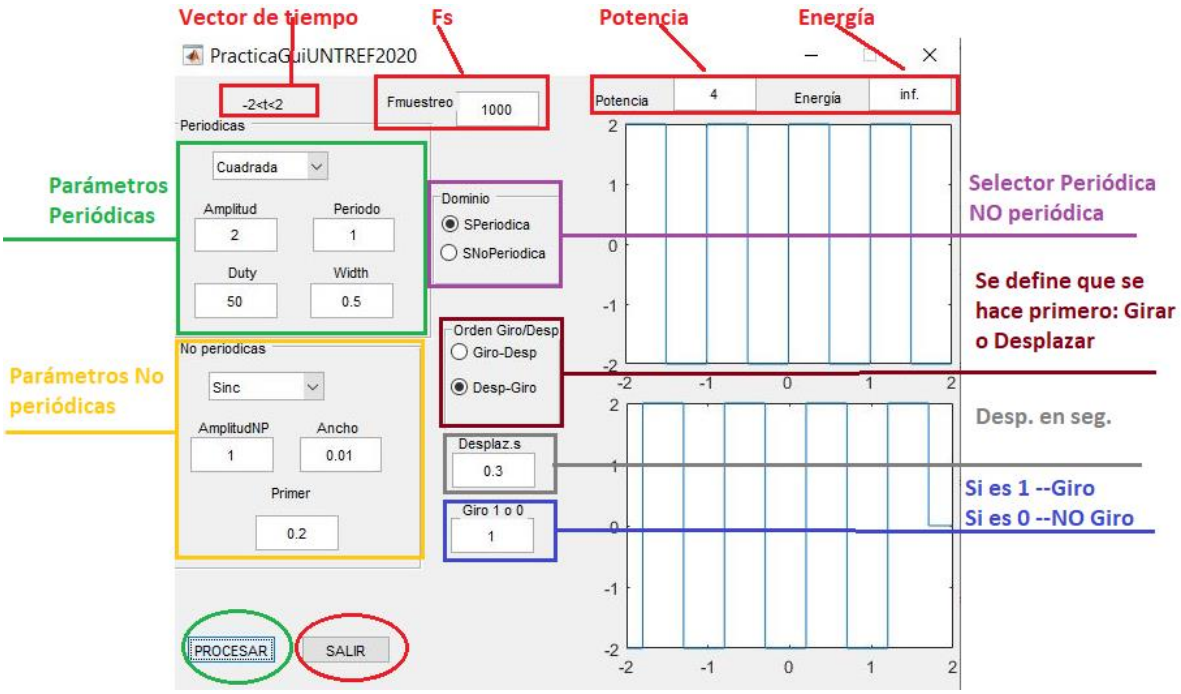
ASIGNACIÓN 1 PARTE 1

Este cuatrimestre haremos dos Trabajos Prácticos. El primero será con Matlab y el segundo con Python. El Trabajo Práctico 1 consistirá en diseñar un pequeño simulador en Matlab con interfaz gráfica. Lo iremos desarrollando por partes. La primera parte solo incluirá generación, manipulación de Señales y cálculo de Energía y Potencia.

Se desea crear una interfaz gráfica como la siguiente



Puede observar un sector destinado a generación de señales periódicas y otro a generación de señales no periódicas.

Se generará un vector de tiempo entre -2 y 2seg., con paso definido como ts=1/fs que es el inverso de fs, valor que introducirá el usuario. Se sugiere usen fs=1000.

Se le entrega una función llamada generador2020 que permite generar **3 señales periódicas y 3 no periódicas**. Esta será el punto de partida para generar todas las señales que se le exigen.

