ESCUELA POLITECNICA NACIONAL

FACULTAD DE INGENIERIA ELECTRICA

MANUAL Y LISTADO DE RUTINAS DEL MODULO DE PROCESAMIENTO DIGITAL DE SEÑALES (PDS)

"PROCESAMIENTO DIGITAL DE SEÑALES UTILIZANDO EL PAQUETE MATLAB"

TESIS PREVIA A LA OBTENCION DEL TITULO DE

INGENIERO EN ELECTRONICA Y CONTROL

RODOLFO DONOSO MENESES

MARZO, 1995

CONTENIDO

		Pág.
A.I.	MANUAL DEL USUARIO DEL MODULO DE PROCESAMIENTO DIGITAL DE SEÑALES (PDS).....	1
A.1.	Procesamiento digital de señales.....	3
A.1.1.	Escoger tipo de planta a simular.....	4
A.1.2.	Análisis espectral.....	4
A.1.3.	Diseño de filtros digitales.....	4
A.1.3.1.	Filtros recursivos.....	5
A.1.3.1.1.	Utilizando la transformada bilineal.....	5
A.1.3.1.2.	Método de Yule-Walker.....	5
A.1.3.1.3.	Método de Prony.....	6
A.1.3.1.4.	Método de error mínimo cuadrado en el dominio de frecuencia.....	6
A.1.3.2.	Filtros no recursivos.....	6
A.1.3.2.1.	Diseño de fase lineal de longitud definida.....	7
A.1.3.2.2.	Diseño de fase lineal general.....	7
A.1.3.2.3.	Diseño por el algoritmo de Parks-McClellan.....	8
A.1.4.	Filtrado digital de señales contaminadas.....	8
A.1.4.1.	Filtrado mediante diseño de filtros recursivos.....	9
A.1.4.1.1.	Filtrado de fase cero.....	9
A.1.4.1.2.	Filtrado en forma directa transpuesta.....	9
A.1.4.2.	Filtrado mediante diseño de filtros no recursivos.....	9
A.1.5.	Ajuste polinomial: Interpolación y extrapolación.....	10
A.II.	LISTADO DE RUTINAS DEL MODULO DE PROCESAMIENTO DIGITAL DE SEÑALES (PDS).....	12
A.2.	Implementación de software para procesamiento digital de señales.....	13
A.2.1.	Simulación de plantas.....	13
A.2.2.	Análisis espectral.....	25
A.2.3.	Diseño de filtros digitales.....	27
A.2.3.1.	Transformada bilineal.....	29
A.2.3.2.	Método de Yule-Walker.....	38
A.2.3.3.	Método de Prony.....	40
A.2.3.4.	Método de minimización del error medio cuadrático.....	42
A.2.3.5.	Método de fase lineal.....	43
A.2.3.6.	Método de fase lineal en el dominio de la frecuencia.....	58
A.2.3.7.	Método de diseño de Parks- McClellan.....	75
A.2.4.	Métodos de filtrado de las señales.....	77

A.2.5. Ajuste polinomial de señales digitales..... 84
A.2.6. Señal binaria pseudo aleatoria..... 86
A.2.7. Subrutinas de ayuda llamadas desde las rutinas..... 87
A.2.8. Rutinas adicionales..... 92

APENDICE I

MANUAL DEL USUARIO DEL MODULO DE PROCESAMIENTO DIGITAL DE SEÑALES (PDS)

El presente manual, tiene por objetivo el ayudar en la instalación y el manejo del módulo PDS (Procesamiento Digital de Señales), software que trabaja sobre la base algorítmica del paquete MATLAB (versión 3.5a) utilizando la librería del "Signal Toolbox" del mismo.

Para utilizar éste módulo, necesariamente se debe tener cargado el MATLAB en disco c: en directorio \MATLAB\. Se realiza las siguientes recomendaciones para una óptima instalación del módulo:

1) Chequear el archivo MATLAB.BAT con un editor de texto cualquiera y cambiarlo al siguiente modelo, o crearlo si no existe, con las siguientes instrucciones:

```
@ECHO OFF
SET MATLABPATH = \MATLAB;\MATLAB\SIGNAL;\MATLAB\CONTROL;
                 \MATLAB\PDS;\MATLAB\MSCDEMOS;
:LOOP
PCMATLAB VGA
IF NOT EXIST EDIT.MAT GOTO END
EDIT
GOTO LOOP
:END
```

En este caso se está utilizando el editor de texto EDIT del DOS 6.1. En caso de utilizar otro editor (edlin, norton editor, etc., se deberá cambiar la instrucción EDIT por la ejecutable correspondiente del programa editor que se vaya a utilizar, como también el driver de PCMATLAB, el que en este caso es para tarjetas VGA (para tarjetas monocromo Hércules:

HGC, etc.).[4]

2) Revisar en un editor de texto cualquiera, el archivo GRAPH.BAT, o crearlo si es necesario, el cual se utiliza para realizar impresiones gráficas de los resultados del módulo. El archivo utiliza el comando GPP.EXE (Graphics Post Processor) que funciona a nivel DOS y contiene ciertos argumentos que definen la calidad y el pórtico de salida para impresión (DEPSD = impresora EPSON con impresión de baja calidad; DEPSF = impresora EPSON con impresión de alta calidad; FPRN = primer puerto paralelo; FLPT2 = segundo puerto paralelo, etc.).[4]

El archivo contiene actualmente la siguiente configuración:

```
GPP %1.MET /DEPSD /FPRN
```

3) Crear si no existe, o chequear que el archivo PDS.BAT esté grabado en el directorio raíz del disco c: y en el subdirectorio \MATLAB\PDS\; este archivo facilita el acceso directo al MATLAB. Su configuración es la siguiente:

```
@ECHO OFF
C:
CD \MATLAB\PDS
MATLAB
```

Luego de realizada esta introducción a la instalación del módulo PDS, es conveniente saber que cuando se quiere imprimir los gráficos de resultados del módulo, el programa del MATLAB es abandonado, ya que el procesador de gráficos del mismo trabaja a nivel de DOS. Sin embargo, al dejar el programa, todas las variables del espacio de trabajo, son grabadas automáticamente en el archivo MATLAB.MAT lo que permite cargarlas al volver a ingresar al programa mediante el comando LOAD, tecleado desde la línea de comandos del MATLAB. Lo mismo sucede al abandonar el programa voluntariamente desde el menú principal del PDS, pudiendo cargarse las variables de la última sesión con el módulo, en

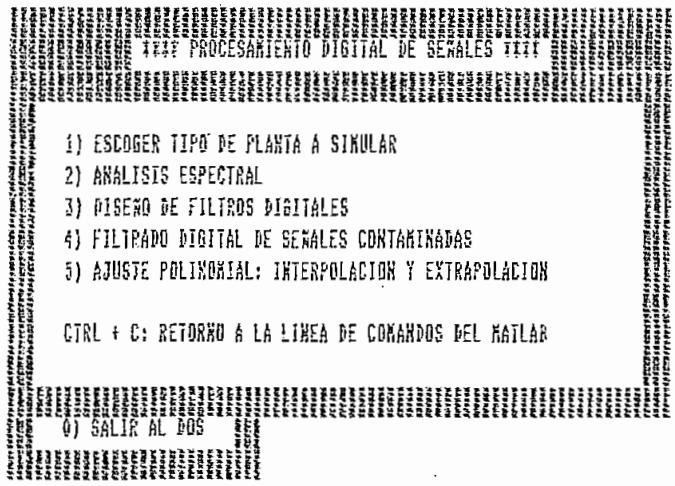
la próxima ocasión.

A continuación se describe el campo de acción del módulo PDS en base a los distintos menús y submenús que posee, con breve descripción de las variables más importantes de cada rutina, con el fin de tener conocimiento de ellas en caso de querer utilizarlas directamente (como variables) en los análisis que se realicen (en lugar de difinir toda la función).

El menú principal proporciona la opción de salir temporalmente a la línea de comandos del MATLAB, con el fin de definir cualquier función o correr alguna subrutina para utilizar sus variables dentro del módulo de PDS. Para volver al programa o entrar a él por primera vez, se teclea PDS en la línea de comandos.

A.1. PROCESAMIENTO DIGITAL DE SEÑALES

El menú principal consta de las siguientes opciones, las que se encuentran subdivididas en menús secundarios, los cuales se explican dentro de cada opción:



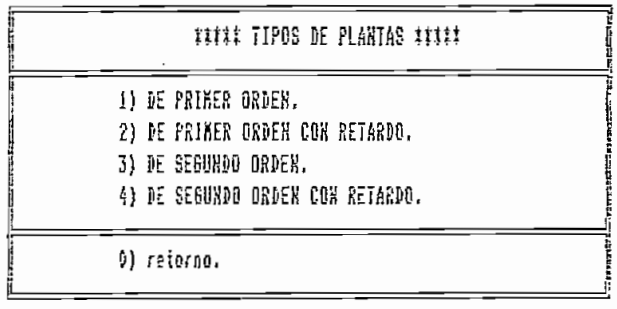
ESCUELA POLITECNICA NACIONAL

FACULTAD DE INGENIERIA ELECTRICA

A.1.1. ESCOGER TIPO DE PLANTA A SIMULAR.

En este menú, todas las opciones poseen las siguientes variables de interés:

- g: señal de ruido que se introduce.
- y: respuesta de la planta.
- y0: señal definida por el usuario.
- y2: respuesta de la planta, contaminada con ruido.

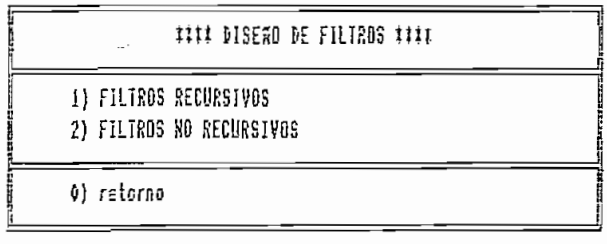


A.1.2. ANALISIS ESPECTRAL.

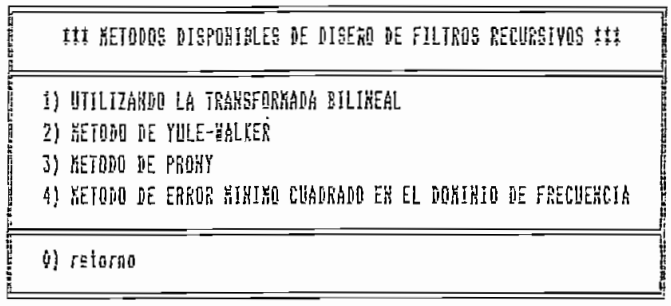
Esta rutina no posee submenús, y sus variables de interés son:

- fs: frecuencia de muestreo.
- y2: respuesta de la planta contaminada con ruido.
- y3: señal u(to) definida por el usuario.
- g: señal de ruido presente en la respuesta de la planta.

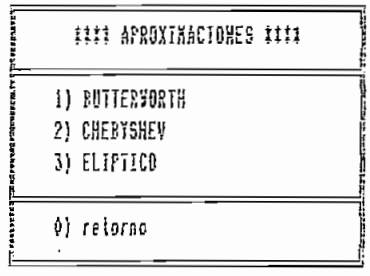
A.1.3. DISEÑO DE FILTROS DIGITALES.



A.1.3.1. FILTROS RECURSIVOS.



A.1.3.1.1. UTILIZANDO LA TRANSFORMADA BILINEAL.



Estos filtros digitales poseen las siguientes variables de interés:

- fs: frecuencia de muestreo.
- Fp: frecuencia de paso.
- Fs: frecuencia de bloqueo.
- Rp: rizado en la región de paso.
- Rs: rizado en la región de bloqueo.
- N: orden del filtro.
- Wn: frecuencia natural del filtro (de corte: (2*Fp)/fs).
- B: coeficientes del filtro en el numerador.
- A: coeficientes del filtro en el denominador.
- Hm: magnitud de respuesta de frecuencia del filtro.

A.1.3.1.2. METODO DE YULE-WALKER.

Esta rutina no posee submenús, y las variables de interés son las siguientes:

- F: vector de frecuencias.
- M: vector de magnitudes.
- N: orden del filtro.
- fs: frecuencia de muestreo.
- B: coeficientes del numerador del filtro.
- A: coeficientes del denominador del filtro.

A.1.3.1.3. METODO DE PRONY.

Al igual que el anterior no posee submenús, y sus variables más importantes son:

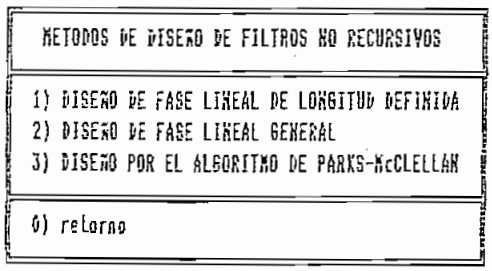
- NB: orden del numerador.
- NA: orden del denominador.
- H1: respuesta impulso del filtro.
- B: coeficientes del numerador del filtro.
- A: coeficientes del denominador del filtro.
- Hm: magnitud de respuesta de frecuencia del filtro.

A.1.3.1.4. METODO DE ERROR MINIMO CUADRADO EN EL DOMINIO DE FRECUENCIA.

A semejanza de los dos anteriores, esta rutina no posee submenús. Las variables de interés son:

- NB: orden del numerador del filtro.
- NA: orden del denominador del filtro.
- B: coeficientes del numerador.
- A: coeficientes del denominador.
- H: respuesta de frecuencia.

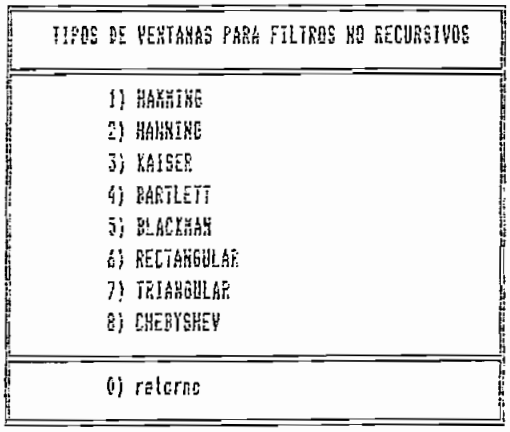
A.1.3.2. FILTROS NO RECURSIVOS.



A.1.3.2.1. DISEÑO DE FASE LINEAL DE LONGITUD DEFINIDA.

Todos los filtros diseñados por este método, poseen las siguientes variables de interés:

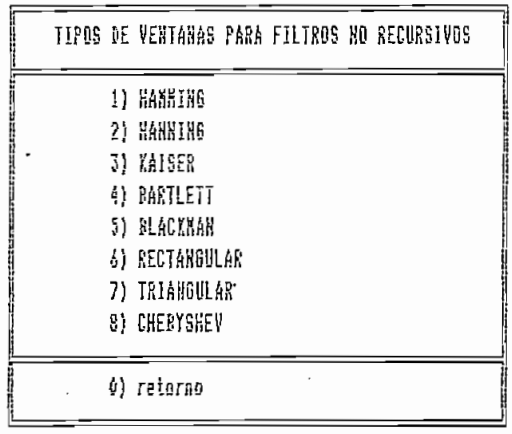
- fs: frecuencia de muestreo.
- Fc: frecuencia de corte.
- N1: orden del filtro.
- B: coeficientes del filtro.
- Wn1: frecuencia natural (2*Fc)/fs.
- H: respuesta de frecuencia.
- beta: para la ventana de Kaiser (la cual es beta valuada).



A.1.3.2.2. DISEÑO DE FASE LINEAL GENERAL.

Este método posee las mismas ventanas para el diseño que el método anterior, su diferencia radica en que el presente método trabaja en el dominio de la frecuencia, por lo tanto sus variables principales son:

- fs: frecuencia de muestreo.
- N: orden del filtro.
- F: vector de frecuencias.
- M: vector de magnitudes.
- B: coeficientes del filtro.
- H: respuesta de frecuencia.
- Hm: magnitud de respuesta de frecuencia.



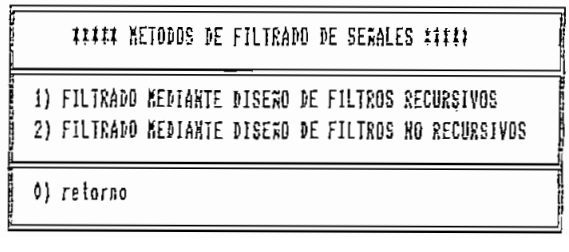
Con el método anterior se pueden diseñar filtros multibanda, ya que se puede definir el vector de frecuencias de corte y el vector de magnitudes de la magnitud de respuesta de frecuencia del filtro, de manera que puedan haber combinaciones entre filtros pasa bajos, pasa banda, pasa altos y elimina banda.

A.1.3.2.3. DISEÑO POR EL ALGORITMO DE PARKS-McCLELLAN.

