿Qué diferencias observa? (Tal vez quiera utilizar la función **plot** en lugar de **stem**. También puede usar colores distintos u otros tipos de línea para visualizar las dos señales).
- (e) Represente la DFT de 256 puntos de  $z[n]$  tal como lo hizo en el apartado (a). Compare las representaciones que obtuvo en dicho apartado. Asegúrese de mantener el mismo eje horizontal en todas las representaciones para, así, poder compararlas.
- (f) Al atenuar las frecuencias superiores a 15 Hz, hemos filtrado de forma eficaz el componente de 20 Hz de  $y[n]$  y hemos recuperado  $y_1[n]$  de forma aproximada. Si  $Y_1(\omega)$  y  $Z(\omega)$  representan las DFT de  $y_1[n]$  y  $z_1[n]$ , represente  $|Y_1(\omega) - Z(\omega)|$  para visualizar la diferencia entre sus transformadas de Fourier.

Compruebe que ha incluido todas las representaciones y el código fuente.

## 4. Reflejo y convolución

En MATLAB, cree los siguientes vectores:

```
>> n = 0:127;
>> x1 = cos(2*pi*32/128*n);
>> x2 = cos(2*pi*56/128*n);
>> xm = cos(2*pi*16/128*n);
```

- Ahora reflejaremos (multiplicaremos)  $x_1$  por  $x_m$ . Sabemos que la convolución en el dominio del tiempo tiene como resultado la multiplicación en el dominio de la frecuencia. Del mismo modo, el reflejo (multiplicación) en el dominio del tiempo se corresponde con la convolución en el dominio de la frecuencia. Demuestre este hecho mediante la representación de las DFT de  $x_1$ ,  $x_m$  y  $x_1 \cdot x_m$  en tres representaciones distintas dentro de una misma página.
- Ahora agregue las señales  $x_1$  y  $x_2$ . En las tres representaciones, de nuevo represente las DFT de  $x_1$ ,  $x_2$  y  $x_1 + x_2$ . ¿Obtiene los resultados que esperaba?
- Refleje ahora  $x_1 + x_2$  con  $x_m$ . Represente la DFT resultante. Dado que el reflejo en el dominio del tiempo se corresponde con la convolución en el dominio de la frecuencia, esperamos que el resultado sea la convolución de  $x_1 + x_2$  y  $x_m$ , ¿verdad? ¿Es ése el resultado? Explique el fenómeno que ha tenido lugar. Este fenómeno recibe dos nombres distintos: ¿cuáles son?

## 5. Creación de máscaras psicoacústicas sencillas

La siguiente función de MATLAB lleva a cabo una sencilla prueba psicoacústica. Crea un ruido con banda limitada centrado en 1000 Hz y crea también un senoide. Posteriormente, reproduce el ruido por separado y, más tarde, el ruido combinado con el senoide. Pruebe con distintos valores de  $f$  y  $A$  para ver si puede detectar el senoide. Represente varios valores de  $f$  con respecto a  $A$  para calcular la curva simple de máscara.

```
function mask(f,A)
% MASK lleva a cabo una prueba de máscara psicoacústica sencilla creando
% un ruido con banda limitada de unos 1000 Hz y un senoide único con
% frecuencia f y amplitud A. Después, reproduce el ruido
% por separado y, por último, el ruido combinado con el senoide
%
% f - frecuencia del senoide (0-11025)
% A - amplitud del senoide (0-1)
% Establece la tasa de muestreo en 22050 Hz
fs = 22050;
% Cree un filtro de paso de banda centrado en 1000 Hz. Como la
% tasa de muestreo es 22050, la frecuencia de Nyquist será 11025.
% 1000/11025 es aproximadamente 0,09, de ahí que los valores de la frecuencia estén
% entre 0,08 y 0,1, y por debajo. Si desea más información, escriba "help butter".
[b,a] = butter(4,[0,08 0,1]);
% Cree un vector de ruido blanco aleatorio (constante en todas las frecuencias)
wn = rand(1,22050);
% Filtre el ruido blanco con nuestro filtro
wf = filter(b,a,wn);
% Al filtrarlo, hemos reducido la potencia del ruido, por lo que podemos normalizar:
wf = wf/max(abs(wf))
% Cree el senoide con frecuencia f y amplitud A:
s = A*cos(2*pi*f/fs[0:fs-1]);
% Reproduzca los sonidos
sound(wf,22050)
pause(1) % Introduce una pausa de un segundo entre los sonidos
sound(wf+s,22050)
```