Las señales periódicas de su interfaz deben incluir:
Sinusoides (y=**Amplitud**.*cos(2*pi***frecuencia***t)), (**Periodo**=1/frecuencia)
Tren de pulso(y=**Amplitud**.*square(2*pi***frecuencia***t,**duty**)), y
Diente de sierra(y=**Amplitud**.*sawtooth(2*pi***frecuencia***t,**width**)).
Observe los elementos en **negritas** y compare con los parámetros que solicita la interfaz. Estas 3 señales están incluidas en la función generador2020.

Las señales no periódicas deben incluir
Sinc (y=**Amplitud**.*sinc(t/**primercorte**)),

Pulso rectangular (y=**Amplitud**.*rectpuls(t,**ancho**))
Pulso triangular (y=**Amplitud**.*tripuls(t,**ancho**)),
Estas tres las contiene la función generador. Observe los parámetros en **negritas** y compare con los parámetros que solicita la interfaz.
Además debe generar

- La señal escalón u(t) (y=heaviside(t)),
- La señal signo (y=sign(t)),
- Una señal exponencial decreciente unilateral (y=exp(-**ancho***t).*heaviside(t)).
- Una señal exponencial bilateral (y=exp(-**ancho***abs(t))).

Observe que usará la variable **ancho** (ya usada en los pulsos rectangular y triangular) para definir las constantes de las exponenciales.
Obviamente debe modificar la función generador2020 para incluir estas 4 señales no periódicas.

Una vez que funcione la generación de todas las señales, verificando que funcionan para diversos parámetros elegidos por el usuario, se pasará a modificarlas. El usuario puede solicitar dos cosas: girar la señal y desplazarla una cantidad de tiempo específica, positiva o negativa. También puede elegir **el orden** en que se harán esas modificaciones sobre la señal generada (Ver radio button Orden Giro/Desp). Allí se selecciona si primero Girar y luego Desplazar, o al revés. Se le entregan dos funciones que permiten hacer estas manipulaciones: manipular2020 (primero Giro y luego Desplazamiento) y manipular2020b (primero Desplazamiento y luego Giro).

Finalmente debe determinar la Energía y la Potencia de cada señal generada. Para las señales Periódicas E=infinito y P= mean(x.*x). Para las **no periódicas** E=(1/fs)*sum(x.*x) y P=0, excepto para el escalón y el signo que son señales de potencia E=infinito y P=mean(x.*x)

En la ventana superior se graficará la señal original (periódica o no periódica de acuerdo a la elección del usuario) y en la inferior la señal modificada.

Se le sugiere que diseñe un script, que logre todo lo solicitado con la ayuda de las funciones que se le ofrecen (generador2020, manipular2020 y manipular2020b). ***SOLO CUANDO EL SCRIPT FUNCIONE CORRECTAMENTE DEBE ABORDAR LA INTERFAZ GRÁFICA.***

Para aprender a realizar la interfaz, debe ver un vídeo colocado en el Drive, llamado **Como diseñar una GUI con GUIDE**. Allí se diseña una interfaz gráfica; no es exactamente la solicitada en esta oportunidad, pero se parece bastante. Lo único que no está explicado, es como colocar indicadores como el de Energía y Potencia. TÓMESE LA LIBERTAD DE CAMBIAR EL DISEÑO DE LA GUI SI ASÍ LO DESEA.
Para agregar los indicadores se colocan dos ventanas Edit Text. Coloque sus tags como **energía y potencia** en minúsculas. En el .m debe determinar E y P tal y como se explicó anteriormente. Cuando deba escribir que la energía vale infinito, use Strings o cadenas de caracteres (`E='inf.'`)
Luego, en el .m emplee el siguiente comando para colocar los valores en esos Edit Text. Observe que es vez de usar **get**, que permite tomar los valores de variables que el usuario escoge, usamos **set.**