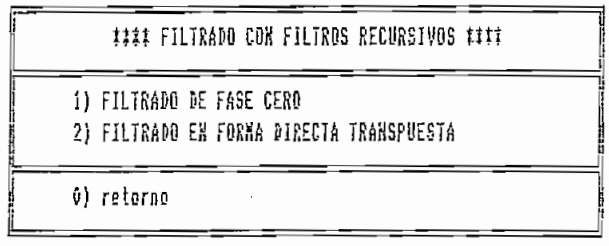
Este método no posee submenú, y con él se pueden realizar diseños multibanda; sus variables más importantes son las siguientes:

- fs: frecuencia de muestreo.
- N: orden del filtro.
- F: vector de frecuencias de la respuesta de frecuencia.
- M: vector de magnitudes de la respuesta de frecuencia.
- B: coeficientes del filtro.
- H: respuesta de frecuencia (compleja).
- Hm: magnitud de la respuesta de frecuencia.

A.1.4. FILTRADO DIGITAL DE SEÑALES CONTAMINADAS.



A.1.4.1. FILTRADO MEDIANTE DISEÑO DE FILTROS RECURSIVOS.



A.1.4.1.1. FILTRADO DE FASE CERO.

Este método de filtrado dobla el orden del filtro, y posee fase cero debido a que filtra la señal dos veces en sentido opuesto, por lo que el desfasamiento se anula. Las variables de interés son las siguientes:

- fs: frecuencia de muestreo.
- y2: señal de salida contaminada de la planta.
- y3: señal definida por el usuario.
- F0: señal filtrada.

A.1.4.1.2. FILTRADO EN FORMA DIRECTA TRANSPUESTA.

Este método es de característica menos bondadosa que el primero, ya que no dobla el orden del filtro, y no posee fase cero. Sus variables más importantes son:

- fs: frecuencia de muestreo.
- y2: señal de salida contaminada de la planta.
- y3: señal definida por el usuario.
- F1: señal filtrada.

A.1.4.2. FILTRADO MEDIANTE DISEÑO DE FILTROS NO RECURSIVOS.

Este método utiliza la TRF (Transformada Rápida de Fourier), para filtrar la señal. Sus variables principales son:

- fs: frecuencia de muestreo.
- y2: señal de salida contaminada de la planta.

- y3: señal definida por el usuario.
- F2: señal filtrada.

A.1.5. AJUSTE POLINOMIAL: INTERPOLACION Y EXTRAPOLACION.

Esta rutina no posee submenús. A más de realizar modelación polinomial de señales digitales, contiene aplicaciones en pérdida de datos, como la interpolación y la extrapolación. Sus variables más importantes son:

- xk: valores de k (discreto).
- yk: valores de x(k).
- Nk: orden del modelo polinómico.
- xk´: valores de k´ (para interpolación o extrapolación).

Es necesario mencionar, que dependiendo del computador que se esté utilizando para correr el MATLAB y el módulo PDS, puede ocurrir que luego de algún tiempo de estar trabajando con el paquete, se observe un error en pantalla, "out of memory", que es debido a las nuevas variables que el MATLAB va creando conforme realiza nuevos cálculos y que empiezan a consumir memoria del computador por no ser variables temporales. La solución para este problema es utilizar uno de los siguientes métodos:

- Teclear CLEAR en la línea de comandos del MATLAB para borrar el espacio de trabajo del mismo (con todas sus variables).

- Utilizar el comando PACK, que "empaca" todas las variables del espacio de trabajo, para proporcionar más memoria al mismo (no siempre funciona ya que depende del tipo de variables). Se recomienda utilizar un mismo valor de frecuencia de muestreo en todas las rutinas que se utilicen relacionadas entre sí, por ejemplo, si se simula la respuesta de una planta, y la simulación es hecha a una frecuencia de 100 Hz, al realizar el análisis espectral, el diseño del filtro y el filtrado de la señal contaminada, es necesario seguir utilizando la frecuencia de muestreo de 100

Hz. En caso de definir una señal cualquiera, debe utilizarse la fs adecuada para dicha señal, teniendo en cuenta el teorema del muestreo que manifiesta que fs > 2*f máxima del sistema.

APENDICE II

LISTADO DE RUTINAS DEL MODULO DE PROCESAMIENTO DIGITAL DE SEÑALES (PDS)

Las rutinas presentadas a continuación están ordenadas según los diagramas de flujo mostrados en el capítulo IV del texto de tesis.

A.2. IMPLEMENTACION DE SOFTWARE PARA PROCESAMIENTO DIGITAL DE SEÑALES.

MENUO.M

```
%
%          !!!! PROCESAMIENTO DIGITAL DE SEÑALES !!!!
%
%
%          1) ESCOGER TIPO DE PLANTA A SIMULAR
%          2) ANALISIS ESPECTRAL
%          3) DISEÑO DE FILTROS DIGITALES
%          4) FILTRADO DIGITAL DE SEÑALES CONTAMINADAS
%          5) AJUSTE POLINOMIAL: INTERPOLACION Y EXTRAPOLACION
%
%          CTRL + C: RETORNO A LA LINEA DE COMANDOS DEL MATLAB
%
%          0) SALIR AL DOS
%
%
%          ESCUELA POLITECNICA NACIONAL
%
%          FACULTAD DE INGENIERIA ELECTRICA
%
```

PDS.M

```
clc
delete pds.mat
%
%       EL PROGRAMA PERMITE ESCOGER LAS DISTINTAS OPCIONES DEL
%       PROCESAMIENTO DIGITAL DE SEÑALES
%
```

```
echo off
while 1
        item = ['escoger '
                'espectro'
                'filtros '
                'filtrado'
                'modelo  '];
        clc
        help menu0
        n0 = input('    SELECCIONAR OPCION: ');
        if ((n0<=0) | (n0>5))
                save
                clear
                exit
        end
        item = item (n0,:);
        eval (item)
end
clc
```

A.2.1. SIMULACION DE PLANTAS.

MENU1.M

```
%
%
%          ##### TIPOS DE PLANTAS #####
%
%                  1) DE PRIMER ORDEN.
%                  2) DE PRIMER ORDEN CON RETARDO.
%                  3) DE SEGUNDO ORDEN.
%                  4) DE SEGUNDO ORDEN CON RETARDO.
%
%                  0) retorno.
%
%
%
%
```

ESCOGER.M

```
% Este programa posibilita el escoger una opción del menú
% de selección del tipo de planta:
while 1
        item=['primer1 '
              'primer1r'
              'segundo2'
              'retardo '];
        clc
```

```
        help menui
        n=input('       Seleccione un tipo de planta: ');
        if ((n<=0) | (n>4))
             break
        end
        item=item(n,:);
        eval(item)
end
return
clc
```

PRIMER1.M

```
clc
echo on
% Este programa genera la respuesta de una planta de 1er orden a
% una entrada paso, o a una entrada definida, sin ruido y con la
% introducción de ruido aleatorio.
%
% La forma de la función de transferencia de la planta es:
%
%                      K
%      F.T.(S) = ------------
%                  aIS + b
%
echo off
delete pds.mat
j=sqrt(-1);
fs=input('El valor de la frecuencia de muestreo es fs (Hz):  ');
k=input('El valor de K es: ');
a=input('El valor de a es: ');
b=input('El valor de b es: ');
ds=input('El porcentaje de ruido con respecto al valor estable es (0-100%): ');
clc
echo on
%
%
% 1) Respuesta a la señal paso.
% 2) Respuesta a una señal cualquiera.
%
%
echo off
es=input('Escoja el tipo de señal de entrada a la planta: ');
echo on
clc
%
%
% 1) Introducción de ruido con distribución normal.
% 2) Introducción de ruido con distribución uniforme.
% 3) Introducción de ruido PRBS.
%
%
echo off
ruido=input('Escoja el tipo de distribución de la señal de ruido: ');
```

```
t=0:1/fs:999/fs;
clc
echo on
% espere por favor...
echo off
if ruido==1
rand('normal')
g=k*rand(t)/(3*b);
elseif ruido==2
rand('uniform')
g=k*2*(rand(t)-0.5)/b;
elseif ruido==3
eval(prbs);
load temp
g=(k/b)*sal(1:1000);
end
veruido=input('Desea ver señal de ruido que se va a introducir?  1=SI, 2=NO ');
if veruido==1
if ruido==3
plot(to(1:1000),g(1:1000)*ds/100),title('SEÑAL DE RUIDO QUE SE INTRODUCE'),...
xlabel('tiempo(seg)'),pause
else
plot(t,g*ds/100),title ('SEÑAL DE RUIDO QUE SE INTRODUCE'),...
xlabel('tiempo(seg)'),pause
end
elseif veruido==2
end
if es==1
        if ruido==3
        y=step(k,[a,b],to(1:1000));
        else
        y=step(k,[a,b],t);
        end
        h=g';
%        y1 = 1 + (ds/100)*g;
        y2=y+(h*ds/100);
%        y2=lsim(k,[a,b],y1,t);
elseif es==2
        if ruido==3
        y0=input('la señal de entrada al sistema es [f(to)]:  ');
        y=lsim(k,[a,b],y0(1:1000),to(1:1000));
        else
        y0=input('la señal de entrada al sistema es [f(t)]:  ');
        y=lsim(k,[a,b],y0,t);
        end
        h=g';
%        y1=y0+(ds/100)*g;
        y2=y+(h*ds/100);
%        y2=lsim(k,[a,b],y1,t);
end
echo on
%
%
pause % oprima cualquier tecla para graficar
%
%
echo off
if es==1
```

```
        subplot(211),plot(t(1:1000),y(1:1000)),title('Respuesta a entrada paso'),xlabel...
        ('tiempo(seg)');
        if ruido==3
        subplot(212),plot(to(1:1000),y2(1:1000)),title('Respuesta a entrada paso con ruido'),...
        xlabel('tiempo(seg)');
        else
        subplot(212),plot(t(1:1000),y2(1:1000)),title('Respuesta a entrada paso con ruido'),...
        xlabel('tiempo(seg)');
        end
        pause
        clg
elseif es==2
        subplot(211),plot(t(1:200),y(1:200)),title('Respuesta a entrada definida'),...
        xlabel('tiempo(seg)');
        if ruido==3
        subplot(212),plot(to(1:200),y2(1:200)),title('Respuesta a entrada definida con ruido'),...
        xlabel('tiempo(seg)');
        else
        subplot(212),plot(t(1:200),y2(1:200)),title('Respuesta a entrada definida con ruido'),...
        xlabel('tiempo(seg)');
        end
        pause
        clg
end
clc
zoom=input('Desea modificar el fondo de escala del gráfico?: 1=SI, 2=NO ');
if zoom==1
pun=input('Con el fin de visualizar mejor el gráfico digite un valor entre .1 y 10: ');
subplot(211),plot(t(1:pun*100),y(1:pun*100)),title('Respuesta a la función definida'),...
xlabel('tiempo(seg)');
subplot(212),plot(t(1:pun*100),y2(1:pun*100)),title('Con introducción de ruido'),...
xlabel('tiempo(seg)');
pause
elseif zoom==2
end
clc
clg
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
clg
if zoom==1
plot(t(1:pun*100),y(1:pun*100)),meta pds,title('Respuesta a la función definida'),...
xlabel('tiempo(seg)'),pause;
plot(t(1:pun*100),y2(1:pun*100)),meta,title('Con introducción de ruido'),...
xlabel('tiempo(seg)');
pause
elseif zoom==2
plot(t(1:1000),y(1:1000)),meta          pds,title('Respuesta          a          la          función
definida'),xlabel('tiempo(seg)'),pause;
plot(t(1:1000),y2(1:1000)),meta,title('Con introducción de ruido'),xlabel('tiempo(seg)'),pause;
end
clc
eval(imprimir)
elseif imprime==2
end
clc
clg
```

PRIMER1R.M

```
clc
echo on
% Este programa analiza las respuestas de una planta de 1er orden
% con tiempo de retardo (Td), mediante la aproximación de Paddé a
% una entrada paso y a una señal definida, con y sin ruido aleato-
% rio; el modelo de la planta es el siguiente:
%
%                                      -S.Td
%                                   K.e
%                     F.T.(S) =   ---------------
%                                   a*S + b
%
%
% La aproximación de Paddé de 2do. orden es:
%
%                                          2    2
%                     -S.Td      (1 - S.Td/2 + S^ .Td^ /8)
%                    e       =  --------------------------
%                                          2    2
%                               (1 + S.Td/2 + S^ .Td^ /8)
%
%
echo off
delete pds.mat
j=sqrt(-1);
fs=input('El valor de la frecuencia de muestreo es fs (Hz):  ');
k=input('El valor de K es: ');
a=input('El valor de a es: ');
b=input('El valor de b es: ');
td=input('El valor de Td es: ');
ds=input('El ruido en porcentaje del valor estable de la señal es (0-100%): ');
t=0:1/fs:999/fs;
c=td/2;
d=td*td/8;
clc
echo on
%
%
% 1) Respuesta a la señal paso.
% 2) Respuesta a una señal cualquiera.
%
%
echo off
es=input('Escoja el tipo de señal de entrada a la planta:  ');
echo on
clc
%
%
% 1) Introducción de ruido con distribución normal.
% 2) Introducción de ruido con distribución uniforme.
% 3) Introducción de ruido PRBS.
%
%
```

```
echo off
ruido=input('Escoja el tipo de distribución de la señal de ruido: ');
if ruido==1
rand('normal')
g=k*rand(t)/(3*b);
elseif ruido==2
rand('uniform')
g=k*2*(rand(t)-0.5)/b;
elseif ruido==3
eval(prbs);
load temp
g=(k/b)*sal(1:1000);
end
veruido=input('Desea ver señal de ruido que se va a introducir?  1=SI, 2=NO ');
if veruido==1
plot(t,g*ds/100),title ('SEÑAL DE RUIDO QUE SE INTRODUCE'),...
xlabel('tiempo(seg)'),pause;
elseif veruido==2
end
clc
apro=input('Desea la aproximación de Padé de 1ero o 2do grado?  ');
echo on
%
%
% espere por favor...
%
%
echo off
if td~=0
if apro==1
num=[-k*c k];
den=[a*c a+(b*c) b];
elseif apro==2
num=[k*d,-k*c,k];
den=[a*d,a*c+b*d,a+b*c,b];
end
elseif td==0
num=k;
den=[a,b];
end
if es==1
        y=step(num,den,t);
%        y1=1+ds*rand(t);
        h=g';
        y2=y+(h*ds/100);
%        y2=lsim(-num,den,y1,t);
elseif es==2
        y0=input('la señal de entrada al sistema es [f(t)]:  ');
        y=lsim(num,den,y0,t);
%        y1=y0+ds*rand(t);
%        y2=lsim(-num,den,y1,t);
        h=g';
        y2=y+(h*ds/100);
end
echo on
%
%
pause % oprima cualquier tecla para graficar
```

```
%
%
echo off
if es==1
        subplot(211),plot(t,y),title('Respuesta a la función paso'),...
        xlabel('tiempo(seg)');
        subplot(212),plot(t,y2),title('Con introducción de ruido'),...
        xlabel('tiempo(seg)');
        pause
        clg
elseif es==2
        subplot(211),plot(t(1:500),y(1:500)),title('Respuesta a la función definida'),...
        xlabel('tiempo(seg)');
        subplot(212),plot(t(1:500),y2(1:500)),title('Con introducción de ruido'),...
        xlabel('tiempo(seg)');
        pause
        clg
end
clc
zoom=input('Desea modificar el fondo de escala del gráfico? : 1=SI, 2=NO ');
if zoom==1
pun=input('Con el fin de visualizar mejor el gráfico digite un valor entre .1 y 10: ');
subplot(211),plot(t(1:pun*100),y(1:pun*100)),title('Respuesta a la función definida'),...
xlabel('tiempo(seg)');
subplot(212),plot(t(1:pun*100),y2(1:pun*100)),title('Con introducción de ruido'),...
xlabel('tiempo(seg)');
pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
clg
if zoom==1
plot(t(1:pun*100),y(1:pun*100)),meta pds,title('Respuesta a la función definida'),...
xlabel('tiempo(seg)'),pause;
plot(t(1:pun*100),y2(1:pun*100)),meta,title('Con introducción de ruido'),...
xlabel('tiempo(seg)');
pause
elseif zoom==2
plot(t(1:1000),y(1:1000)),meta pds,title('Respuesta a la función
definida'),xlabel('tiempo(seg)'),pause;
plot(t(1:1000),y2(1:1000)),meta,title('Con introducción de ruido'),xlabel('tiempo(seg)'),pause;
end
clc
eval(imprimir)
elseif imprime==2
end
clc
clg
```

