7-8-2013

|  |
| --- |
| Universidad Politécnica de Durango. |



**INTEGRANTES**

* ERIK EMILIO FLORES DELFÍN.
* VIDAL EDGARDO MENDOZA AGUIRRE.

|  |  |
| --- | --- |
|  | Filtros Butterworth aplicados a una señal en Matlab. |

# Introducción.

Un filtro digital es un tipo de señal que opera sobre señales discretas y cuantizadas, implementados con tecnología digital.Un filtro digital es un sistema que, dependiendo de las variaciones de las señales de entrada en el tiempo y amplitud, se realiza un procesamiento matemático sobre dicha señal; generalmente se hace uso de la Transformada de Fourier. Se le considera filtros digitales por el proceso interno que manejan. Comúnmente se usan para atenuar frecuencias o amplificarlas.

En el tema de los filtros digitales es muy importante el manejo de corte de frecuencia de Nyquist-Shannon. De acuerdo con la parte del espectro que dejan pasar y que atenúan hay:

* Filtros pasa alto.
* Filtros pasa bajo.
* Filtros pasa banda.
* Banda eliminada.
* Multibanda.
* Pasa todo.
* Resonador.
* Oscilador.
* Filtro peine (Comb filter).
* Filtro ranura o filtro rechaza banda (Notch filter).

De acuerdo con su orden:

* primer orden
* segundo orden

De acuerdo con el tipo de respuesta ante entrada unitaria:

* FIR (Finite Impulse Response)
* IIR (Infinite Impulse Response)
* TIIR (Truncated Infinite Impulse Response)

De acuerdo con la estructura con que se implementa:

* Laticce
* Varios en cascada
* Varios en paralelo

En nuestro caso manejamos los filtros butterworth mediante el programa Matlab. Este tipo de filtros, son de los más básicos de electrónica, pues su respuesta es muy plana hasta la frecuencia de corte, es decir, su salida se mantiene constante casi hasta la frecuencia de corte. Fue descrito por primera vez por el ingeniero británico S. Butterworth.

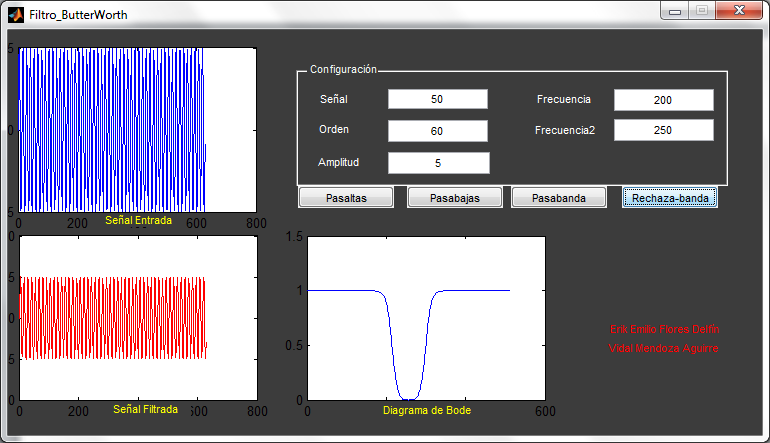
Cabe mencionar que los filtros Butterworth son los únicos que mantienen su forma para órdenes mayores.

# Desarrollo.

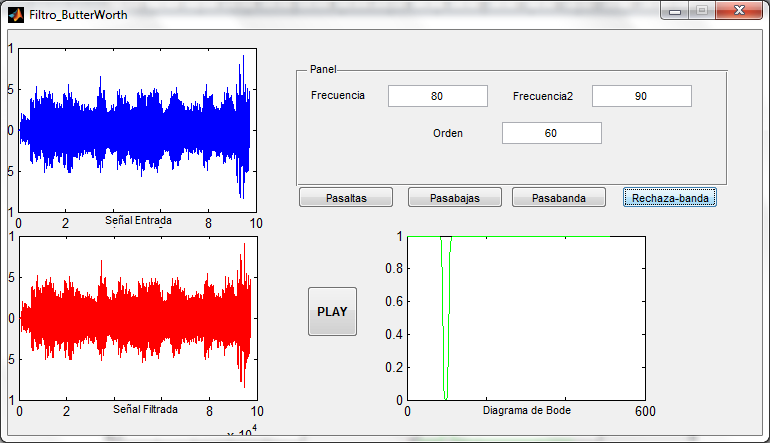
En nuestro caso, nos tocó programar el algoritmo para los siguientes 4 filtros Butterworth en Matlab:

## Filtro Rechaza-Banda:

También conocido como elimina banda, filtro notch, filtro trampa, filtro de rechazo de banda, es un filtro electrónico que no permite el paso de señales cuyas frecuencias se encuentran comprendidas entre las frecuencias de corte superior e inferior.