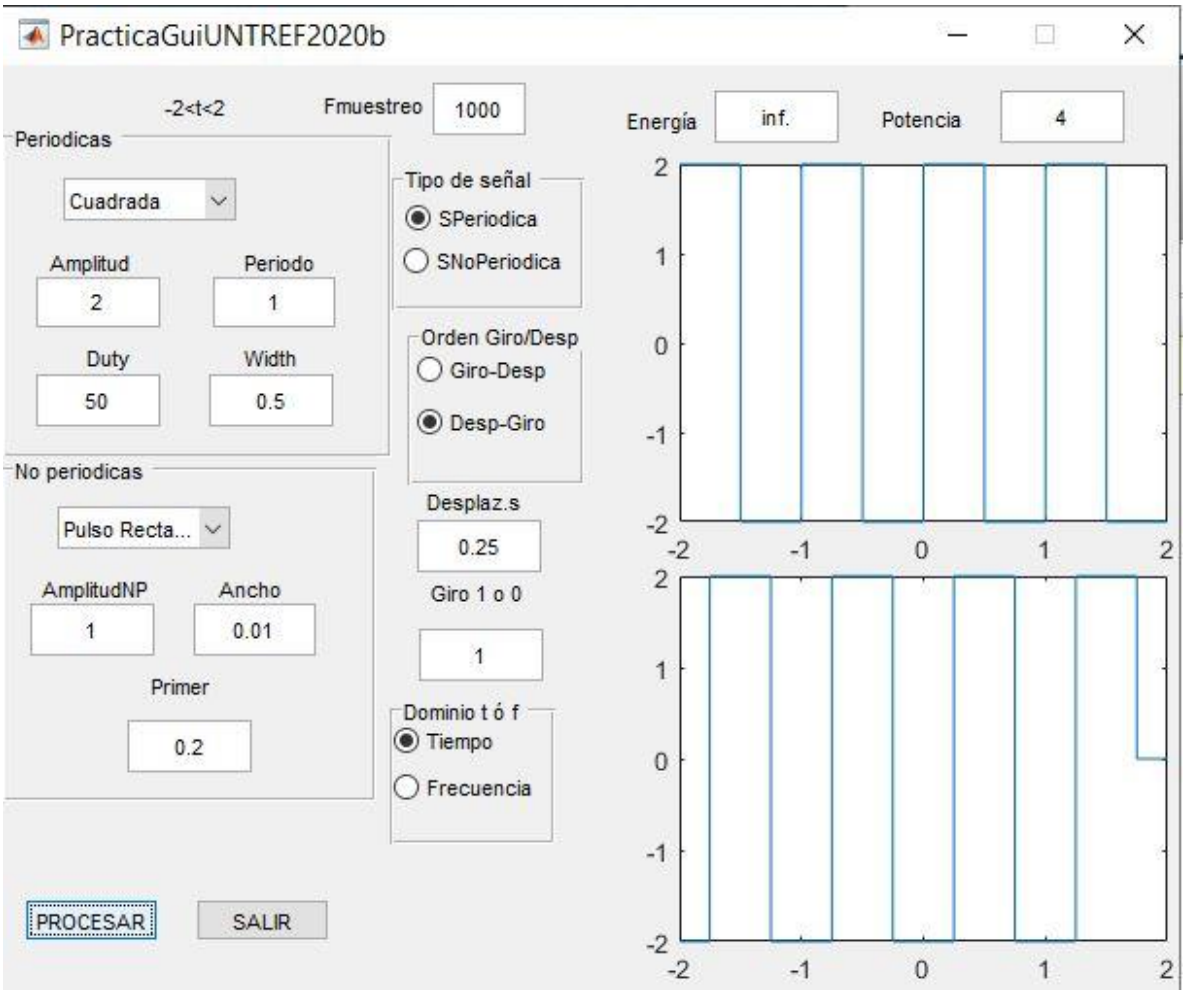
```
set(handles.energia,'String',E)

set(handles.potencia,'String',P)
```

El producto final, para esta primera etapa, debe ser un vídeo donde se muestren diferentes pruebas con la interfaz gráfica. Deben mostrar la generación de algunas señales periódicas y no periódicas, cambiar algunos de sus parámetros y verificar el efecto sobre la señal; luego mostrarán casos variados de manipulación de señales con las dos modalidades de Giro/Despl. Para las siguientes entregas se incluirá la visualización en frecuencia, la convolución y la aplicación de filtros.