SEGUNDO2.M

```
clc
echo on
```

```
% Este programa analiza la respuesta de una planta de 2do. orden a
% una entrada paso, y una función definida, sin ruido y con la in-
% troducción de un ruido aleatorio.
%
% El modelo de la función de transferencia de la planta es:
%
%                                    2
%                                 K*Wn^
%              F.T.(S) = ----------------------
%                            2                 2
%                         S^  + 2*e*Wn*S + Wn^
%
echo off
delete pds.met
j=sqrt(-1);
fs=input('El valor de la frecuencia de muestreo es fs (Hz):  ');
k=input('El valor de K es: ');
wn=input('El valor de Wn es: ');
e=input('El valor de e es: ');
ds=input('El ruido en porcentaje del valor estable de la señal es (0-100%): ');
echo on
clc
%
%
% 1) Respuesta a la señal paso.
% 2) Respuesta a una señal cualquiera.
%
%
echo off
es=input('Escoja el tipo de señal de entrada a la planta:  ');
echo on
clc
%
%
% 1) Introducción de ruido con distribución normal.
% 2) Introducción de ruido con distribución uniforme.
% 3) Introducción de ruido PRBS.
%
%
echo off
ruido=input('Escoja el tipo de distribución de la señal de ruido: ');
t=0:1/fs:999/fs;
echo on
clc
% espere por favor...
echo off
if ruido==1
rand ('normal');
g=k*(rand(t)/3);
elseif ruido==2
rand ('uniform');
g=k*2*(rand(t)-0.5);
elseif ruido==3
eval(prbs);
load temp;
g=k*sal(1:1000);
end
h=g';
```

```
veruido=input('Desea ver señal de ruido que se va a introducir?  1=SI, 2=NO ');
if veruido==1
plot(t,g*ds/100),title ('SEÑAL DE RUIDO QUE SE INTRODUCE'),...
xlabel('tiempo(seg)'),pause
elseif veruido==2
end
if es==1
       y=step(k*wn*wn,[1,2*e*wn,wn*wn],t);
%       y1=1 + ds*rand(t);
%       y2=lsim(wn*wn,[1,2*e*wn,wn*wn],y1,t);
       y2=y+h*ds/100;
elseif es==2
       y0=input('la señal de entrada al sistema es [r(t)]:  ');
       y=lsim(k*wn*wn,[1,2*e*wn,wn*wn],y0,t);
%       y1=y0+ds*rand(t);
%       y2=lsim(wn*wn,[1,2*e*wn,wn*wn],y1,t);
        y2=y+h*ds/100;
end
echo on
%
%
pause % oprima cualquier tecla para graficar
%
%
echo off
if es==1
       subplot(211),plot(t,y),title('Respuesta a la función paso'),...
       xlabel('tiempo(seg)');
       subplot(212),plot(t,y2),title('Respuesta con introducción de ruido'),...
       xlabel('tiempo(seg)');
       pause
       clg
elseif es==2
       subplot(211),plot(t(1:500),y(1:500)),title('Respuesta a la función definida'),...
       xlabel('tiempo(seg)');
       subplot(212),plot(t(1:500),y2(1:500)),title('Respuesta con introducción de ruido'),...
       xlabel('tiempo(seg)');
       pause
       clg
end
clc
zoom=input('Desea modificar el fondo de escala del gráfico? : 1=SI, 2=NO  ');
if zoom==1
pun=input('Con el fin de visualizar mejor el gráfico digite un valor entre .1 y 10: ');
subplot(211),plot(t(1:pun*100),y(1:pun*100)),title('Respuesta a la función definida'),...
xlabel('tiempo(seg)');
subplot(212),plot(t(1:pun*100),y2(1:pun*100)),title('Con introducción de ruido'),...
xlabel('tiempo(seg)');
pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
clg
if zoom==1
plot(t(1:pun*100),y(1:pun*100)),meta pds,title('Respuesta a la función definida'),...
xlabel('tiempo(seg)'),pause;
```

```
plot(t(1:punt100),y2(1:punt100)),meta,title('Con introducción de ruido'),...
xlabel('tiempo(seg)');
pause
elseif zoom==2
plot(t(1:1000),y(1:1000)),meta          pds,title('Respuesta          a          la          función
definida'),xlabel('tiempo(seg)'),pause;
plot(t(1:1000),y2(1:1000)),meta,title('Con introducción de ruido'),xlabel('tiempo(seg)'),pause;
end
clc
eval(imprimir)
elseif imprime==2
end
clc
clg
```

RETARDO.M

```
clc
echo on
% Este programa analiza las respuestas de una planta de 2do orden
% con tiempo de retardo (Td), a una entrada paso y a una señal de-
% finida, con y sin ruido aleatorio; el modelo de la planta es el
% siguiente:
%
%                                        2   -S.Td
%                                   K.Wn^ .e^
%                     F.T.(S) = ------------------------
%                                    2              2
%                                  S^  + 2*e*Wn*S + Wn^
%
%
% La aproximación de Paddé de 2do. orden es:
%
%                                              2    2
%                      -S.Td      (1 - S.Td/2 + S^ .Td^ /8)
%                     e     =  ---------------------------
%                                              2    2
%                                (1 + S.Td/2 + S^ .Td^ /8)
%
%
echo off
delete pds.mat
j=sqrt(-1);
fs=input('El valor de la frecuencia de muestreo es fs (Hz):  ');
k=input('El valor de K es: ');
wn=input('El valor de Wn es: ');
e=input('El valor de e es: ');
td=input('El valor de Td es: ');
ds=input('El ruido en porcentaje del valor estable de la señal es (0-100%): ');
clc
echo on
%
%
% 1) Respuesta a la señal paso.
```

```
% 2) Respuesta a una señal cualquiera.
%
%
echo off
es=input('Escoja el tipo de señal de entrada a la planta:  ');
clc
echo on
%
%
% 1) Introducción de ruido con distribución normal.
% 2) Introducción de ruido con distribución uniforme.
% 3) Introducción de ruido PRBS.
%
%
echo off
ruido=input('Escoja el tipo de distribución de la señal de ruido: ');
t=0:1/fs:999/fs;
echo on
clc
%
% espere por favor...
%
echo off
if ruido==1
rand ('normal');
g=k*(rand(t)/3);
elseif ruido==2
rand ('uniform');
g=k*2*(rand(t)-0.5);
elseif ruido==3
eval(prbs);
load temp;
g=k*sal(1:1000);
end
h=g';
veruido=input('Desea ver señal de ruido que se va a introducir?  1=SI, 2=NO ');
if veruido==1
plot(t,g*ds/100),title ('SEÑAL DE RUIDO QUE SE INTRODUCE'),...
xlabel('tiempo(seg)'),pause
elseif veruido==2
end
clc
a=k*wn^wn;
b=2*e*wn;
c=wn^wn;
d=td/2;
e1=td*td/8;
if td~=0
apro=input('Desea la aproximación de Padé de 1ero o 2do grado?  ');
if apro==1
num=[-a*d,a];
den=[d,1+b*d,b+c*d,c];
if es==1
        y=step(num,den,t);
        y2=y+h*ds/100;
elseif es==2
        y0=input('la señal de entrada al sistema es [f(t)]:  ');
        y=lsim(num,den,y0,t);
```

```
        y2=y+hids/100;
end
elseif apro==2
if es==1
        y=step([aie1,-aid,a],[e1,d+bie1,1+bid+cie1,b+cid,c],t);
%        y1=1 + dsirand(t);
%        y2=lsim(wni*wn,[1,2*e*wn,wn*wn],y1,t);
        y2=y+hids/100;
elseif es==2
        y0=input('la señal de entrada al sistema es [f(t)]:  ');
        y=lsim([aie1,-aid,a],[e1,d+bie1,1+bid+cie1,b+cid,c],y0,t);
%        y1=y0+dsirand(t);
%        y2=lsim(wni*wn,[1,2*e*wn,wn*wn],y1,t);
        y2=y+hids/100;
end
end
elseif tc==0
if es==1
        y=step([aie1,-aid,a],[e1,d+bie1,1+bid+cie1,b+cid,c],t);
%        y1=1 + dsirand(t);
%        y2=lsim(wni*wn,[1,2*e*wn,wn*wn],y1,t);
        y2=y+hids/100;
elseif es==2
        y0=input('la señal de entrada al sistema es [f(t)]:  ');
        y=lsim([aie1,-aid,a],[e1,d+bie1,1+bid+cie1,b+cid,c],y0,t);
%        y1=y0+dsirand(t);
%        y2=lsim(wni*wn,[1,2*e*wn,wn*wn],y1,t);
        y2=y+hids/100;
end
end
echo on
%
%
pause % oprima cualquier tecla para graficar
%
%
echo off
if es==1
        subplot(211),plot(t,y),title('Respuesta a la función paso'),...
        xlabel('tiempo(seg)');
        subplot(212),plot(t,y2),title('Con introducción de ruido'),...
        xlabel('tiempo(seg)');
        pause
        clg
elseif es==2
        subplot(211),plot(t(1:500),y(1:500)),title('Respuesta a la función definida'),...
        xlabel('tiempo(seg)');
        subplot(212),plot(t(1:500),y2(1:500)),title('Con introducción de ruido'),...
        xlabel('tiempo(seg)');
        pause
        clg
end
clc
zoom=input('Desea modificar el fondo de escala del gráfico? : 1=SI, 2=NO ');
if zoom==1
pun=input('Con el fin de visualizar mejor el gráfico digite un valor entre .1 y 10: ');
subplot(211),plot(t(1:pun*100),y(1:pun*100)),title('Respuesta a la función definida'),...
xlabel('tiempo(seg)');
```

```
subplot(212),plot(t(1:punt100),y2(1:punt100)),title('Con introducción de ruido'),...
xlabel('tiempo(seg)');
pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
clg
if zoom==1
plot(t(1:punt100),y(1:punt100)),meta pds,title('Respuesta a la función definida'),...
xlabel('tiempo(seg)'),pause;
plot(t(1:punt100),y2(1:punt100)),meta,title('Con introducción de ruido'),...
xlabel('tiempo(seg)');
pause
elseif zoom==2
plot(t(1:1000),y(1:1000)),meta pds,title('Respuesta a entrada paso'),xlabel('tiempo(seg)'),pause;
plot(t(1:1000),y2(1:1000)),meta,title('Respuesta a entrada paso con
ruido'),xlabel('tiempo(seg)'),pause;
end
clc
eval(imprimir)
elseif imprime==2
end
clc
clg
```

A.2.2. ANALISIS ESPECTRAL.

ESPECTRO.M

```
echo on
clc
%
%       Este programa realiza el análisis espectral de potencia de una
%       señal contaminada, obtenida de cualquier planta utilizada como
%       modelo, o de cualquier señal definida por el usuario.
%
%       Se encuentra primeramente la transformada discreta de Fourier
%       de la señal contaminada, mediante la transformada rápida de
%       Fourier.
%
echo off
delete pds.met
escoge=input('Desea utilizar la señal obtenida de la planta?  1=SI, 2=NO  ');
if escoge==2
preg=input('Desea utilizar la función PRBS?: 1=SI, 2=NO  ');
if preg==1
eval(prbs);
load temp
for z=1:1000
u(z)=sal(z);
```

```
end
elseif preg==2
fs=input('La frecuencia de muestreo de los datos es (Hz): ');
to=0:1/fs:999*1/fs;
y3=input('Por favor defina la señal u(to) que se quiere analizar:                    ');
u=y3;
end
plot(to(1:1000),u(1:1000)),meta pds,title('Señal que se va a analizar:'), xlabel('tiempo (seg.)'),pause
elseif escoge==1
%fs=100;
u=y2;
plot(t,y2),meta pds,title('Señal que se va a analizar:'), xlabel('tiempo (seg.)'),pause
end
clc
echo on
%
% espere por favor...
%
echo off
Y = fft(u,256);
echo on
pause % oprima cualquier tecla
clc
%
%      Se halla la densidad espectral de potencia, la cual es una medida
%      de la energía a varias frecuencias:
%
% espere por favor...
%
echo off
Pyy = Y .* conj(Y);
echo on
pause % oprima cualquier tecla para graficar:
echo off
f = fs/256*(0:127);
plot(f,Pyy(1:128)),meta, title('Densidad espectral de potencia'),..
xlabel('Frecuencia (Hz)'), pause
zoom=input('Desea modificar el fondo de escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*128/f(128);
plot(f(1:fz1),Pyy(1:fz1)),meta, title('Densidad espectral de potencia'),..
xlabel('Frecuencia (Hz)'), pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

A.2.3. DISEÑO DE FILTROS DIGITALES.

MENU2.M

```
%
%
%
%          ++++ DISEÑO DE FILTROS ++++
%
%          1) FILTROS RECURSIVOS
%          2) FILTROS NO RECURSIVOS
%
%          0) retorno
%
%
%
%
```

FILTROS.M

```
%
%       El programa permite escoger un tipo de filtro digital:
%
%       - recursivos
%       - no recursivos
%
while 1
        item = ['recursiv'
                'norecur '];
        clc
        help menu2
        n1 = input ('       seleccionar tipo de filtro: ');
        if ((n1<=0) | (n1>2))
                break
        end
        item = item (n1,:);
        eval [item)
end
return
clc
```

RECURSIV.M

```
%       El programa permite escoger un método de diseño de filtro
%       recursivo.
%
        while 1 .
```

```
        item = ['bilineal'
                'donyule '
                'donprony'
                'doninvfz'];
        clc
        help menu3
        n2 = input('   seleccionar método de diseño: ');
        if ((n2<=0) | (n2>4))
                break
        end
        item = item (n2,:);
        eval (item)
end
return
clc
```

MENU3.M

```
%
%
%     !!! METODOS DISPONIBLES DE DISEÑO DE FILTROS RECURSIVOS !!!
%
%     1) UTILIZANDO LA TRANSFORMADA BILINEAL
%     2) METODO DE YULE-WALKER
%     3) METODO DE PRONY
%     4) METODO DE ERROR MINIMO CUADRADO EN EL DOMINIO DE FRECUENCIA
%
%     0) retorno
%
%
%
%
```

NORECUR.M

```
%       El programa permite escoger un método de diseño de filtro
%       no recursivo.
%
        while 1
              item = ['donfir1 '
                      'donfir2 '
                      'donremez'];
              clc
              help menu6
              n5 = input('          seleccionar método de diseño: ');
              if ((n5<=0) | (n5>3))
                    break
              end
              item = item (n5,:);
              eval (item)
```

```
        end
        return
        clc
```

MENU6.M

```
%
%
%          ┌──────────────────────────────────────────────┐
%          │  METODOS DE DISEÑO DE FILTROS NO RECURSIVOS  │
%          ├──────────────────────────────────────────────┤
%          │ 1) DISEÑO DE FASE LINEAL DE LONGITUD DEFINIDA│
%          │ 2) DISEÑO DE FASE LINEAL GENERAL             │
%          │ 3) DISEÑO POR EL ALGORITMO DE PARKS-McCLELLAN│
%          ├──────────────────────────────────────────────┤
%          │ 0) retorno                                   │
%          └──────────────────────────────────────────────┘
%
%
%
%
%
```

A.2.3.1. TRANSFORMADA BILINEAL.

BILINEAL.M

```
clc
%
%       Escoge el tipo de filtro por medio de aproximaciones a los filtros
%       análogos, utilizando la transformada bilineal.
%
while 1
        item = ['donbutt'
                'doncheb'
                'donelip'];
        clc
        help menu4
        n3 = input ('                    seleccionar un tipo de filtro: ');
        if ((n3<=0) | (n3>3))
                break
        end
        item = item (n3,:);
        eval (item)
end
return
clc
```

MENU4.M

```
%
%
%
%                  ¡¡¡¡ APROXIMACIONES ¡¡¡¡
%
%                  1) BUTTERWORTH
%                  2) CHEBYSHEV
%                  3) ELIPTICO
%
%                  0) retorno
%
%
%
%
```

DONBUTT.M

```
clc
echo on
%       Diseño de filtro digital de Butterworth
%
%       [B,A] = butter (N,Wn) diseña un filtro digital pasabajos de
%       Butterworth de orden N y retorna los coeficientes del filtro
%       en los vectores B y A de longitud N+1. La frecuencia de corte
%       Wn debe estar entre 0 y 1 con el valor de 1 correspondiente a
%       la mitad de la frecuencia de muestreo.
%
%       Si Wn es un vector de dos elementos, Wn = [W1 W2], butter re-
%       torna un filtro pasabanda de orden 2N con banda de paso:
%       W1 < W < W2.
%
%       [B,A] = butter (N,Wn,'high') diseña un filtro pasa altos.
%       [B,A] = butter (N,Wn,'stop') es un filtro elimina banda si
%       Wn = [W1 W2].
pause % oprima cualquier tecla
%
%       Siendo Rp el valor permisible en decibelios que pierde en la
%       región de paso y Rs el valor en decibelios que tiene como
%       mínimo de atenuación en la región de eliminación, la banda
%       de paso va desde 0 hasta Wp y la banda de eliminación se
%       extiende desde Ws hasta 1 que es la frecuencia de Nyquist.
pause % oprima cualquier tecla
%
%       Wn es la frecuencia natural de Butterworth.
%
echo off
delete pds.met
fs=input('El valor de la frecuencia de muestreo es (Hz): ');
echo on
%
```

```
%
% 1) filtro pasa bajos
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el tipo de filtro requerido: ');
clc
if tipo==1
%eval(pb)
help pb
Fp = input ('El valor de la frecuencia de paso es Fp (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es Fs (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = buttord (Fp*2/fs,Fs*2/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1): ');
eval(espere)
[B,A]=butter(N1,Wn1)
elseif tipo==2
%eval(pa)
help pa
Fp = input ('El valor de la frecuencia de paso es Fp (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es Fs (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = buttord (Fp*2/fs,Fs*2/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1): ');
eval(espere)
[B,A]=butter(N1,Wn1,'high')
elseif tipo==3
%eval(pban)
help pban
Fp = input ('El valor de la frecuencia de paso es [Fpl Fpu] (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es [Fsl Fsu] (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = buttord (Fp*2/fs,Fs*2/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1) [W1 W2]: ');
eval(espere)
[B,A]=butter(N1/2,Wn1)
elseif tipo==4
%eval(eban)
help eban
Fp = input ('El valor de la frecuencia de paso es [Fpl Fpu] (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es [Fsl Fsu] (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = buttord (Fp*2/fs,Fs*2/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1) [W1 W2]: ');
eval(espere)
[B,A]=butter(N1/2,Wn1,'stop')
```