**Figura 1. Imagen generada por el programa Matlab sobre el filtro Butterworth rechazabanda.**



**Figura 3. Señal de entrada y filtrada de un archivo de audio WAV, mediante el filtro rechazabanda butterworth.**

### Código:

*%RECHAZABANDA*

*clc*

*fci=str2double(get(handles.fc1, 'String'));*

*fcs=str2double(get(handles.fc2, 'String'));*

*a=str2double(get(handles.Amp,'String'));*

*ent=str2double(get(handles.Signal,'String'));*

*orden=str2double(get(handles.Orden, 'String'));*

*rs=orden;*

*x=-pi:0.01:pi;*

*if(fcs<50)*

*fm=100;*

*else*

*fm=900;*

*end*

*y=a\*sin(x\*ent);*

*plot(handles.axes1,y)*

*fe=fm/2;*

*%banda de paso*

*wp=[fci fcs]/fe*

*faa=(fci/10)*

*fbb=(fcs/2)+fcs*

*%banda eliminada*

*ws=[faa fbb]/fe*

*%atenuacion 3db*

*rp=3;*

*%atenuacion banda eliminada*

*% rs=40;*

*%ahora obtenemos el orden n y frec de corte*

*[n,wn]=buttord(wp, ws, rp, rs);*

*[b,a]=butter(n,wn,'stop');*

*[h,f]=freqz(b,a,[],fm);*

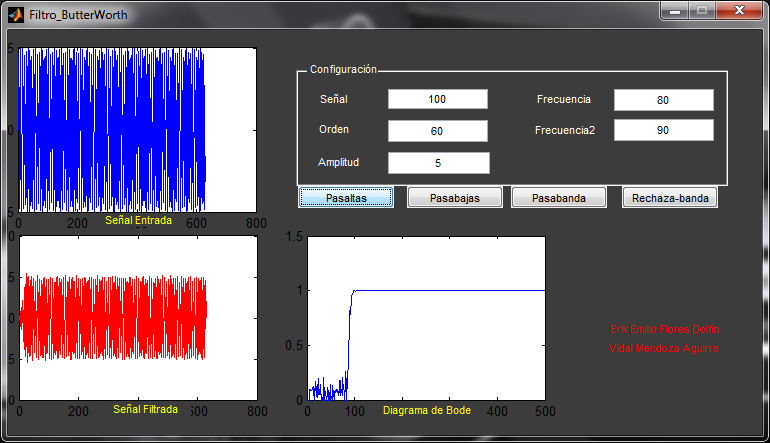
*plot(handles.axes2, abs(h))*

*y=filter(b,a,y);*

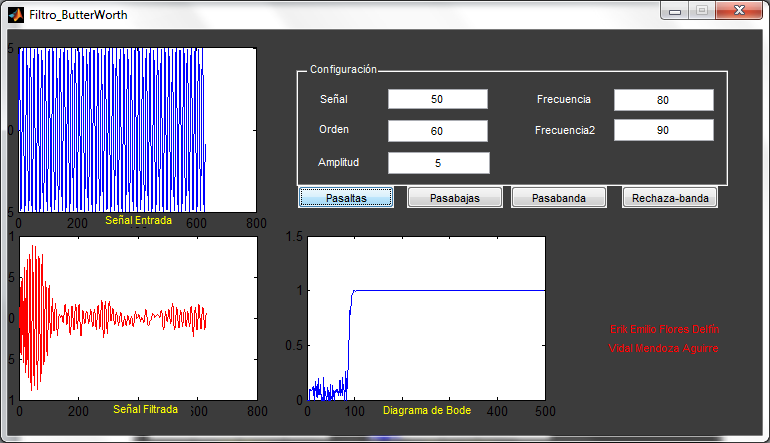
*plot(handles.axes3,y)*

## Filtro Pasa-Altas

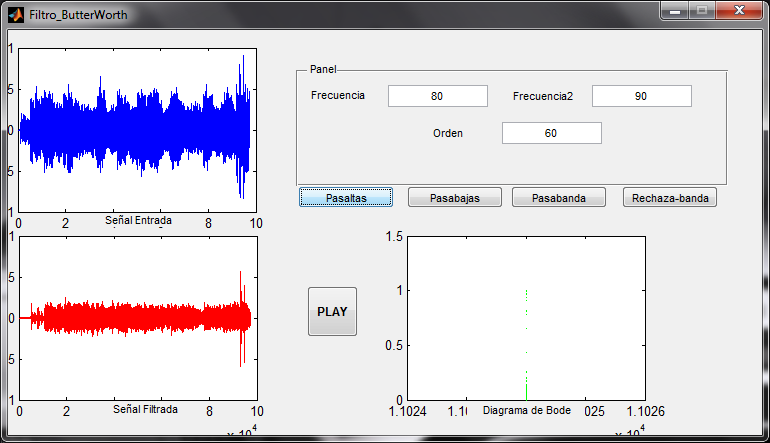
Es un tipo de filtro electrónico en cuya respuesta se atenúan las componentes de baja frecuencia pero no las de alta frecuencia, éstas incluso pueden amplificarse en los filtros activos.



**Figura 4. Imagen de la señal original introducida para el filtro pasa altas.**



**Figura 5. Filtro pasa altas en acción.**



**Figura 6. Señal de entrada y filtrada de un archivo de audio WAV, mediante el filtro pasabanda butterworth.**

### Código:

*clc*

*x=-pi:0.01:pi;*

*fci=str2double(get(handles.fc1,'string'));*

*a=str2double(get(handles.Amp,'String'));*

*ent=str2double(get(handles.Signal,'String'));orden=str2double(get(handles.Orden,'String'));*