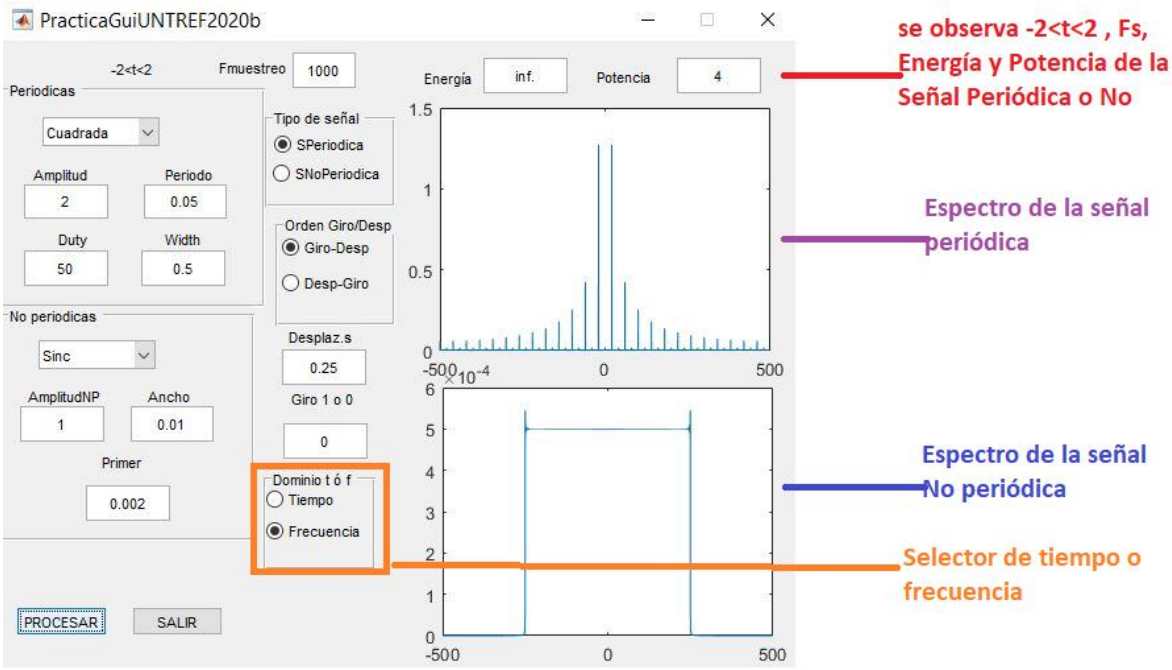
PARTE 2

En esta parte agregaremos la visualización de la señal en Frecuencia. Supongamos que estamos interesados en analizar un Sinc y una señal periódica de pulsos (Duty 50%). Mostraré primero esas señales en tiempo



Ahora mostraremos que ocurre cuando elegimos verlas en frecuencia. Como la señal modificada tiene la misma representación en frecuencia en MAGNITUD, **usaremos las ventanas o ejes para ver la representación en frecuencia de la periódica (arriba, axes 1) y de la no periódica (abajo, axes2)**.