```
end
echo on
pause % oprima cualquier tecla
clc
%
% Ahora calcula la magnitud de respuesta de frecuencia compleja:
%
% espere por favor...
%
echo off
[H,w]=freqz(B,A,512);
plot (w*fs/(2*pi),abs(H)),meta pds, title('Respuesta de frecuencia Butterworth obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia Butterworth obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

DONCHEB.M

```
clc
echo on
%       Diseño de filtro digital de Chebyshev
%
%       [B,A] = cheby (N,R,Wn) diseña un filtro digital pasabajos de
%       Chebyshev de orden N y retorna los coeficientes del filtro
%       en los vectores B y A de longitud N+1, con R decibelios en la
%       región de paso o de eliminación, según el tipo de filtro es-
%       cogido (1 o 2).
%
%       La frecuencia de corte Wn debe estar entre 0 y 1 con el valor
%       de 1 correspondiente a la mitad de la frecuencia de muestreo.
%
%       Si Wn es un vector de dos elementos, Wn = [W1 W2], cheby re-
%       torna un filtro pasabanda de orden 2N con banda de paso:
%       W1 < W < W2.
%
%       Use R=0.5 como un punto de inicio, si no está seguro acerca de
%       su valor.
%
%       [B,A] = cheby (N,R,Wn,'high') diseña un filtro pasa altos.
%       [B,A] = cheby (N,R,Wn,'stop') es un filtro elimina banda si
```

```
%       Wn = [W1 W2].
pause % oprima cualquier tecla
%
%       Siendo R el valor permisible en decibelios que pierde en la
%       región de paso y R el valor en decibelios que tiene como
%       mínimo de atenuación en la región de eliminación, según si se
%       escoge un filtro de Chebyshev de tipo 1 o 2 respectivamente.
pause % oprima cualquier tecla
%
%       Wn es la frecuencia natural de Chebyshev.
%
echo off
delete pds.mat
fs=input('El valor de la frecuencia de muestreo es (Hz): ');
echo on
%
%
% 1) Filtro de Chebyshev de tipo I
% 2) Filtro de Chebyshev de tipo II
%
%
echo off
clase=input('seleccione el tipo de filtro: ')
clc
if clase==1
eval(donchebi)
elseif clase==2
eval(doncheb2)
end
echo on
pause % oprima cualquier tecla
clc
%
% Ahora calcula la magnitud de respuesta de frecuencia compleja:
%
% espere por favor...
%
echo off
[H,w]=freqz(B,A,512);
plot (w*fs/(2*pi),abs(H)),meta pds, title('Respuesta de frecuencia Chebyshev obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta,title('Respuesta de frecuencia Chebyshev obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

DONCHEB1.M

```
echo on
%
%
% 1) filtro pasa bajos
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el filtro requerido: ');
clc
if tipo==1
eval(pb)
Fp = input ('El valor de la frecuencia de paso es Fp (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es Fs (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = cheb1ord (Fp*2/fs,Fs*2/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1): ');
R=input('El valor permisible de rizado en la región de paso es R (dB): ');
eval(espere)
[B,A]=cheby1(N1,R,Wn1)
elseif tipo==2
eval(pa)
Fp = input ('El valor de la frecuencia de paso es Fp (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es Fs (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = cheb1ord (Fp*2/fs,Fs*2/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1): ');
R=input('El valor permisible de rizado en la región de paso es R (dB): ');
eval(espere)
[B,A]=cheby1(N1,R,Wn1,'high')
elseif tipo==3
eval(pban)
Fp = input ('El valor de la frecuencia de paso es [Fpl Fpu] (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es [Fsl Fsu] (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = cheb1ord (Fp*2/fs,Fs*2/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1) [W1 W2]: ');
R=input('El valor permisible de rizado en la región de paso es R (dB): ');
eval(espere)
[B,A]=cheby1(N1/2,R,Wn1)
elseif tipo==4
eval(eban)
Fp = input ('El valor de la frecuencia de paso es [Fpl Fpu] (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es [Fsl Fsu] (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
```

```
[N,Wn] = chebiord (Fp12/fs,Fs12/fs,Rp,Rs)
Ni= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1) [W1 W2]: ');
R=input('El valor permisible de rizado en la región de paso es R (dB): ');
eval(espere)
[B,A]=cheby1(N1/2,R,Wn1,'stop')
end
```

DONCHEB2.M

```
echo on
%
%
% 1) filtro pasa bajos
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el filtro requerido: ');
clc
if tipo==1
eval(pb)
Fp = input ('El valor de la frecuencia de paso es Fp (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es Fs (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = cheb2ord (Fp12/fs,Fs12/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1): ');
R=input('El valor mínimo de atenuación en la región de eliminación es R (dB): ');
eval(espere)
[B,A]=cheby2(N1,R,Wn1)
elseif tipo==2
eval(pa)
Fp = input ('El valor de la frecuencia de paso es Fp (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es Fs (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = cheb2ord (Fp12/fs,Fs12/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1): ');
R=input('El valor mínimo de atenuación en la región de eliminación es R (dB): ');
eval(espere)
[B,A]=cheby2(N1,R,Wn1,'high')
elseif tipo==3
eval(pban)
Fp = input ('El valor de la frecuencia de paso es [Fpl Fpu] (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es [Fsl Fsu] (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = cheb2ord (Fp12/fs,Fs12/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
```

```
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1) [W1 W2]: ');
R=input('El valor mínimo de atenuación en la región de eliminación es R (dB): ');
eval(espere)
[B,A]=cheby2(N1/2,R,Wn1)
elseif tipo==4
eval(eban)
Fp = input ('El valor de la frecuencia de paso es [Fpl Fpu] (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es [Fsl Fsu] (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = cheb2ord (Fp12/fs,Fs12/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1) [W1 W2]: ');
R=input('El valor mínimo de atenuación en la región de eliminación es R (dB): ');
eval(espere)
[B,A]=cheby2(N1/2,R,Wn1,'stop')
end
```

DONELIP.M

```
clc
echo on
%       Diseño de filtro digital Elíptico o de Cauer
%
%       [B,A] = ellip (N,Rp,Rs,Wn) diseña un filtro digital pasabajos
%       elíptico de orden N y retorna los coeficientes del filtro
%       en los vectores B y A de longitud N+1, con Rp decibelios de
%       rizado en la región de paso y una atenuación de Rs decibelios
%       en la región de eliminación.
%
%       La frecuencia de corte Wn debe estar entre 0 y 1 con el valor
%       de 1 correspondiente a la mitad de la frecuencia de muestreo.
%       Utilice Rp=0.5 y Rs=20 como punto inicial de partida.
%
%       Si Wn es un vector de dos elementos, Wn = [W1 W2], ellip re-
%       torna un filtro pasabanda de orden 2N con banda de paso:
%       W1 < W < W2.
%
%       [B,A] = ellip (N,Rp,Rs,Wn,'high') diseña un filtro pasa altos.
%       [B,A] = ellip (N,Rp,Rs,Wn,'stop') es un filtro elimina banda si
%       Wn = [W1 W2].
pause % oprima cualquier tecla
%
%       Siendo Rp el valor permisible en decibelios que pierde en la
%       región de paso y Rs el valor en decibelios que tiene como
%       mínimo de atenuación en la región de eliminación, la banda
%       de paso va desde 0 hasta Wp y la banda de eliminación se
%       extiende desde Ws hasta 1 que es la frecuencia de Nyquist.
pause % oprima cualquier tecla
%
%       Wn es la frecuencia natural elíptica.
%
echo off
delete pds.met
```

```
fs=input('El valor de la frecuencia de muestreo es (Hz): ');
echo on
%
%
% 1) filtro pasa bajos
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el tipo de filtro requerido: ');
clc
if tipo==1
eval(pb)
Fp = input ('El valor de la frecuencia de paso es Fp (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es Fs (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = ellipord (Fp*2/fs,Fs*2/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1): ');
eval(espere)
[B,A]=ellip(N1,Rp,Rs,Wn1)
elseif tipo==2
eval(pa)
Fp = input ('El valor de la frecuencia de paso es Fp (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es Fs (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminacion es Rs (dB): ');
[N,Wn] = ellipord (Fp*2/fs,Fs*2/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1): ');
eval(espere)
[B,A]=ellip(N1,Rp,Rs,Wn1,'high')
elseif tipo==3
eval(pban)
Fp = input ('El valor de la frecuencia de paso es [Fpl Fpu] (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es [Fsl Fsu] (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = ellipord (Fp*2/fs,Fs*2/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1) [W1 W2]: ');
eval(espere)
[B,A]=ellip(N1/2,Rp,Rs,Wn1)
elseif tipo==4
eval(eban)
Fp = input ('El valor de la frecuencia de paso es [Fpl Fpu] (Hz): ');
Fs = input ('El valor de la frecuencia de eliminación es [Fsl Fsu] (Hz): ');
Rp = input ('El valor permisible de rizado en la región de paso es Rp (dB): ');
Rs = input ('El valor de atenuación en la región de eliminación es Rs (dB): ');
[N,Wn] = ellipord (Fp*2/fs,Fs*2/fs,Rp,Rs)
N1= input('El nuevo orden del filtro es (par): ');
Wn1 = input ('El valor de la frecuencia natural es (entre 0 y 1) [W1 W2]: ');
eval(espere)
[B,A]=ellip(N1/2,Rp,Rs,Wn1,'stop')
end
```

```
echo on
pause % oprima cualquier tecla
clc
%
% Ahora calcula la magnitud de respuesta de frecuencia compleja:
%
% espere por favor...
%
echo off
[H,w]=freqz(B,A,512);
plot (w*fs/(2*pi),abs(H)),meta pds,title('Respuesta de frecuencia Eliptica obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud');
pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia Eliptica obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud');
pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

A.2.3.2. METODO DE YULE-WALKER.

DONYULE.M

```
clc
echo on
%       YULEWALK permite el diseño de un filtro recursivo, usando un mé-
%       todo de mínimos cuadrados.
%
%       [B,A] = YULEWALK (N,F,M) encuentra los coeficientes B y A del fil-
%       tro recursivo de orden N tal que el filtro:
%
%                                  -1           -(n-1)
%       B(z)     b(1) + b(2)z  +...+ b(n)z
%      ----- = ------------------------------------
%                                  -1           -(n-1)
%       A(z)       1 + a(1)z  +...+ a(n)z
%
%
%       empareja la magnitud de la respuesta de frecuencia dada por los
%       vectores F y M.
```

```
%
%       Los vectores F y M especifican los puntos de corte de la frecuencia
%       y la magnitud  (0-1) para el filtro tal que al graficar M vs. F de-
%       bería mostrarse un gráfico de la respuesta de frecuencia deseada.
%       Las frecuencias en F deben estar entre 0 y la mitad de la frecuen-
%
pause % oprima cualquier tecla
%
%       cia de muestreo, ordenadas en orden ascendente y comenzando con 0
%       y terminando con la frecuencia correspondiente a la mitad de la
%       frecuencia de muestreo fs.
%
%       Mediante el método de Yule-Walker es posible realizar diseños de
%       filtros multibanda.
%
pause % oprima cualquier tecla
clc
eval(tutor)
echo off
delete pds,met
clc
fs=input('La frecuencia de muestreo es (Hz): ');
N=input('El orden del filtro es: ');
F=input('El vector de frecuencias es (Hz): ');
M=input('El vector de magnitudes es: ');
echo on
% espere por favor...
%
% el gráfico de la respuesta de frecuencia deseada será:
%
pause % oprima cualquier tecla para graficar
echo off
F0=2*F/fs
plot(F,M),meta pds,title('Respuesta de Frecuencia deseada'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'), pause
clc
preg=input('La respuesta de frecuencia deseada está correcta? 1=SI 2=NO : ');
if preg==1
eval(donyule1)
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
elseif preg==2
end
clc
clg
```

DONYULE1.M

```
echo on
%
%
% espere por favor...
```

```
%
echo off
[B,A] = yulewalk(N,F0,M);
echo on
%
% El diseño está completo y ahora se puede mirar los coeficientes del filtro:
%
pause % oprima cualquier tecla
echo off
format compact
clc
B,A
echo on
%
% estos son los coeficientes del filtro
%
% ahora se calcula la respuesta de frecuencia del filtro diseñado y se la
% compara con la del filtro deseado:
%
pause % oprima cualquier tecla
%
% espere por favor...
echo off
format
H=freqz(B,A,512);
Ha=abs(H);
F1=fs/(2*512)*(0:511);
echo on
%
pause % oprima cualquier tecla para graficar
%
echo off
plot (F,M,F1,Ha),meta,title('Respuesta de frecuencia deseada vs. diseñada'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
zoom=input('Desea modificar el fondo de escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/F1(512);
plot (F1(1:fz1),Ha(1:fz1)),meta,title('Respuesta de frecuencia Yule-Walker'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
elseif zoom==2
end
clc
clg
```

A.2.3.3. METODO DE PRONY.

DONPRONY.M

```
clc
echo on
%
% Método de Prony para el diseño de filtros de respuesta impulso infinita
% en el dominio del tiempo.
```

```
%
% [B,A] = PRONY (H,NB,NA) halla un filtro de orden NB en el numerador y
% un denominador de orden NA, teniendo la respuesta impulso en el vector
% H. Los coeficientes del filtro de respuesta impulso infinita (IIR) son
% retornados en vectores fila B y A, de longitud NB+1 y NA+1 respectiva-
% mente ordenados en potencias descendentes de Z.
%
% Este método se puede utilizar como una alternativa de diseño de un fil-
% tro digital, siempre y cuando se haya diseñado un filtro digital con
% cualquiera de los métodos de diseño de filtros recursivos anteriormen-
% te descritos, ya que la respuesta impulso que se analiza es la de un
% filtro ya diseñado, que cumple con los requerimientos, pero del que se
% desea modificar el orden de los coeficientes de numerador y/o denomina-
% dor para fines prácticos.
%
%
echo off
delete pds.met
NB=input('El orden deseado del numerador es: ');
NA=input('El orden deseado del denominador es: ');
clc
echo on
%
% Ahora se procede a calcular la respuesta impulso del filtro diseñado
% por cualquiera de los métodos anteriores.
%
pause % oprima cualquier tecla...
%
%
echo off
ti=0:1/(1000*fs):1/fs;
Hi=dimpulse(B,A,500);
echo on
%
pause % oprima cualquier tecla para graficar la respuesta impulso...
%
echo off
plot(ti(1:500),Hi),meta pds, title('RESPUESTA IMPULSO DEL FILTRO'), xlabel('tiempo')
pause
clc
echo on
%
% Ahora se puede diseñar el filtro digital, mediante la respuesta
% impulso obtenida...
%
% espere por favor...
echo off
[B,A]=prony(Hi(1:50),NB,NA)
echo on
%
pause % el filtro está diseñado, oprima cualquier tecla...
%
%
echo off
[H,w]=freqz(B,A,512);
plot (w*fs/(2*pi),abs(H)),meta, title('Respuesta de frecuencia Prony obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
```

```
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia Prony obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

A.2.3.4. METODO DE MINIMIZACION DEL ERROR MEDIO CUADRATICO.

DONINVFZ.M

```
clc
echo on
% La función INVFREQZ es la operación inversa de FREQZ, y puede conver-
% tir datos de magnitud y fase en una función de transferencia de la
% transformada Z en tiempo discreto.
%
% [B,A] = INVFREQZ (H,W,NB,NA) retorna los coeficientes del numerador
% y el denominador del filtro en los vectores B y A desde la función
% de transferencia cuya respuesta de frecuencia compleja está dada en
% el vector H a los puntos de frecuencia especificados en el vector W.
%
% Los escalares NB y NA especifican los órdenes deseados de los polino-
% mios del numerador y el denominador. La frecuencia está especificada
% en radianes/seg entre cero y pi, y la longitud de W es igual a la
% longitud de H.
%
% Este método se puede utilizar como una alternativa de diseño de un fil-
% tro digital, siempre y cuando se haya diseñado un filtro digital con
% cualquiera de los métodos de diseño de filtros recursivos anteriormen-
% te descritos, ya que la respuesta de frecuencia compleja que se analiza
% es la de un filtro ya diseñado, que cumple con los requerimientos, pero
% del que se desea modificar el orden de los coeficientes de numerador y/o
% denominador para fines prácticos.
%
pause % oprima cualquir tecla
echo off
delete pds.met
NB=input('El orden deseado del numerador es: ');
NA=input('El orden deseado del denominador es: ');
clc
echo on
%
```

```
% Ahora se procede a calcular el nuevo filtro, mediante la respuesta de
% frecuencia compleja del filtro diseñado anteriormente.
%
pause % oprima cualquier tecla...
%
%
% espere por favor...
%
echo off
[B,A]=invfreqz(H,w,NB,NA)
echo on
%
pause % el filtro está diseñado, oprima cualquier tecla...
%
%
echo off
[H,w]=freqz(B,A,512);
plot (w*fs/(2*pi),abs(H)),meta pds, title('Respuesta de frecuencia obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

A.2.3.5. METODO DE FASE LINEAL.