*if(fci<50)*

*opc=100;*

*else*

*opc=1000*

*end*

*y=a\*sin(x\*ent)*

*plot(handles.axes1,y)*

*fe=opc/2*

*wp=(fci+(opc/100))/fe*

*ws=(fci-(opc/100))/fe*

*rp=3;*

*%Orden y frecuencia normalizada necesarios para usar filtro*

*[n,w]=buttord(wp,ws, rp, orden, 's')*

*%funcion transferencia en tiempo*

*[B,A]=butter(n,w,'high');*

*[H,F]=freqz(B,A,[],opc);*

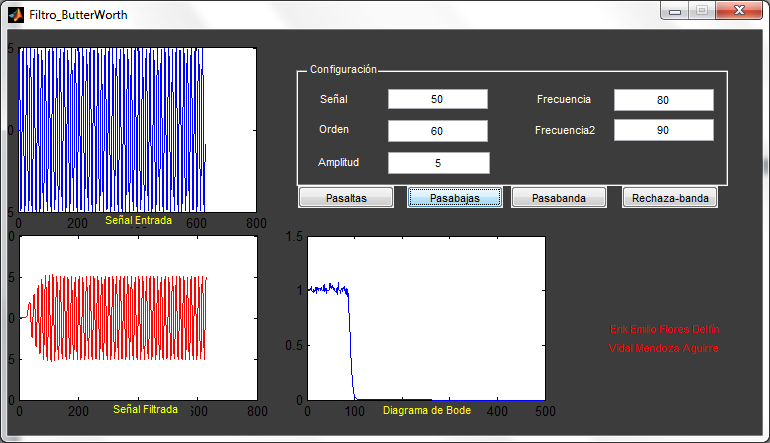
*plot(handles.axes2,F,abs(H))*

*y=filter(B,A,y);*

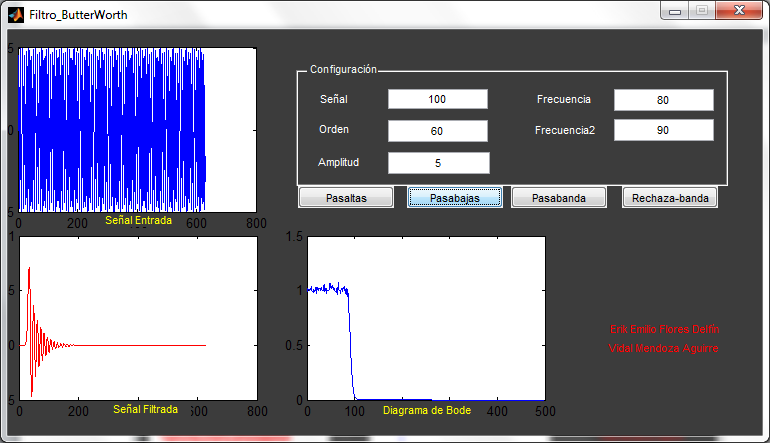
*plot(handles.axes3,y)*

## Filtro pasa bajas:

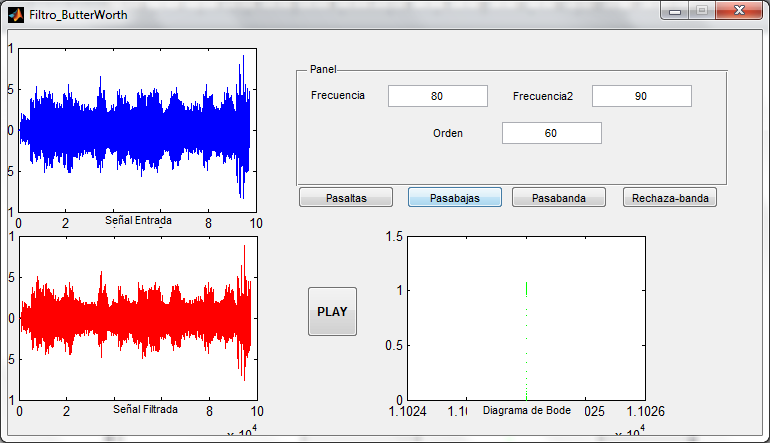
Corresponde a un filtro caracterizado por permitir el paso de las frecuencias más bajas y atenuar las frecuencias más altas.