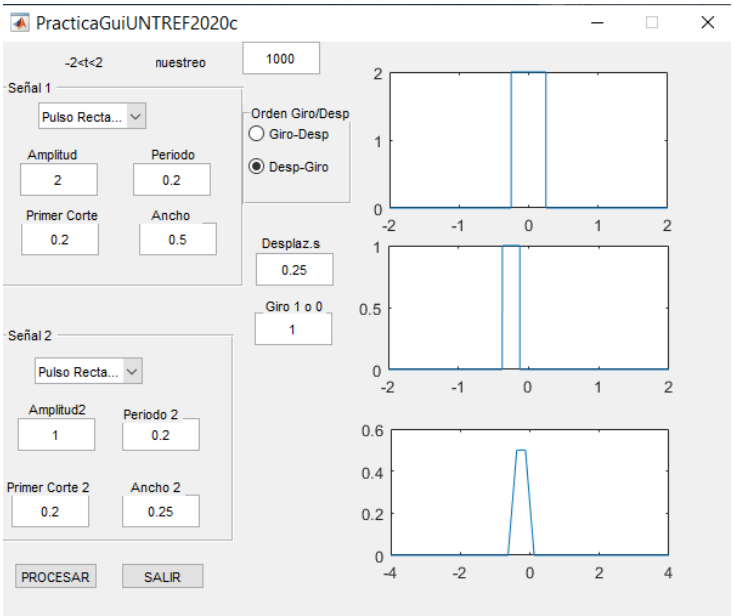
En la siguiente figura se muestra el resultado para la señal periódica tren de pulsos y para el Sinc. La función que usará para ver el espectro se llama fft6 ([MX,f]=fft6(x,fs,g)). A esta se le suministran como parámetros de entrada la señal **x**, la frecuencia de muestreo **fs** y un parámetro **g** que define si se trabajará en unidades de voltaje (g=1) o potencia (g=2). Usaremos g=1 por ahora.
El producto final, para esta segunda etapa, debe ser un nuevo vídeo donde se muestren diferentes pruebas con la interfaz gráfica. Deben mostrar la generación de algunas señales periódicas y no periódicas, cambiar algunos de sus parámetros y/o manipular la señal y verificar su efecto; luego mostrarán casos variados de señales visualizadas en tiempo y en frecuencia.



Como tenemos solo dos ejes gráficos, uno se usa para presentar el espectro de la señal periódica y el otro para presentar el espectro de la no periódica.

PARTE 3

La GUI para el caso de convolución luciría como sigue



El vector de tiempo sigue siendo el mismo entre -2 y 2. En cuanto a las señales a convolucionar llamadas x1 y x2, ambas podrán elegirse del siguiente conjunto: Coseno, Sinc, Pulso rectangular, Pulso triangular, Escalón y Signo. Se incluye la posibilidad de **manipular solo a x2** para generar una

mayor variedad de señales. Se les suministra la función convo que hace uso de la función conv (propia de MATLAB), pero que además devuelve al usuario el vector de tiempo del resultado de la convolución. **Debe agregar el selector para poder ver el resultado en frecuencia a elección del usuario**. Se tomará como un extra, poder incorporar la animación de la convolución que se les facilitó. Aparecería en el tercer eje gráfico. Advertencia: Debe tomar en cuenta algunas limitaciones prácticas que se presentarán, producto de restringir el vector de tiempo entre -2 y 2 seg. Por ejemplo, las señales escalón y signo que teóricamente tienen duración ilimitada, acá no la tendrán.

PARTE 4

Agregaremos filtraje.

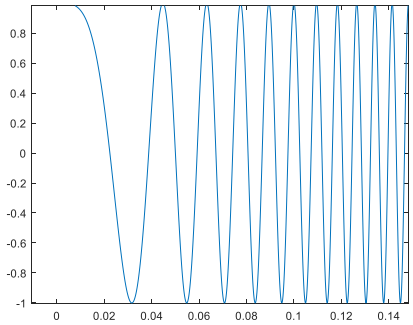
Se generarán los siguientes tipos de señales a ser filtradas: Señal de Voz femenina, Señal de Voz masculina, Suma de sinusoides y señal chirp. Explicaremos cada caso en detalle.

Señales de voz: Utilizaremos una frecuencia de muestreo de 8000 muestras/seg. Debe grabar sus propias señales (una voz masculina y una femenina) de 1 segundo de duración. Al final de este escrito se le ofrece un script que le facilitará esta tarea.