DONFIR1.M

```
%       El programa permite escoger una ventana para el diseño de filtro
%       no recursivo.
%
        while 1
              item = ['donhamm '
                      'donhann '
                      'donkais '
                      'donbart '
                      'donblac '
                      'donrect '
                      'dontria '
                      'donchew '
                      'pds     '];
```

```
        clc
        help menu7
        n6 = input('            seleccionar tipo de ventana: ');
        if ((n6<=0) | (n6>9))
                break
        end
        item = item (n6,:);
        eval (item)
  end
  return
  clc
```

MENU7.M

```
%
%          TIPOS DE VENTANAS PARA FILTROS NO RECURSIVOS
%
%                1) HAMMING
%                2) HANNING
%                3) KAISER
%                4) BARTLETT
%                5) BLACKMAN
%                6) RECTANGULAR
%                7) TRIANGULAR
%                8) CHEBYSHEV
%
%                0) retorno
%
%
%
```

DONHAMM.M

```
clc
echo on
%       Diseño de filtro digital usando método de ventanas
%
%       B = fir1 (N,Wn) diseña un filtro digital pasabajos de
%       FIR de orden N y retorna los coeficientes del filtro
%       en el vector B de longitud N+1. La frecuencia de corte
%       Wn debe estar entre 0 y 1 con el valor de 1 correspon-
%       diente a la mitad de la frecuencia de muestreo.
%
%       Si Wn es un vector de dos elementos, Wn = [W1 W2], fir1
%       retorna un filtro pasabanda de orden N con banda de paso:
%       W1 < W < W2.
%
%       [B,A] = fir1 (N,Wn,'high') diseña un filtro pasa altos.
%       [B,A] = fir1 (N,Wn,'stop') es un filtro elimina banda si
%       Wn = [W1 W2].
%
%       Sin ninguna especificación, fir1 usa una ventana de Ha-
```

```
%       mming
%
%       Para filtros pasa altos y elimina banda, N debe ser par.
%
pause % oprima cualquier tecla
%
echo off
delete pds.mat
fs=input('El valor de la frecuencia de muestreo es (Hz): ');
echo on
%
%
% 1) filtro pasa bajos
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el tipo de filtro requerido: ');
clc
format compact
if tipo==1
eval(pbi)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
Ni= input('El orden del filtro es: ');
Wni = Fc*2/fs;
eval(espere)
B=fir1(Ni,Wni)
elseif tipo==2
eval(pai)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
Ni= input('El orden del filtro es (par): ');
Wni = Fc*2/fs;
eval(espere)
B=fir1(Ni,Wni,'high')
elseif tipo==3
eval(pbani)
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu] (Hz): ');
Ni= input('El orden del filtro es: ');
Wni = Fc*2/fs;
eval(espere)
B=fir1(Ni,Wni)
elseif tipo==4
eval(ebani)
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu] (Hz): ');
Ni= input('El orden del filtro es (par): ');
Wni = Fc*2/fs;
eval(espere)
B=fir1(Ni,Wni,'stop')
end
format
echo on
pause % oprima cualquier tecla
clc
%
% Ahora calcula la magnitud de respuesta de frecuencia compleja:
%
```

```
% espere por favor...
%
echo off
[H,w]=freqz(B,1,512);
plot (w*fs/(2*pi),abs(H)),meta pds, title('Respuesta de frecuencia Hamming obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia Hamming obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

DONHANN.M

```
clc
echo on
%       Diseño de filtro digital usando método de ventanas
%
%       B = fir1 (N,Wn,hanning(N+1)) diseña un filtro digital
%       pasabajos FIR de orden N y retorna los coeficientes del
%       filtro en el vector B de longitud N+1. La frecuencia de
%       corte Wn debe estar entre 0 y 1 con el valor de 1 corres-
%       pondiente a la mitad de la frecuencia de muestreo.
%
%       Si Wn es un vector de dos elementos, Wn = [W1 W2], fir1
%       retorna un filtro pasabanda de orden N con banda de paso:
%       W1 < W < W2.
%
%       [B,A] = fir1 (N,Wn,'high',hanning(N+1)) diseña un filtro
%       pasa altos.
%       [B,A] = fir1 (N,Wn,'stop',hanning(N+1)) es un filtro eli-
%       mina banda si Wn = [W1 W2].
%
%       Para filtros pasa altos y elimina banda, N debe ser par.
%
pause % oprima cualquier tecla
%
echo off
delete pds.met
fs=input('El valor de la frecuencia de muestreo es (Hz): ');
echo on
%
%
% 1) filtro pasa bajos
```

```
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el tipo de filtro requerido: ');
clc
format compact
if tipo==1
eval(pb1)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
N1= input('El orden del filtro es: ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,hanning(N1+1))
elseif tipo==2
eval(pa1)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'high',hanning(N1+1))
elseif tipo==3
eval(pban1)
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu] (Hz): ');
N1= input('El orden del filtro es: ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,hanning(N1+1))
elseif tipo==4
eval(eban1)
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu] (Hz): ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'stop',hanning(N1+1))
end
format
echo on
pause % oprima cualquier tecla
clc
%
% Ahora calcula la magnitud de respuesta de frecuencia compleja:
%
% espere por favor...
%
echo off
[H,w]=freqz(B,1,512);
plot (w*fs/(2*pi),abs(H)),meta pds, title('Respuesta de frecuencia Hanning obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia Hanning obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
```

```
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

DONKAIS.M

```
clc
echo on
%       Diseño de filtro digital usando método de ventanas
%
%       B = fir1 (N,Wn,kaiser((N+1),beta) diseña un filtro digital
%       pasabajos FIR de orden N y retorna los coeficientes del
%       filtro en el vector B de longitud N+1. La frecuencia de
%       corte Wn debe estar entre 0 y 1 con el valor de 1 corres-
%       pondiente a la mitad de la frecuencia de muestreo.
%
%       Si Wn es un vector de dos elementos, Wn = [W1 W2], fir1
%       retorna un filtro pasabanda de orden N con banda de paso:
%       W1 < W < W2.
%
%       [B,A] = fir1 (N,Wn,'high',kaiser((N+1),beta) diseña un filtro
%       pasa altos.
%       [B,A] = fir1 (N,Wn,'stop',kaiser((N+1),beta) es un filtro eli-
%       mina banda si Wn = [W1 W2].
%
%       Para filtros pasa altos y elimina banda, N debe ser par.
%       La ventana es beta-evaluada.
%
pause % oprima cualquier tecla
%
echo off
delete pos.met
fs=input('El valor de la frecuencia de muestreo es (Hz): ');
beta=input('El valor de beta es: ');
echo on
%
%
% 1) filtro pasa bajos
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el tipo de filtro requerido: ');
clc
format compact
if tipo==1
eval(pbi)
```

```
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
Ni= input('El orden del filtro es: ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,kaiser((N1+1),beta))
elseif tipo==2
eval(pai)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
Ni= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'high',kaiser((N1+1),beta))
elseif tipo==3
eval(pban1)
Fc = input ('El valor de la frecuencia de corte es [Fc1 Fcu] (Hz): ');
Ni= input('El orden del filtro es: ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,kaiser((N1+1),beta))
elseif tipo==4
eval(eban1)
Fc = input ('El valor de la frecuencia de corte es [Fc1 Fcu] (Hz): ');
Ni= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'stop',kaiser((N1+1),beta))
end
format
echo on
pause % oprima cualquier tecla
clc
%
% Ahora calcula la magnitud de respuesta de frecuencia compleja:
%
% espere por favor...
%
echo off
[H,w]=freqz(B,1,512);
plot (w*fs/(2*pi),abs(H)),meta pds, title('Respuesta de frecuencia Kaiser obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia Kaiser obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

DONBART.M

```
clc
echo on
%       Diseño de filtro digital usando método de ventanas
%
%       B = fir1 (N,Wn,bartlett(N+1)) diseña un filtro digital
%       pasabajos FIR de orden N y retorna los coeficientes del
%       filtro en el vector B de longitud N+1. La frecuencia de
%       corte Wn debe estar entre 0 y 1 con el valor de 1 corres-
%       pondiente a la mitad de la frecuencia de muestreo.
%
%       Si Wn es un vector de dos elementos, Wn = [W1 W2], fir1
%       retorna un filtro pasabanda de orden N con banda de paso:
%       W1 < W < W2.
%
%       [B,A] = fir1 (N,Wn,'high',bartlett(N+1)) diseña un filtro
%       pasa altos.
%       [B,A] = fir1 (N,Wn,'stop',bartlett(N+1)) es un filtro eli-
%       mina banda si Wn = [W1 W2].
%
%       Para filtros pasa altos y elimina banda, N debe ser par.
%
pause % oprima cualquier tecla
%
echo off
delete pds.mat
fs=input('El valor de la frecuencia de muestreo es (Hz): ');
echo on
%
%
% 1) filtro pasa bajos
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el tipo de filtro requerido: ');
clc
format compact
if tipo==1
eval(pb1)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
N1= input('El orden del filtro es : ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,bartlett(N1+1))
elseif tipo==2
eval(pa1)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'high',bartlett(N1+1))
elseif tipo==3
eval(pban1)
```

```
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu] (Hz): ');
Ni= input('El orden del filtro es: ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,bartlett(N1+1))
elseif tipo==4
eval(ebanl)
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu] (Hz): ');
Ni= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'stop',bartlett(N1+1))
end
format
echo on
pause % oprima cualquier tecla
clc
%
% Ahora calcula la magnitud de respuesta de frecuencia compleja:
%
% espere por favor...
%
echo off
[H,w]=freqz(B,1,512);
plot (w*fs/(2*pi),abs(H)),meta pds, title('Respuesta de frecuencia Bartlett obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia Bartlett obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

DONBLAC.M

```
clc
echo on
%       Diseño de filtro digital usando método de ventanas
%
%       B = fir1 (N,Wn,blackman(N+1)) diseña un filtro digital
%       pasabajos FIR de orden N y retorna los coeficientes del
%       filtro en el vector B de longitud N+1. La frecuencia de
%       corte Wn debe estar entre 0 y 1 con el valor de 1 corres-
%       pondiente a la mitad de la frecuencia de muestreo.
%
%       Si Wn es un vector de dos elementos, Wn = [W1 W2], fir1
```

```
%       retorna un filtro pasabanda de orden N con banda de paso:
%       W1 < W < W2.
%
%       [B,A] = fir1 (N,Wn,'high',blackman(N+1)) diseña un filtro
%       pasa altos.
%       [B,A] = fir1 (N,Wn,'stop',blackman(N+1)) es un filtro eli-
%       mina banda si Wn = [W1 W2].
%
%       Para filtros pasa altos y elimina banda, N debe ser par.
%
pause % oprima cualquier tecla
%
echo off
delete pds.met
fs=input('El valor de la frecuencia de muestreo es (Hz): ');
echo on
%
%
% 1) filtro pasa bajos
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el tipo de filtro requerido: ');
clc
format compact
if tipo==1
eval(pb1)
Fc = input ('El valor de la frecuencia de corte es Fc: ');
N1= input('El orden del filtro es: ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,blackman(N1+1))
elseif tipo==2
eval(pa1)
Fc = input ('El valor de la frecuencia de corte es Fc: ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'high',blackman(N1+1))
elseif tipo==3
eval(pban1)
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu]: ');
N1= input('El orden del filtro es: ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,blackman(N1+1))
elseif tipo==4
eval(eban1)
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu]: ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'stop',blackman(N1+1))
end
format
```

```
echo on
pause % oprima cualquier tecla
clc
%
% Ahora calcula la magnitud de respuesta de frecuencia compleja:
%
% espere por favor...
%
echo off
[H,w]=freqz(B,1,512);
plot (w*fs/(2*pi),abs(H)),meta pds, title('Respuesta de frecuencia Blackman obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia Blackman obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

DONRECT.M

```
clc
echo on
%       Diseño de filtro digital usando método de ventanas
%
%       B = fir1 (N,Wn,boxcar(N+1)) diseña un filtro digital
%       pasabajos FIR de orden N y retorna los coeficientes del
%       filtro en el vector B de longitud N+1. La frecuencia de
%       corte Wn debe estar entre 0 y 1 con el valor de 1 corres-
%       pondiente a la mitad de la frecuencia de muestreo.
%
%       Si Wn es un vector de dos elementos, Wn = [W1 W2], fir1
%       retorna un filtro pasabanda de orden N con banda de paso:
%       W1 < W < W2.
%
%       [B,A] = fir1 (N,Wn,'high',boxcar(N+1)) diseña un filtro
%       pasa altos.
%       [B,A] = fir1 (N,Wn,'stop',boxcar(N+1)) es un filtro eli-
%       mina banda si Wn = [W1 W2].
%
%       Para filtros pasa altos y elimina banda, N debe ser par.
%
pause % oprima cualquier tecla
%
echo off
delete pds.mat
```

```
fs=input('El valor de la frecuencia de muestreo es (Hz): ');
echo on
%
%
% 1) filtro pasa bajos
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el tipo de filtro requerido: ');
clc
format compact
if tipo==1
eval(pb1)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
N1= input('El orden del filtro es: ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,boxcar(N1+1))
elseif tipo==2
eval(pa1)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'high',boxcar(N1+1))
elseif tipo==3
eval(pban1)
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu] (Hz): ');
N1= input('El orden del filtro es: ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,boxcar(N1+1))
elseif tipo==4
eval(eban1)
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu] (Hz): ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'stop',boxcar(N1+1))
end
format
echo on
pause % oprima cualquier tecla
clc
%
% Ahora calcula la magnitud de respuesta de frecuencia compleja:
%
% espere por favor...
%
echo off
[H,w]=freqz(B,1,512);
plot (w*fs/(2*pi),abs(H)),meta pds, title('Respuesta de frecuencia Rectangular obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
```

```
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia Rectangular obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

DONTRIA.M

```
clc
echo on
%       Diseño de filtro digital usando método de ventanas
%
%       B = fir1 (N,Wn,triang(N+1)) diseña un filtro digital
%       pasabajos FIR de orden N y retorna los coeficientes del
%       filtro en el vector B de longitud N+1. La frecuencia de
%       corte Wn debe estar entre 0 y 1 con el valor de 1 corres-
%       pondiente a la mitad de la frecuencia de muestreo.
%
%       Si Wn es un vector de dos elementos, Wn = [W1 W2], fir1
%       retorna un filtro pasabanda de orden N con banda de paso:
%       W1 < W < W2.
%
%       [B,A] = fir1 (N,Wn,'high',triang(N+1)) diseña un filtro
%       pasa altos.
%       [B,A] = fir1 (N,Wn,'stop',triang(N+1)) es un filtro eli-
%       mina banda si Wn = [W1 W2].
%
%       Para filtros pasa altos y elimina banda, N debe ser par.
%
pause % oprima cualquier tecla
%
echo off
delete pds.mat
fs=input('El valor de la frecuencia de muestreo es (Hz): ');
echo on
%
%
% 1) filtro pasa bajos
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el tipo de filtro requerido: ');
clc
format compact
```

```
if tipo==1
eval(pb1)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
N1= input('El orden del filtro es: ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,triang(N1+1))
elseif tipo==2
eval(pa1)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'high',triang(N1+1))
elseif tipo==3
eval(pban1)
Fc = input ('El valor de la frecuencia de corte es [Fc1 Fcu] (Hz): ');
N1= input('El orden del filtro es: ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,triang(N1+1))
elseif tipo==4
eval(eban1)
Fc = input ('El valor de la frecuencia de corte es [Fc1 Fcu] (Hz): ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'stop',triang(N1+1))
end
fores:
echo on
pause % oprima cualquier tecla
clc
%
% Ahora calcula la magnitud de respuesta de frecuencia compleja:
%
% espere por favor...
%
echo off
[H,w]=freqz(B,1,512);
plot (w*fs/(2*pi),abs(H)),meta pds, title('Respuesta de frecuencia Triangular obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia Triangular obtenida'),..
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

DONCHEW.M

```
clc
echo on
%       Diseño de filtro digital usando método de ventanas
%
%       B = fir1 (N,Wn,chebwin((N+1),R)) diseña un filtro digital
%       pasabajos FIR de orden N y retorna los coeficientes del
%       filtro en el vector B de longitud N+1. La frecuencia de
%       corte Wn debe estar entre 0 y 1 con el valor de 1 corres-
%       pondiente a la mitad de la frecuencia de muestreo.
%
%       Si Wn es un vector de dos elecentos, Wn = [W1 W2], fir1
%       retorna un filtro pasabanda de orden N con banda de paso:
%       W1 < W < W2.
%
%       [B,A] = fir1 (N,Wn,'high',chebwin((N+1),R)) diseña un filtro
%       pasa altos.
%       [B,A] = fir1 (N,Wn,'stop',chebwin((N+1),R)) es un filtro eli-
%       mina banda si Wn = [W1 W2].
%
%       N debe ser par.
%
pause % oprima cualquier tecla
%
echo off
delete pds.mat
fs=input('El valor de la frecuencia de muestreo es (Hz): ');
R=input('El valor de atenuación de rizado es (dB): ');
echo on
%
%
% 1) filtro pasa bajos
% 2) filtro pasa altos
% 3) filtro pasa banda
% 4) filtro elimina banda
%
%
echo off
tipo=input('seleccione el tipo de filtro requerido: ');
clc
format compact
if tipo==1
eval(pb1)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,chebwin((N1+1),R))
elseif tipo==2
eval(pa1)
Fc = input ('El valor de la frecuencia de corte es Fc (Hz): ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc*2/fs;
eval(espere)
B=fir1(N1,Wn1,'high',chebwin((N1+1),R))
elseif tipo==3
```

```
eval(pban1)
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu] (Hz): ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc(2/fs;
eval(espere)
B=fir1(N1,Wn1,chebwin((N1+1),R))
elseif tipo==4
eval(eban1)
Fc = input ('El valor de la frecuencia de corte es [Fcl Fcu] (Hz): ');
N1= input('El orden del filtro es (par): ');
Wn1 = Fc(2/fs;
eval(espere)
B=fir1(N1,Wn1,'stop',chebwin((N1+1),R))
end
format
echo on
pause % oprima cualquier tecla
clc
%
% Ahora calcula la magnitud de respuesta de frecuencia compleja:
%
% espere por favor...
%
echo off
[H,w]=freqz(B,1,512);
plot (w*fs/(2*pi),abs(H)),meta pos, title('Respuesta de frecuencia Chebyshev obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'), pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/(w(512)*fs/(2*pi));
plot(w(1:fz1)*fs/(2*pi),abs(H(1:fz1))),meta, title('Respuesta de frecuencia Chebyshev obtenida'),...
xlabel('Frecuencia (Hz)'), ylabel ('Magnitud'),pause
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

A.2.3.6. METODO DE FASE LINEAL EN EL DOMINIO DE LA FRECUENCIA.