**Figura 7. Espectro de una señal original.**



**Figura 8. Señal filtrada mediante el filtro Butterworth pasa bajas.**



**Figura 9. Señal de entrada y filtrada de un archivo de audio WAV, mediante el filtro pasabajas Butterworth.**

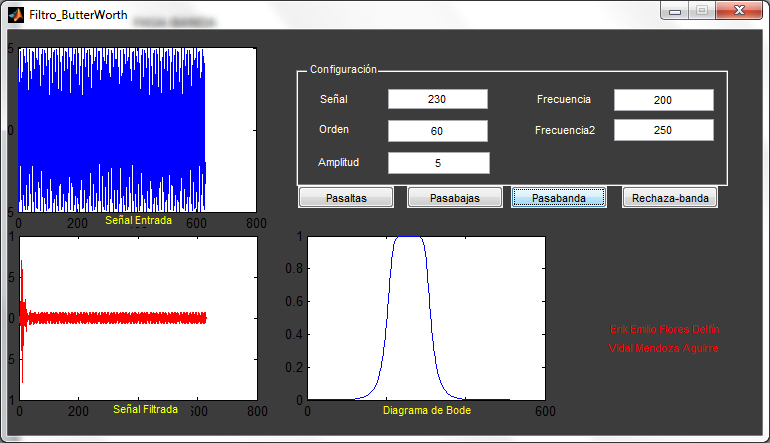
### Código:

*%PASABAJAS*

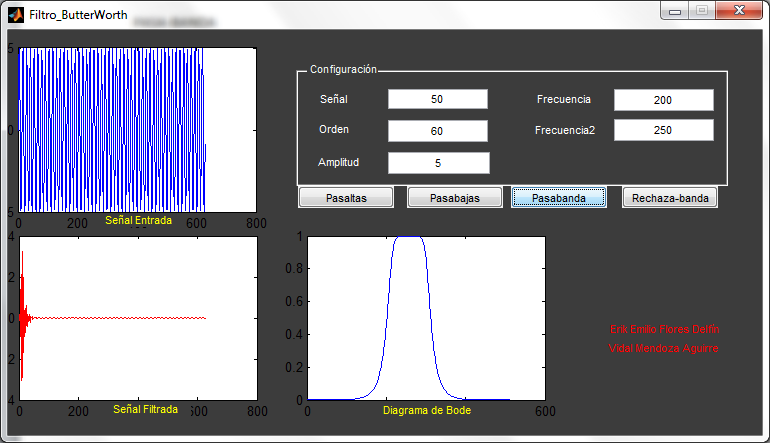
*clcx=-pi:0.01:pi;fci=str2double(get(handles.fc1,'string'));a=str2double(get(handles.Amp,'String'));ent=str2double(get(handles.Signal,'String'));orden=str2double(get(handles.Orden,'String'));if(fci<50)opc=100;elseopc=1000;endy=a\*sin(x\*ent);plot(handles.axes1,y)fe=opc/2;wp=(fci+(opc/100))/fe;ws=(fci-(opc/100))/fe;rp=3;%Orden y frecuencia normalizada necesarios para usar filtro[n,w]=buttord(wp, ws, rp, orden);%Funcion transferencia en tiempo[B,A]=butter(n,w,'low');[H,F]=freqz(B,A,[],opc);plot(handles.axes2,F,abs(H))y=filter(B,A,y);plot(handles.axes3,y)*