Suma de sinusoides: se generará la suma de 5 sinusoides, todas de amplitud 1, fase cero, y de frecuencias f1, 2f1, 3f1, 4f1 y 5f1, siendo f1=fs/200.

Chirp: Es una señal cuya frecuencia instantánea cambia con el tiempo. En este caso usaremos una variación lineal (ver figura). Este tipo de señal se usa, por ejemplo, para ubicar yacimientos de gas y petróleo con una técnica llamada VIBROSEIS. También sirve para visualizar la forma de la respuesta en frecuencia de un sistema, ya que como cada frecuencia está afectada por el valor de H a esa frecuencia, si H decrece, la amplitud de la señal de salida también. Finalmente, en su Trabajo Práctico 2, utilizarán una chirp de variación no lineal, para obtener la respuesta impulsiva de un recinto o sala.

```
x=chirp(t,0,1,Ffinal); REVISE EL HELP DE ESA FUNCIÓN
```



En cuanto al filtraje. Se le suministra una función que determina los parámetros de la función de transferencia de 16 tipos de filtros. Esta función se llama filtrosTP1

```
[y,b,a]=filtrosTP1(tipo,x,t,fs,fc,fl,fh,rp,rs,n)
```

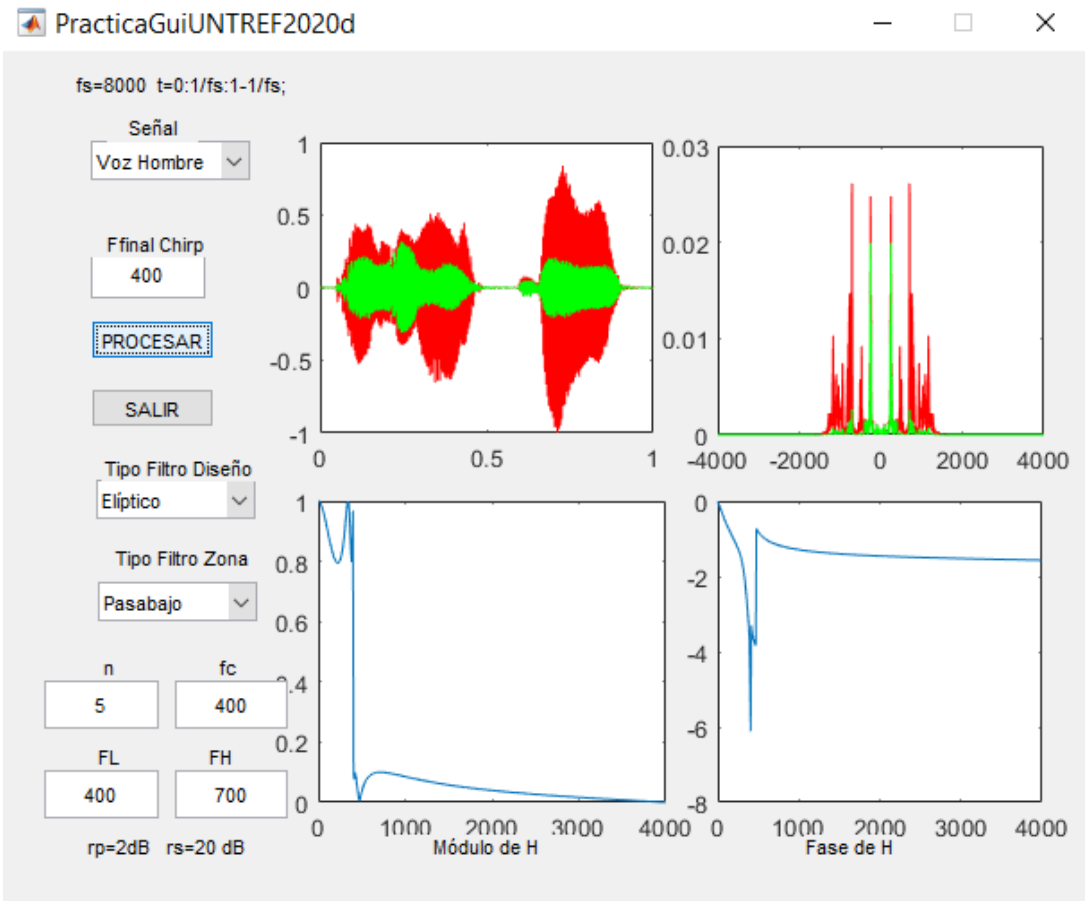
Sea PB un filtro Pasabajo o Pasabajas, PA un filtro Pasaalto o Pasaaltas, RB un filtro Rechaza Banda y BP un filtro Pasabanda (Band Pass). Tendremos además filtros Butterworth (B), Chebyshev 1 o directo (Ch1), Chebyshev 2 o inverso (Ch2) y Elíptico (E). La función filtrosTP1 genera 16 tipos diferentes, identificados por la variable tipo que va del 1 al 16 y que representa el siguiente caso (tomado del help de la función)

```
%      1= B-PB     2= B-PA     3= B-RB     4= B-BP
%      5= Ch1-PB   6= Ch1-PA   7= Ch1-RB   8= Ch1-BP
%      9= Ch2-PB   10= Ch2-PA  11= Ch2-RB  12= Ch2-BP
%      13= E-PB    14= E-PA    15= E-RB    16= E-BP
```