DONFIR2.M

```
%       El programa permite escoger una ventana para el diseño de filtro
%       no recursivo generalizado.
%
        while 1
```

```
        item = ['donhamm1'
                'donhann1'
                'donkais1'
                'donbart1'
                'donblac1'
                'donrect1'
                'dontria1'
                'donchew1'
                'pds     '];
        clc
        help menu7
        n6 = input('            seleccionar tipo de ventana: ');
        if ((n6<=0) | (n6>9))
                break
        end
        item = item (n6,:);
        eval (item)
end
return
clc
```

MENU7.M

```
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
%
```

TIPOS DE VENTANAS PARA FILTROS NO RECURSIVOS
1) HAMMING
2) HANNING
3) KAISER
4) BARTLETT
5) BLACKMAN
6) RECTANGULAR
7) TRIANGULAR
8) CHEBYSHEV
0) retorno

DONHAMM1.M

```
clc
echo on
%
%       Diseño de filtro FIR usando el método de ventanas, especificando
%       la forma del filtro (respuesta de frecuencia).
%
%       B = fir2 (N,F,M) diseña un filtro digital FIR de orden N con la
%       respuesta de frecuencia especificada por vectores F y M, y retorna
%       los coeficientes del filtro en longitud N+1 en el vector B.
```

```
%
%       Los vectores F y M especifican los puntos de corte de la frecuencia
%       y la magnitud  (0-1) para el filtro tal que al graficar M vs. F de-
%       bería mostrarse un gráfico de la respuesta de frecuencia deseada.
%       Las frecuencias en F deben estar entre 0 y la mitad de la frecuen-
%       cia de muestreo, ordenadas en orden ascendente y comenzando con 0
%       y terminando con la frecuencia correspondiente a la mitad de la
%       frecuencia de muestreo fs.
%
%       Mediante este método es posible realizar diseños de filtros multi-
%       banda.
%
%       FIR2 usa una ventana de Hamming si no existe especificación.
%
pause % oprima cualquier tecla
clc
eval(tutor)
echo off
delete pds.met
clc
fs=input('La frecuencia de muestreo es (Hz): ');
N=input('El orden del filtro es: ');
F=input('El vector de frecuencias es (Hz): ');
M=input('El vector de magnitudes es: ');
echo on
% espere por favor...
%
% el gráfico de la respuesta de frecuencia deseada será:
%
pause % oprima cualquier tecla para graficar
echo off
F0=2*F/fs
plot(F,M),meta pds,title('Respuesta de Frecuencia deseada ideal'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'), pause
clc
preg=input('La respuesta de frecuencia deseada está correcta? 1=SI 2=NO : ');
if preg==1
eval(donhamm2)
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
elseif preg==2
end
```

DONHAMM2.M

```
echo on
%
%
% espere por favor...
%
echo off
```

```
B = fir2(N,F0,M);
echo on
%
% El diseño está completo y ahora se puede mirar los coeficientes del filtro:
%
pause % oprima cualquier tecla
echo off
format compact
clc
B
echo on
%
% estos son los coeficientes del filtro
%
% ahora se calcula la respuesta de frecuencia del filtro diseñado y se la
% compara con la del filtro deseado:
%
pause % oprima cualquier tecla
%
% espere por favor...
echo off
format
H=freqz(B,1,512);
Hm=abs(H);
F1=fs/(2*512)*(0:511);
echo on
%
pause % oprima cualquier tecla para graficar
%
echo off
plot (F,M,F1,Hm),meta,title('Respuesta de frecuencia deseada vs. Hamming'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
zoom=input('Desea ampliar la escala del gráfico? 1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/F1(512);
plot (F1(1:fz1),Hm(1:fz1)),meta,title('Respuesta de frecuencia Hamming'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
elseif zoom==2
end
clc
clg
```

DONHANN1.M

```
clc
echo on
%
%       Diseño de filtro FIR usando el método de ventanas, especificando
%       la forma del filtro (respuesta de frecuencia).
%
%       B = fir2 (N,F,M,hanning(N+1)) diseña un filtro digital FIR de orden
%       N con la respuesta de frecuencia especificada por vectores F y M, y
%       retorna los coeficientes del filtro en longitud N+1 en el vector B.
%
```

```
%       Los vectores F y M especifican los puntos de corte de la frecuencia
%       y la magnitud  (0-1) para el filtro tal que al graficar M vs. F de-
%       beria mostrarse un gráfico de la respuesta de frecuencia deseada.
%       Las frecuencias en F deben estar entre 0 y la mitad de la frecuen-
%       cia de muestreo, ordenadas en orden ascendente y comenzando con 0
%       y terminando con la frecuencia correspondiente a la mitad de la
%       frecuencia de muestreo fs.
%
%       Mediante este método es posible realizar diseños de filtros multi-
%       banda.
%
pause % oprima cualquier tecla
clc
eval(tutor)
echo off
delete pds.mat
clc
fs=input('La frecuencia de muestreo es (Hz): ');
N=input('El orden del filtro es: ');
F=input('El vector de frecuencias es (Hz): ');
M=input('El vector de magnitudes es: ');
echo on
% espere por favor...
%
% el gráfico de la respuesta de frecuencia deseada será:
%
pause % oprima cualquier tecla para graficar
echo off
F0=2*F/fs
plot(F,M),meta pds,title('Respuesta de Frecuencia deseada ideal'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'), pause
clc
preg=input('La respuesta de frecuencia deseada está correcta? 1=SI 2=NO : ');
if preg==1
eval(donhann2)
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
elseif preg==2
end
```

DONHANN2.M

```
echo on
%
%
% espere por favor...
%
echo off
B = fir2(N,F0,M,hanning(N+1));
echo on
%
```

```
% El diseño está completo y ahora se puede mirar los coeficientes del filtro:
%
pause % oprima cualquier tecla
echo off
format compact
clc
B
echo on
%
% estos son los coeficientes del filtro
%
% ahora se calcula la respuesta de frecuencia del filtro diseñado y se la
% compara con la del filtro deseado:
%
pause % oprima cualquier tecla
%
% espere por favor...
echo off
format
H=freqz(B,1,512);
Hm=abs(H);
F1=fs/(2*512)*(0:511);
echo on
%
pause % oprima cualquier tecla para graficar
%
echo off
plot (F,M,F1,Hm),meta,title('Respuesta de frecuencia deseada vs. Hanning'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
zoom=input('Desea ampliar la escala del gráfico? 1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/F1(512);
plot (F1(1:fz1),Hm(1:fz1)),meta,title('Respuesta de frecuencia Hanning'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
elseif zoom==2
end
clc
clg
```

DONKAIS1.M

```
clc
echo on
%
%       Diseño de filtro FIR usando el método de ventanas, especificando
%       la forma del filtro (respuesta de frecuencia).
%
%       B = fir2 (N,F,M,kaiser((N+1),beta)) diseña un filtro digital FIR de
%       orden N con la respuesta de frecuencia especificada por vectores F
%       y M, y retorna los coeficientes del filtro en longitud N+1 en el
%       vector B.
%
%       Los vectores F y M especifican los puntos de corte de la frecuencia
%       y la magnitud  (0-1) para el filtro tal que al graficar M vs. F de-
```

```
%       bería mostrarse un gráfico de la respuesta de frecuencia deseada.
%       Las frecuencias en F deben estar entre 0 y la mitad de la frecuen-
%       cia de muestreo, ordenadas en orden ascendente y comenzando con 0
%       y terminando con la frecuencia correspondiente a la mitad de la
%       frecuencia de muestreo fs.
%
%       Mediante este método es posible realizar diseños de filtros multi-
%       banda.
%
pause % oprima cualquier tecla
clc
eval(tutor)
echo off
delete pds.mat
clc
fs=input('La frecuencia de muestreo es (Hz): ');
N=input('El orden del filtro es: ');
F=input('El vector de frecuencias es (Hz): ');
M=input('El vector de magnitudes es: ');
beta=input('El valor de beta es: ');
echo on
% espere por favor...
%
% el gráfico de la respuesta de frecuencia deseada será:
%
pause % oprima cualquier tecla para graficar
echo off
F0=2*F/fs
plot(F,M),meta pds,title('Respuesta de Frecuencia deseada ideal'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'), pause
clc
preg=input('La respuesta de frecuencia deseada está correcta? 1=SI 2=NO : ');
if preg==1
eval(donkais2)
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
elseif preg==2
end
clc
clg
```

DONKAIS2.M

```
echo on
%
%
% espere por favor...
%
echo off
B = fir2(N,F0,M,kaiser((N+1),beta));
echo on
```

```
%
% El diseño está completo y ahora se puede mirar los coeficientes del filtro:
%
pause % oprima cualquier tecla
echo off
format compact
clc
B
echo on
%
% estos son los coeficientes del filtro
%
% ahora se calcula la respuesta de frecuencia del filtro diseñado y se la
% compara con la del filtro deseado:
%
pause % oprima cualquier tecla
%
% espere por favor...
echo off
format
H=freqz(B,1,512);
Hm=abs(H);
F1=fs/(2*512)*(0:511);
echo on
%
pause % oprima cualquier tecla para graficar
%
echo off
plot (F,M,F1,Hm),meta,title('Respuesta de frecuencia deseada vs. Kaiser'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/F1(512);
plot (F1(1:fz1),Hm(1:fz1)),meta,title('Respuesta de frecuencia Kaiser'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
elseif zoom==2
end
clc
clg
```

DONBART1.M

```
clc
echo on
%
%       Diseño de filtro FIR usando el método de ventanas, especificando
%       la forma del filtro (respuesta de frecuencia).
%
%       B = fir2 (N,F,M,bartlett(N+1)) diseña un filtro digital FIR de
%       orden N con la respuesta de frecuencia especificada por vectores F
%       y M, y retorna los coeficientes del filtro en longitud N+1 en el
%       vector B.
%
%       Los vectores F y M especifican los puntos de corte de la frecuencia
```

```
%       y la magnitud  (0-1) para el filtro tal que al graficar M vs. F de-
%       bería mostrarse un gráfico de la respuesta de frecuencia deseada.
%       Las frecuencias en F deben estar entre 0 y la mitad de la frecuen-
%       cia de muestreo, ordenadas en orden ascendente y comenzando con 0
%       y terminando con la frecuencia correspondiente a la mitad de la
%       frecuencia de muestreo fs.
%
%       Mediante este método es posible realizar diseños de filtros multi-
%       banda.
%
pause % oprima cualquier tecla
clc
eval(tutor)
echo off
delete pds.met
clc
fs=input('La frecuencia de muestreo es (Hz): ');
N=input('El orden del filtro es: ');
F=input('El vector de frecuencias es (Hz): ');
M=input('El vector de magnitudes es: ');
echo on
% espere por favor...
%
% el gráfico de la respuesta de frecuencia deseada será:
%
pause % oprima cualquier tecla para graficar
echo off
F0=2*F/fs
plot(F,M),meta pds,title('Respuesta de Frecuencia deseada ideal'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'), pause
clc
preg=input('La respuesta de frecuencia deseada está correcta? 1=SI 2=NO : ');
if preg==1
eval(donbart2)
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
elseif preg==2
end
clc
clg
```

DONBART2.M

```
echo on
%
%
% espere por favor...
%
echo off
B = fir2(N,F0,M,bartlett(N+1));
echo on
```

```
%
% El diseño está completo y ahora se puede mirar los coeficientes del filtro:
%
pause % oprima cualquier tecla
echo off
format compact
clc
B
echo on
%
% estos son los coeficientes del filtro
%
% ahora se calcula la respuesta de frecuencia del filtro diseñado y se la
% compara con la del filtro deseado:
%
pause % oprima cualquier tecla
%
% espere por favor...
echo off
format
H=freqz(B,1,512);
Ha=abs(H);
F1=fs/(2*512)*(0:511);
echo on
%
pause % oprima cualquier tecla para graficar
%
echo off
plot (F,M,F1,Ha),meta,title('Respuesta de frecuencia deseada vs. Bartlett'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/F1(512);
plot (F1(1:fz1),Ha(1:fz1)),meta,title('Respuesta de frecuencia Bartlett'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
elseif zoom==2
end
clc
clg
```

DONBLAC1.M

```
clc
echo on
%
%       Diseño de filtro FIR usando el método de ventanas, especificando
%       la forma del filtro (respuesta de frecuencia).
%
%       B = fir2 (N,F,M,blackman(N+1)) diseña un filtro digital FIR de
%       orden N con la respuesta de frecuencia especificada por vectores F
%       y M, y retorna los coeficientes del filtro en longitud N+1 en el
%       vector B.
%
%       Los vectores F y M especifican los puntos de corte de la frecuencia
```

```
%       y la magnitud  (0-1) para el filtro tal que al graficar M vs. F de-
%       beria mostrarse un gráfico de la respuesta de frecuencia deseada.
%       Las frecuencias en F deben estar entre 0 y la mitad de la frecuen-
%       cia de muestreo, ordenadas en orden ascendente y comenzando con 0
%       y terminando con la frecuencia correspondiente a la mitad de la
%       frecuencia de muestreo fs.
%
%       Mediante este método es posible realizar diseños de filtros multi-
%       banda.
%
pause % oprima cualquier tecla
clc
eval(tutor)
echo off
delete pds.mat
clc
fs=input('La frecuencia de muestreo es (Hz): ');
N=input('El orden del filtro es: ');
F=input('El vector de frecuencias es (Hz): ');
M=input('El vector de magnitudes es: ');
echo on
% espere por favor...
%
% el gráfico de la respuesta de frecuencia deseada será:
%
pause % oprima cualquier tecla para graficar
echo off
F0=2*F/fs
plot(F,M),meta pds,title('Respuesta de Frecuencia deseada ideal'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'), pause
clc
preg=input('La respuesta de frecuencia deseada está correcta? 1=SI 2=NO : ');
if preg==1
eval(donblac2)
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
elseif preg==2
end
```

DONBLAC2.M

```
echo on
%
%
% espere por favor...
%
echo off
B = fir2(N,F0,M,blackman(N+1));
echo on
%
% El diseño está completo y ahora se puede mirar los coeficientes del filtro:
```

```
%
pause % oprima cualquier tecla
echo off
format compact
clc
B
echo on
%
% estos son los coeficientes del filtro
%
% ahora se calcula la respuesta de frecuencia del filtro diseñado y se la
% compara con la del filtro deseado:
%
pause % oprima cualquier tecla
%
% espere por favor...
echo off
format
H=freqz(B,1,512);
Hm=abs(H);
F1=fs/(2*512)*(0:511);
echo on
%
pause % oprima cualquier tecla para graficar
%
echo off
plot (F,M,F1,Hm),meta,title('Respuesta de frecuencia deseada vs. Blackman'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/F1(512);
plot (F(1:fz1),Hm(1:fz1)),meta,title('Respuesta de frecuencia Blackman'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
elseif zoom==2
end
clc
clg
```