## Filtro pasa banda:

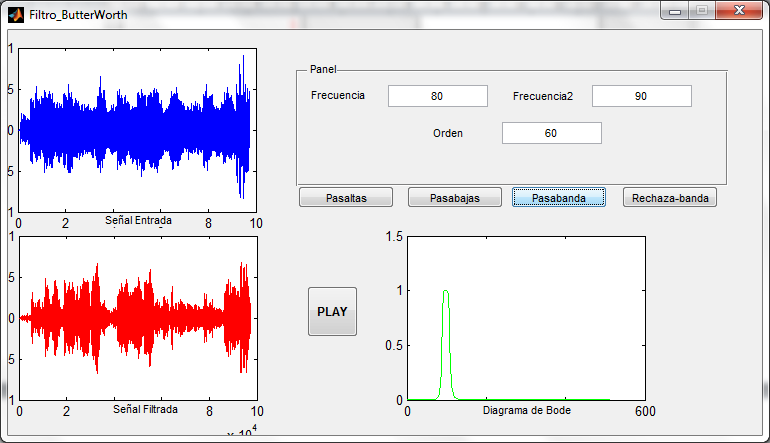
Es un tipo de filtro electrónico que deja pasar un determinado rango de frecuencias de una señal y atenúa el paso del resto.



**Figura 10. Señal original mostrada por el programa de filtros Butterworth.**



**Figura 11. Demostración de la señal filtrada con un filtro pasa banda.**



**Figura 12. Señal de entrada y filtrada de un archivo de audio WAV, mediante el filtro pasabanda butterworth.**

### Código:

*%%PASABANDA*

*clcorden=str2double(get(handles.Orden, 'String'));rs=orden;*

*fa=str2double(get(handles.fc1, 'String'));fb=str2double(get(handles.fc2, 'String'));am=str2double(get(handles.Amp,'String'));ent=str2double(get(handles.Signal,'String'));t=-pi:0.01:pi;if(fb<50)*

*fm=100;elsefm=900;*

*end*

*fe=fm/2;%banda de pasowp=[fa fb]/fefaa=(fa/10)fbb=(fb/2)+fb%banda eliminadaws=[faa fbb]/fe%atenuación 3dbrp=3;%atenuación banda eliminada% rs=40;%ahora obtenemos el orden n y frec de corte*

*[n,wn]=buttord(wp, ws, rp, rs); [b,a]=butter(n,wn); [h,f]=freqz(b,a,[],fm);plot(handles.axes2, abs(h))y=am\*sin(t\*ent);plot(handles.axes1,y)y=filter(b,a,y);plot(handles.axes3,y)*

# Filtros con aplicación de un sonido.

En este caso, el código de cada filtro solo se modificó en unas cuantas líneas que fueran eliminadas, tales como la entrada de señal y la entrada de amplitud, pues a partir de un archivo de sonido con extensión WAV, se introdujeron estos parámetros implícitamente.

Las 2 líneas de código que se ingresaron para el archivo de sonido fueron:

*[y,fe]=wavread('tigre.wav')*

*wavplay(y,fe);*

Esto con el fin de reproducir el sonido original. En el caso de mostrar el sonido derivado por el filtro se agregaron estos comandos:

*y=filter[b,a,t]*

*wavplay(y,fe);*

Reproduciendo así el archivo filtrado.

# Conclusiones.

Los filtros Butterworth son sencillos a partir del programa Matlab y sus funciones, el único inconveniente que se nos presento fue cuando tratamos de realizar los filtros y las frecuencias no se encontraban normalizadas debido a un error en el corte de frecuencia de Nyquist-Shannon, pues en filtros digitales es de uso obligado.

Otro inconveniente fue que uno de los filtros no funcionaba del todo bien a partir de frecuencias de 1000Hz.

# Referencias.

* <http://es.wikipedia.org/wiki/Filtro_digital>
* http://es.wikipedia.org/wiki/Filtro\_de\_Butterworth