Para minimizar el número de parámetros que se le solicitarán al usuario, en cuanto al diseño del filtro, se fijarán el ripple de la banda pasante rp en 2dB y el ripple de la banda de rechazo en rs=20 dB. El usuario seleccionará uno de los 16 tipos, el orden y *la o las* frecuencias de corte.

Aquí se usará una función diferente de representación espectral, se llama fftplot [**f,MX,AX,X**]=fftplot(x,fs,N). Solicita x (señal en tiempo), fs y N o número de puntos. Allí coloque el tamaño del vector x (N=length(x)). Ofrece la salida en frecuencia **X**, su magnitud y fase **MX** y **AX** y el vector del eje de frecuencias **f**.

La interfaz lucirá como sigue.



En el primer gráfico se mostrarán las señales original y filtrada en dos colores, en el dominio del tiempo. En la segunda se mostrarán las señales original y filtrada en dos colores, en el dominio de la frecuencia. En la tercera gráfica se mostrará la magnitud de H y en la 4 la fase de H. Deben agregar un selector que permita que el usuario pueda ver, en la tercera gráfica, o la magnitud de H o la respuesta impulsiva h.

La función filtrosTP1 devuelve también los coeficientes del filtro a y b. Con ellos puede obtener la respuesta en frecuencia y la respuesta impulsiva

[H,f] = freqz(b,a,512,FS); REVISE el help de freqz
[h,t] = impz(b,a)

PARA GRABAR SEÑALES DE VOZ

```
fs=8000;                % se define la frecuencia de muestreo
tiempo=1;               % se define el tiempo de grabación en segundos
ts=1/fs;                % ts es el tiempo entre muestras
t=0:ts:tiempo-ts;       % se construye el vector de tiempo
%Creación del objeto de grabación (Fs, NBITS, NCHANS)
senal_salida=audiorecorder(fs,16,1);
% Grabación del sonido en señal_salida
recordblocking(senal_salida,tiempo);
% Se pasan los valores del objeto a una señal
senal_grabada=getaudiodata(senal_salida, 'single');
%Grabamos y guardamos la señal
y=senal_grabada;
y=y'; % se traspone para obtener un vector fila
% Para guardarla en disco formato .wav se usa audiowrite
audiowrite('Nombre.wav',y,fs);
% Para oírla
sound(y,fs)
%para graficar
plot(t,y)
```