DONRECT1.M

```
clc
echo on
%
%       Diseño de filtro FIR usando el método de ventanas, especificando
%       la forma del filtro (respuesta de frecuencia).
%
%       B = fir2 (N,F,M,boxcar(N+1)) diseña un filtro digital FIR de
%       orden N con la respuesta de frecuencia especificada por vectores F
%       y M, y retorna los coeficientes del filtro en longitud N+1 en el
%       vector B.
%
%       Los vectores F y M especifican los puntos de corte de la frecuencia
%       y la magnitud  (0-1) para el filtro tal que al graficar M vs. F de-
%       bería mostrarse un gráfico de la respuesta de frecuencia deseada.
```

```
%       Las frecuencias en F deben estar entre 0 y la mitad de la frecuen-
%       cia de muestreo, ordenadas en orden ascendente y comenzando con 0
%       y terminando con la frecuencia correspondiente a la mitad de la
%       frecuencia de muestreo fs.
%
%       Mediante este método es posible realizar diseños de filtros multi-
%       banda.
%
pause % oprima cualquier tecla
clc
eval(tutor)
echo off
delete pds.met
clc
fs=input('La frecuencia de muestreo es (Hz): ');
N=input('El orden del filtro es: ');
F=input('El vector de frecuencias es (Hz): ');
M=input('El vector de magnitudes es: ');
echo on
% espere por favor...
%
% el gráfico de la respuesta de frecuencia deseada será:
%
pause % oprima cualquier tecla para graficar
echo off
F0=2*F/fs
plot(F,M),meta pds,title('Respuesta de Frecuencia deseada ideal'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'), pause
clc
preg=input('La respuesta de frecuencia deseada está correcta? 1=SI 2=NO : ');
if preg==1
eval(donrect2)
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
elseif preg==2
end
```

DONRECT2.M

```
echo on
%
%
% espere por favor...
%
echo off
B = fir2(N,F0,M,boxcar(N+1));
echo on
%
% El diseño está completo y ahora se puede mirar los coeficientes del filtro:
%
pause % oprima cualquier tecla
```

```
echo off
format compact
clc
B
echo on
%
% estos son los coeficientes del filtro
%
% ahora se calcula la respuesta de frecuencia del filtro diseñado y se la
% compara con la del filtro deseado:
%
pause % oprima cualquier tecla
%
% espere por favor...
echo off
format
H=freqz(B,1,512);
Hm=abs(H);
F1=fs/(2*512)*(0:511);
echo on
%
pause % oprima cualquier tecla para graficar
%
echo off
plot (F,M,F1,Hm),meta,title('Respuesta de frecuencia deseada vs. Rectangular'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/F1(512);
plot (F1(1:fz1),Hm(1:fz1)),meta,title('Respuesta de frecuencia Rectangular'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
elseif zoom==2
end
clc
clg
```

DONTRIA1.M

```
clc
echo on
%
%       Diseño de filtro FIR usando el método de ventanas, especificando
%       la forma del filtro (respuesta de frecuencia).
%
%       B = fir2 (N,F,M,triang(N+1)) diseña un filtro digital FIR de
%       orden N con la respuesta de frecuencia especificada por vectores F
%       y M, y retorna los coeficientes del filtro en longitud N+1 en el
%       vector B.
%
%       Los vectores F y M especifican los puntos de corte de la frecuencia
%       y la magnitud  (0-1) para el filtro tal que al graficar M vs. F de-
%       beria mostrarse un gráfico de la respuesta de frecuencia deseada.
%       Las frecuencias en F deben estar entre 0 y la mitad de la frecuen-
%       cia de muestreo, ordenadas en orden ascendente y comenzando con 0
```

```
%       y terminando con la frecuencia correspondiente a la mitad de la
%       frecuencia de muestreo fs.
%
%       Mediante este método es posible realizar diseños de filtros multi-
%       banda.
%
pause % oprima cualquier tecla
clc
eval(tutor)
echo off
delete pds.met
clc
fs=input('La frecuencia de muestreo es fs (Hz): ');
N=input('El orden del filtro es: ');
F=input('El vector de frecuencias es F (Hz): ');
M=input('El vector de magnitudes es M: ');
echo on
% espere por favor...
%
% el gráfico de la respuesta de frecuencia deseada será:
%
pause % oprima cualquier tecla para graficar
echo off
F0=2*F/fs
plot(F,M),meta pds,title('Respuesta de Frecuencia deseada ideal'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'), pause
clc
preg=input('La respuesta de frecuencia deseada está correcta? 1=SI 2=NO : ');
if preg==1
eval(dontria2)
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
elseif preg==2
end
```

DONTRIA2.M

```
echo on
%
%
% espere por favor...
%
echo off
B = fir2(N,F0,M,triang(N+1));
echo on
%
% El diseño está completo y ahora se puede mirar los coeficientes del filtro:
%
pause % oprima cualquier tecla
echo off
format compact
```

```
clc
B
echo on
%
% estos son los coeficientes del filtro
%
% ahora se calcula la respuesta de frecuencia del filtro diseñado y se la
% compara con la del filtro deseado:
%
pause % oprima cualquier tecla
%
% espere por favor...
echo off
format
H=freqz(B,1,512);
Hm=abs(H);
F1=fs/(2*512)*(0:511);
echo on
%
pause % oprima cualquier tecla para graficar
%
echo off
plot (F,M,F1,Hm),meta,title('Respuesta de frecuencia deseada vs. Triangular'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
zoom=input('Desea ampliar la escala del gráfico?  1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/F1(512);
plot (F1(1:fz1),Hm(1:fz1)),meta,title('Respuesta de frecuencia Triangular'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
elseif zoom==2
end
clc
clg
```

DONCHEW1.M

```
clc
echo on
%
%       Diseño de filtro FIR usando el método de ventanas, especificando
%       la forma del filtro (respuesta de frecuencia).
%
%       B = fir2 (N,F,M,chebwin((N+1),R)) diseña un filtro digital FIR de
%       orden N con la respuesta de frecuencia especificada por vectores F
%       y M, y retorna los coeficientes del filtro en longitud N+1 en el
%       vector B.
%
%       Los vectores F y M especifican los puntos de corte de la frecuencia
%       y la magnitud  (0-1) para el filtro tal que al graficar M vs. F de-
%       bería mostrarse un gráfico de la respuesta de frecuencia deseada.
%       Las frecuencias en F deben estar entre 0 y la mitad de la frecuen-
%       cia de muestreo, ordenadas en orden ascendente y comenzando con 0
%       y terminando con la frecuencia correspondiente a la mitad de la
%       frecuencia de muestreo fs.
```

```
%
%       Mediante este método es posible realizar diseños de filtros multi-
%       banda.
%
%       N debe ser par.
%
pause % oprima cualquier tecla
clc
eval(tutor)
echo off
delete pds.met
clc
fs=input('La frecuencia de muestreo es (Hz): ');
N=input('El orden del filtro es: ');
F=input('El vector de frecuencias es (Hz): ');
M=input('El vector de magnitudes es: ');
R=input('El valor de atenuación del rizado es: ');
echo on
% espere por favor...
%
% el gráfico de la respuesta de frecuencia deseada será:
%
pause % oprima cualquier tecla para graficar
echo off
F0=2*F/fs
plot(F,M),meta pds,title('Respuesta de Frecuencia deseada ideal'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'), pause
clc
preg=input('La respuesta de frecuencia deseada está correcta? 1=SI 2=NO : ');
if preg==1
eval(donchew2)
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
elseif preg==2
end
```

DONCHEW2.M

```
echo on
%
%
% espere por favor...
%
echo off
B = fir2(N,F0,M,chebwin((N+1),R));
echo on
%
% El diseño está completo y ahora se puede mirar los coeficientes del filtro:
%
pause % oprima cualquier tecla
echo off
```

```
format compact
clc
B
echo on
%
% estos son los coeficientes del filtro
%
% ahora se calcula la respuesta de frecuencia del filtro diseñado y se la
% compara con la del filtro deseado:
%
pause % oprima cualquier tecla
%
% espere por favor...
echo off
format
H=freqz(B,1,512);
Ha=abs(H);
F1=fs/(2*512)*(0:511);
echo on
%
pause % oprima cualquier tecla para graficar
%
echo off
plot (F,M,F1,Ha),meta,title('Respuesta de frecuencia deseada vs. Chebyshev'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
zoom=input('Desea ampliar la escala del gráfico? 1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/F1(512);
plot (F1(1:fz1),Ha(1:fz1)),meta,title('Respuesta de frecuencia Chebyshev'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
elseif zoom==2
end
clc
clg
```

A.2.3.7. METODO DE DISEÑO DE PARKS-McCLELLAN.

DONREMEZ.M

```
clc
echo on
%
%       Diseño de filtro digital FIR equirizado óptimo mediante el algorit-
%       mo de Parks-McClellan.
%
%       B=remez(N,F,M) diseña un filtro digital FIR de orden N con la res-
%       puesta de frecuencia especificada por los vectores F y M, y retor-
%       na los coeficientes del filtro en longitud N+1 en el vector B.
%
%       Los vectores F y M especifican los puntos de corte de la frecuencia
%       y la magnitud  (0-1) para el filtro tal que al graficar M vs. F de-
```

```
%       bería mostrarse un gráfico de la respuesta de frecuencia deseada.
%       Las frecuencias en F deben estar entre 0 y la mitad de la frecuen-
%       cia de muestreo, ordenadas en orden ascendente y comenzando con 0
%       y terminando con la frecuencia correspondiente a la mitad de la
%       frecuencia de muestreo fs.
%
%       Mediante este método es posible realizar diseños de filtros multi-
%       banda.
%
pause % oprima cualquier tecla
clc
eval(tutor)
echo off
delete pds.met
clc
fs=input('La frecuencia de muestreo es (Hz): ');
N=input('El orden del filtro es: ');
F=input('El vector de frecuencias es (Hz): ');
M=input('El vector de magnitudes es: ');
echo on
% espere por favor...
%
% el gráfico de la respuesta de frecuencia deseada será:
%
pause % oprima cualquier tecla para graficar
echo off
F0=2*F/fs
plot(F,M),meta pds,title('Respuesta de Frecuencia deseada'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'), pause
clc
preg=input('La respuesta de frecuencia deseada está correcta? 1=SI 2=NO : ');
if preg==1
eval(donreme1)
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
elseif preg==2
end
clc
clg
```

DONREME1.M

```
echo on
%
%
% espere por favor...
%
echo off
B = remez(N,F0,M);
echo on
%
```

```
% El diseño está completo y ahora se puede mirar los coeficientes del filtro:
%
pause % oprima cualquier tecla
echo off
format compact
clc
B
echo on
%
% estos son los coeficientes del filtro
%
% ahora se calcula la respuesta de frecuencia del filtro diseñado y se la
% compara con la del filtro deseado:
%
pause % oprima cualquier tecla
%
% espere por favor...
echo off
format
H=freqz(B,1,512);
Hm=abs(H);
F1=fs/(2*512)*(0:511);
echo on
%
pause % oprima cualquier tecla para graficar
%
echo off
plot (F,M,F1,Hm),meta,title('Respuesta de frecuencia deseada vs. diseñada'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
zoom=input('Desea ampliar la escala del gráfico? 1=SI, 2=NO: ');
if zoom==1
fz=input('El nuevo valor de fondo de escala en el eje de frecuencia es: ');
fz1=fz*512/F1(512);
plot (F1(1:fz1),Hm(1:fz1)),meta,title('Respuesta de frecuencia Parks-McClellan'),...
xlabel('Frecuencia (Hz)'),ylabel('Magnitud'),pause
elseif zoom==2
end
clc
clg
```

A.2.4. METODOS DE FILTRADO DE LAS SEÑALES.

FILTRADO.M

```
clc
echo on
%
% Este programa posibilita el escoger una opción del menú
% de selección de tipo de filtro diseñado para el filtrado:
%
echo off
while 1
       item=['iir0'
```

```
            'fir '];
        clc
        help menu5
        n4=input('          Seleccione el tipo de filtrado: ');
        if ((n4<=0) | (n4>2))
            break
        end
        item=item(n4,:);
        eval(item)
end
return
clc
```

MENU5.M

```
%
%
%           ***** METODOS DE FILTRADO DE SEÑALES *****
%
%           1) FILTRADO MEDIANTE DISEÑO DE FILTROS RECURSIVOS
%           2) FILTRADO MEDIANTE DISEÑO DE FILTROS NO RECURSIVOS
%
%           0) retorno
%
%
%
%
```

IIRO.M

```
clc
echo on
%
% Este programa posibilita el escoger una opción del menú
% de selección de tipo de filtro diseñado para el filtrado,
% cuando se ha realizado el diseño de filtros recursivos:
%
echo off
while 1
        item=['iir1'
            'iir2'];
        clc
        help menu8
        n7=input('          Seleccione el tipo de filtrado: ');
        if ((n7<=0) | (n7>2))
            break
        end
        item=item(n7,:);
        eval(item)
end
return
clc
```

MENU8.M

```
%
%
%
%          ++++ FILTRADO CON FILTROS RECURSIVOS ++++
%
%          1) FILTRADO DE FASE CERO
%          2) FILTRADO EN FORMA DIRECTA TRANSPUESTA
%
%          0) retorno
%
%
%
%
```

IIR1.M

```
clc
echo on
%
%       Este programa realiza el filtrado de la señal contaminada
%       cuando se ha realizado el diseño de un filtro recursivo pa-
%       ra el objeto, mediante el método de fase cero en adelanto
%       y retardo. El filtro está descrito por la ecuación de dife-
%       rencias:
%
%       y(n) = b(1)*x(n) + b(2)*x(n-1) + ... + b(nb+1)*x(n-nb)
%                        - a(2)*y(n-1) - ... - a(na+1)*y(n-na)
%
%       Después de realizar el filtrado hacia adelante, la secuen-
%       cia de filtrado es revertida y regresa a través del filtro.
%
%       La secuencia resultante tiene precisamente distorsión de
%       fase cero y doble el orden del filtro.
%
echo off
delete pds.mat
escoge=input('Desea utilizar la señal obtenida de la planta?  1=SI, 2=NO  ');
if escoge==2
fs=input('La frecuencia de muestreo de los datos es (Hz): ');
t=0:1/fs:999*1/fs;
y3=input('Por favor defina la señal u(t) que se quiere analizar:                    ');
u=y3;
elseif escoge==1
%t=0:0.01:9.99;
u=y2;
end
clc
echo on
%
% espere por favor...
%
echo off
```

```
F0 = filtfilt(B,A,u);
%
echo on
%
pause % oprima cualquier tecla para graficar:
%
clc
echo off
if escoge==2
subplot(211), plot(t,y3), title('señal contaminada'),xlabel('tiempo(seg)');
subplot(212), plot(t,F0), title('señal filtrada F0(t)'),xlabel('tiempo(seg)');
pause
elseif escoge==1
subplot(211), plot(t,y2), title('señal contaminada'),xlabel('tiempo(seg)');
subplot(212), plot(t,F0), title('señal filtrada F0(t)'),xlabel('tiempo(seg)');
pause
end
clc
clg
zoom=input('Desea modificar el fondo de escala del gráfico?: 1=SI, 2=NO ');
if zoom==1
pun=input('Con el fin de visualizar mejor el gráfico entre un valor entre .1 y 10: ');
if escoge==1
subplot(211),plot(t(1:pun*100),y2(1:pun*100)),title('señal contaminada'),xlabel('tiempo(seg)');
subplot(212),plot(t(1:pun*100),F0(1:pun*100)),title('señal                                    filtrada
F0(t)'),xlabel('tiempo(seg)'),pause;
elseif escoge==2
subplot(211),plot(t(1:pun*100),y3(1:pun*100)),title('señal contaminada'),xlabel('tiempo(seg)');
subplot(212),plot(t(1:pun*100),F0(1:pun*100)),title('señal                                    filtrada
F0(t)'),xlabel('tiempo(seg)'),pause;
end
clg
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
if zoom==1
if escoge==1
plot(t(1:pun*100),y2(1:pun*100)),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t(1:pun*100),F0(1:pun*100)),meta, title('señal filtrada F0(t)'),xlabel('tiempo(seg)'),pause;
elseif escoge==2
plot(t(1:pun*100),y3(1:pun*100)),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t(1:pun*100),F0(1:pun*100)),meta, title('señal filtrada F0(t)'),xlabel('tiempo(seg)'),pause;
end
clg
elseif zoom==2
if escoge==1
plot(t,y2),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t,F0),meta, title('señal filtrada F0(t)'),xlabel('tiempo(seg)'),pause;
elseif escoge==2
plot(t,y3),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t,F0),meta, title('señal filtrada F0(t)'),xlabel('tiempo(seg)'),pause;
end
end
eval(imprimir)
elseif imprime==2
end
```

```
espec=input('Desea mirar el nuevo espectro de frecuencia?  1=SI, 2=NO:  ');
if espec==1
%u=F0;
clg
eval(espectro)
elseif espec==2
end
clg
clc
```

IIR2.M

```
clc
echo on
%
%       Este programa realiza el filtrado de la señal contacinada
%       cuando se ha realizado el diseño de un filtro recursivo pa-
%       ra el objeto, mediante la implementación del método de for-
%       ma directa transpuesta de la ecuación de diferencias:
%
%       y(n) = b(1)*x(n) + b(2)*x(n-1) + ... + b(nb+1)*x(n-nb)
%                        - a(2)*y(n-1) - ... - a(na+1)*y(n-na)
%
%
echo off
delete pds.mat
escoge=input('Desea utilizar la señal obtenida de la planta?  1=SI, 2=NO  ');
if escoge==2
fs=input('La frecuencia de muestreo de los datos es (Hz): ');
t=0:1/fs:999*1/fs;
y3=input('Por favor defina la señal u(t) que se quiere analizar:                    ');
u=y3;
elseif escoge==1
%t=0:0.01:9.99;
u=y2;
end
echo on
%
% espere por favor...
%
echo off
F1 = filter(B,A,u);
%
echo on
%
pause % oprima cualquier tecla para graficar:
%
clc
echo off
if escoge==2
subplot(211), plot(t,y3), title('señal contaminada'),xlabel('tiempo(seg)');
subplot(212), plot(t,F1), title('señal filtrada F1(t)'),xlabel('tiempo(seg)');
pause
elseif escoge==1
subplot(211), plot(t,y2), title('señal contaminada'),xlabel('tiempo(seg)');
```

```
subplot(212), plot(t,F1), title('señal filtrada F1(t)'),xlabel('tiempo(seg)');
pause
end
clc
clg
zoom=input('Desea modificar la escala del gráfico?: 1=SI, 2=NO ');
if zoom==1
pun=input('Con el fin de visualizar mejor el gráfico entre un valor entre .1 y 10: ');
if escoge==1
subplot(211),plot(t(1:pun*100),y2(1:pun*100)),title('señal contaminada'),xlabel('tiempo(seg)');
subplot(212),plot(t(1:pun*100),F1(1:pun*100)),title('señal                                    filtrada
F1(t)'),xlabel('tiempo(seg)'),pause;
elseif escoge==2
subplot(211),plot(t(1:pun*100),y3(1:pun*100)),title('señal contaminada'),xlabel('tiempo(seg)');
subplot(212),plot(t(1:pun*100),F1(1:pun*100)),title('señal                                    filtrada
F1(t)'),xlabel('tiempo(seg)'),pause;
end
clg
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
if zoom==1
if escoge==1
plot(t(1:pun*100),y2(1:pun*100)),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t(1:pun*100),F1(1:pun*100)),meta, title('señal filtrada F1(t)'),xlabel('tiempo(seg)'),pause;
elseif escoge==2
plot(t(1:pun*100),y3(1:pun*100)),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t(1:pun*100),F1(1:pun*100)),meta, title('señal filtrada F1(t)'),xlabel('tiempo(seg)'),pause;
end
clg
elseif zoom==2
if escoge==1
plot(t,y2),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t,F1),meta, title('señal filtrada F1(t)'),xlabel('tiempo(seg)'),pause;
elseif escoge==2
plot(t,y3),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t,F1),meta, title('señal filtrada F1(t)'),xlabel('tiempo(seg)'),pause;
end
end
eval(imprimir)
elseif imprime==2
end
espec=input('Desea mirar el nuevo espectro de frecuencia?  1=SI, 2=NO:  ');
if espec==1
%u=F1;
clg
eval(espectro)
elseif espec==2
end
clg
clc
```

FIR.M

```
clc
echo on
%
% Este programa realiza el filtrado de la señal contaminada cuando se ha
% realizado el diseño de un filtro no recursivo para el objeto, mediante
% el uso de la TRF (FFT).
%
echo off
delete pds.met
escoge=input('Desea utilizar la señal obtenida de la planta?  1=SI, 2=NO  ');
if escoge==2
fs=input('La frecuencia de muestreo de los datos es (Hz): ');
t=0:1/fs:999*1/fs;
y3=input('Por favor defina la señal u(t) que se quiere analizar:                    ');
u=y3;
elseif escoge==1
%t=0:0.01:9.99;
u=y2;
end
echo on
%
pause % oprima cualquier tecla
%
% espere por favor
%
echo off
F2 = fftfilt(B,u,256);
echo on
%
pause % oprima cualquier tecla para graficar
%
echo off
clc
if escoge==2
subplot(211), plot(t,y3), title('señal contaminada'),xlabel('tiempo(seg)');
subplot(212), plot(t,F2), title('señal filtrada F2(t)'),xlabel('tiempo(seg)');
pause
elseif escoge==1
subplot(211), plot(t,y2), title('señal contaminada'),xlabel('tiempo(seg)');
subplot(212), plot(t,F2), title('señal filtrada F2(t)'),xlabel('tiempo(seg)');
pause
end
clg
clc
zoom=input('Desea modificar el fondo de escala del gráfico?: 1=SI, 2=NO ');
if zoom==1
pun=input('Con el fin de visualizar mejor el gráfico entre un valor entre .1 y 10: ');
if escoge==1
subplot(211),plot(t(1:pun*100),y2(1:pun*100)),title('señal contaminada'),xlabel('tiempo(seg)');
subplot(212),plot(t(1:pun*100),F2(1:pun*100)),title('señal                                    filtrada
F2(t)'),xlabel('tiempo(seg)'),pause;
elseif escoge==2
subplot(211),plot(t(1:pun*100),y3(1:pun*100)),title('señal contaminada'),xlabel('tiempo(seg)');
subplot(212),plot(t(1:pun*100),F2(1:pun*100)),title('señal                                    filtrada
```

```
F2(t)'),xlabel('tiempo(seg)'),pause;
end
clg
elseif zoom==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
if zoom==1
if escoge==1
plot(t(1:pun*100),y2(1:pun*100)),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t(1:pun*100),F2(1:pun*100)),meta, title('señal filtrada F2(t)'),xlabel('tiempo(seg)'),pause;
elseif escoge==2
plot(t(1:pun*100),y3(1:pun*100)),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t(1:pun*100),F2(1:pun*100)),meta, title('señal filtrada F2(t)'),xlabel('tiempo(seg)'),pause;
end
clg
elseif zoom==2
if escoge==1
plot(t,y2),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t,F2),meta, title('señal filtrada F2(t)'),xlabel('tiempo(seg)'),pause;
elseif escoge==2
plot(t,y3),meta pds, title('señal contaminada'),xlabel('tiempo(seg)'),pause;
plot(t,F2),meta, title('señal filtrada F2(t)'),xlabel('tiempo(seg)'),pause;
end
end
eval(imprimir)
elseif imprime==2
end
espec=input('Desea mirar el nuevo espectro de frecuencia?  1=SI, 2=NO:  ');
if espec==1
%u=F2;
clg
eval(espectro)
elseif espec==2
end
clg
clc
```

A.2.5. AJUSTE POLINOMIAL DE SEÑALES DIGITALES.

MODELO.M

```
clc
echo on
%
%
%       Este programa realiza ajuste polinomial de señales digitales.
%
%       Primeramente se debe introducir los N valores de k (tiempos en
%       los cuales se tienen las muestras [k0,k1,k2,...,k(N-1)]), se-
%       guido de los valores de x(k), es decir, el valor de las muestras
%       a t=k, [x(k0),x(k1),x(k2),...x(k(N-1))].
```

```
%
%       De esta manera, se obtiene un modelo polinómico de la señal di-
%       gital de interés, con el cual, se puede realizar las aplicacio-
%       nes de interpolación y extrapolación de los datos discretos que
%       se posee, si se desea.
%
%       El método utilizado para la modelación polinomial, es el de e-
%       rrores mínimos cuadrados. Si la distribución de los datos no
%       corresponde a una curva suavizada, los resultados de la inter-
%       polación y/o de la extrapolación pueden no ser confiables.
%
%
echo off
delete pds.met
xk=input('Los valores de k, en los cuales se ha obtenido las muestras son: ');
yk=input('Los valores de los datos x(k) son: (también puede ser f(xk)): ');
plot(xk,yk,'o'),meta pds,title('SEÑAL DISCRETA DE DATOS OBTENIDOS'),xlabel('valores de k'),...
ylabel('valores de x(k)'),pause
clc
Nk=input('El orden deseado del modelo polinómico es: ');
echo on
% Ahora se calcula el polinomio:...
%
echo off
ck=polyfit(xk,yk,Nk)
echo on
% Los anteriores coeficientes corresponden al modelo (potencias descendentes)
%
pause % oprima cualquier tecla para graficar el modelo evaluado en k puntos
echo off
ajuste=polyval(ck,xk);
plot(xk,ajuste,xk,yk,'o'),meta,title('SEÑAL DISCRETA "o" Y MODELO'),xlabel('valores de k'),...
ylabel('valores de x(k)'),pause
clc
format compact
echo on
% Los datos x(k) son los siguientes:
%
%
echo off
yk
echo on
%
% Los valores hallados x'(k), evaluando el modelo son:
%
%
echo off
ajuste
echo on
%
pause % oprima cualquier tecla
clc
echo off
aplic=input('Desea realizar interpolación o extrapolación? 1=SI, 2=NO: ');
if aplic==1
aplic1=input('1 = interpolación,   2 = extrapolación:  ');
xk1=input('El (los) valor(es) de k donde se quiere conocer x(k): ');
ajuste1=polyval(ck,xk1);
```

```
yk2=ajuste1;
eval(extra);
echo off
format
if aplic1==1
plot(xk,yk,'o',xk,ajuste,xk1,yk2,'*'),meta,title('SEÑAL DISCRETA "o" Y MODELO'),...
xlabel('valores de k'),ylabel('valores de x(k)'),pause
elseif aplic1==2
plot(xk,yk,'o',xk,ajuste,xk1,yk2,'*',xk1,ajuste1),meta,title('SEÑAL DISCRETA "o" Y MODELO'),...
xlabel('valores de k'),ylabel('valores de x(k)'),pause
end
elseif aplic==2
end
clc
imprime=input('Desea imprimir los anteriores gráficos?  1=SI, 2=NO :  ');
if imprime==1
eval(imprimir)
elseif imprime==2
end
clc
clg
```

A.2.6. SEÑAL BINARIA PSEUDO ALEATORIA (PRBS).

PRBS.M

```
clc
echo on
%
%       Este programa permite la generación de la señal binaria pseudo
%       aleatoria (PRBS), con la posibilidad de generar un modelo fijo,
%       o un modelo definido por el usuario.
%
%       NOTA: la generación de la señal PRBS se realiza únicamente con
%       una frecuencia de muestreo de 100 Hz.
%
%
echo off
valor=input('Desea introducir los datos binarios para la PRBS?  1=SI, 2=NO: ');
if valor==1
x(1)=input('El valor del bit 1 es: ');
x(2)=input('El valor del bit 2 es: ');
x(3)=input('El valor del bit 3 es: ');
x(4)=input('El valor del bit 4 es: ');
x(5)=input('El valor del bit 5 es: ');
x(6)=input('El valor del bit 6 es: ');
elseif valor==2
x(1)=0;
x(2)=1;
x(3)=1;
x(4)=0;
x(5)=0;
x(6)=1;
```

```
end
cont=0;
z=0;
echo on
%
% espere por favor...
%
echo off
fs=100;
for ts=0:0.1:9.9
%fs=input('El valor de la frecuencia de muestreo es (Hz):  ');
if (x(5)==1 & x(6)==1) | (x(5)==0 & x(6)==0)
est=-1;
x(6)=x(5);
x(5)=x(4);
x(4)=x(3);
x(3)=x(2);
x(2)=x(1);
x(1)=0;
else est=1;
x(6)=x(5);
x(5)=x(4);
x(4)=x(3);
x(3)=x(2);
x(2)=x(1);
x(1)=1;
end
        if ts==0
        po=ts+0.1;
                while po>cont
                z=z+1;
                to(z)=ts+cont;
                sal(z)=est;
                cont=cont+0.01;
                end
        elseif ts~=0
        po=ts+0.1;
        cont=0;
                while po>(ts+cont)
                z=z+1;
                to(z)=ts+cont;
                sal(z)=est;
                cont=cont+0.01;
                end
        end
end
save temp to sal
%plot(to(1:1000),sal(1:1000)),title('SEÑAL BINARIA PSEUDO RANDOMICA'),xlabel('tiempo'),...
%ylabel('magnitud'),pause
%clc
%clg
```

A.2.7. SUBRUTINAS DE AYUDA LLAMADAS DESDE LAS RUTINAS.

PB.M

```
clc
echo on
% Esquema de tolerancia de un filtro pasa bajos:
%
%
%                magnitud
%                   |
%                   |
%        Rp(dB)     |
%                   |
%                   |
%                   |
%                   |
%                   |
%            Rs(dB) |
%                   |_____________>Frecuencia(Hz)
%                     Fp  Fs
%
%
pause % oprima cualquier tecla
```

PA.M

```
clc
echo on
% Esquema de tolerancia de un filtro pasa altos
%
%
%
%                magnitud
%                   |
%                   |
%                   |     Rp(dB)
%                   |
%                   |
%                   |
%        Rs(dB)     |
%                   |_____________>Frecuencia(Hz)
%                     Fs  Fp
%
%
pause % oprima cualquier tecla
```

PBAN.M

```
clc
echo on
%  Esquema de tolerancia para filtro pasa banda
%
%
%                      magnitud
%
%
%
%               Rp(dB)
%
%
%
%
%               Rs(dB)
%                                          >Frecuencia(Hz)
%                      Fsl  Fpl    Fpu  Fsu
%
pause % oprima cualquier tecla
```

EBAN.M

```
clc
echo on
% Esquema de tolerancia de un filtro elimina banda
%
%                     magnitud
%
%
%           Rp(dB)
%
%
%
%
%           Rs(dB)
%                                  > Frecuencia (Hz)
%                     Fpl  Fsl    Fsu  Fpu
%
pause % oprima cualquier tecla
```

PB1.M

```
clc
echo on
% Esquema de tolerancia de un filtro pasa bajos:
%
%
%                magnitud
%
%
%            1
%
%
%
%
%
%                                        >Frecuencia(Hz)
%                       Fc
%
%
pause % oprima cualquier tecla
```

PA1.M

```
clc
echo on
% Esquema de tolerancia de un filtro pasa altos
%
%
%
%                magnitud
%
%            1
%
%
%
%
%
%                                        >Frecuencia(Hz)
%                       Fc
%
%.
pause % oprima cualquier tecla
```

PBAN1.M

```
clc
echo on
%  Esquema de tolerancia para filtro pasa banda
%
%
%                        magnitud
%
%
%
%                   1
%
%
%
%
%
%
%                                                  >Frecuencia(Hz)
%                          Fcl         Fcu
%
pause % oprima cualquier tecla
```

EBAN1.M

```
clc
echo on
% Esquema de tolerancia de un filtro elimina banda
%
%                      magnitud
%
%
%
%                  1
%
%
%
%
%
%
%
%                                                  > Frecuencia (Hz)
%                          Fcl         Fcu
%
pause % oprima cualquier tecla
```

TUTOR.M

```
clc
%                              EJEMPLO
%
%
%       Para definir un filtro pasa banda con frecuencias de corte de 200
%       y 300 Hz, se deben definir los vectores F y M de la siguiente ma-
%       nera, representando la respuesta de frecuencia ideal del filtro
%       que se va a diseñar y tomando en cuenta la definición de disconti-
%       nuidad para ello:
%
%                                Magnitud
%
%                          1-
%
%
%
%
%
%
%                          0-                                  >Frecuencia(Hz)
%                            0    200     300          fs/2
%
pause % oprima cualquier tecla
%
%       F = [ 0    200    200    300    300    fs/2 ]
%
%       M = [ 0     0      1      1      0      0  ]
%
%
pause % oprima cualquier tecla
```

A.2.8. RUTINAS ADICIONALES.

IMPRIMIR.M

```
echo on
%
% El programa saldrá al DOS, donde deberá escribirse: GRAPH PDS
%
% GRAPH.BAT está configurado para funcionar con impresoras EPSON.
%
% Luego de imprimir, escribir PDS en línea de comando DOS y en línea
% de comando del MATLAB, para volver a entrar al programa. Para recu-
% perar el espacio de trabajo anterior a la impresión, teclear desde
% línea de comandos del MATLAB "load", el cual cargará todas las va-
% riables desde el archivo matlab.mat.
%
pause % Oprima cualquier tecla...
echo off
save
exit
```

TDF.M

```
delete pds.met
clc
echo on
%
%
%       Programa para calcular el espectro de una señal cualquiera con
%       posibilidad de variar el número de puntos de la Transformada
%       Discreta de Fourier (TDF).
%
%
echo off
%n=input('el valor de n es: ');
%i=-n:.1:n;
fs=input('La frecuencia de muestreo es fs (Hz):  ');
t=0:1/fs:999/fs;
x=input('La señal de interés es f(t):  ');
plot(t,x),meta pds,title('Señal objeto de análisis'),xlabel('tiempo'),pause
n=input('el número de puntos para la TDF es n:  ');
Y=fft(x,n);
%plot(abs(Y)),title('TDF de la función definida'),pause
Pyy=Y.*conj(Y);
f=fs/n*(0:(n/2)-1);
plot(f,Pyy(1:n/2)),meta,title('Densidad Espectral de Potencia'),...
xlabel('Frecuencia (Hz)'),pause
clc
clg
```