4 Filtros

En esta sección se presentan algunas rutinas de Matlab relativas al papel de los filtros. Recuérdese que, a fin de cuentas, cualquier circuito eléctrico es un filtro, ya que no responde de la misma forma a todas las frecuencias aplicadas a su entrada.

En el primer apartado presentamos los cuatro tipos básicos de filtros: paso bajo, paso alto, paso banda, y de banda rechazada (o eliminada). Se indica una rutina distinta para cada tipo. En todos los casos el filtro está caracterizado por su función de transferencia, expresada como el cociente de dos polinomios de la variable s. El usuario ha de definir esta función en las primeras líneas de cada rutina.

Hay ligeras diferencias entre las cuatro rutinas, por lo que resulta conveniente elegir la correspondiente al tipo de filtro que se desea considerar. Si no se sabe a priori a qué tipo de filtro corresponde una determinada función de transferencia, el usuario puede hacer una estimación preliminar usando la instrucción freqs. La inspección de la curva de variación del módulo con la frecuencia proporcionada por esta instrucción nos indicará el tipo de filtro del que se trata.

Las rutinas presentadas en el primer apartado están básicamente enfocadas a proporcionar representaciones gráficas útiles sobre el comportamiento de filtros. En ese sentido tienen mucho parecido con las presentadas en el apartado 2.3. Lo que añaden de novedoso es la representación de la atenuación introducida por el filtro y, sobre todo, el cálculo de frecuencias importantes (frecuencia central, frecuencias que limitan la banda de paso) para describir el comportamiento de un filtro.

El segundo apartado concierne a los filtros con respuesta Butterworth o Chebyshev. Las rutinas contenidas en él permiten diseñar un filtro en función de la respuesta deseada. El diseño empieza siempre con el cálculo del prototipo de filtro paso bajo normalizado. A esta tarea se le dedican dos rutinas específicas, dependiendo de la respuesta elegida por el usuario. En el apartado también se indica cómo pasar del prototipo normalizado al filtro realmente deseado.

4.1 Tipos de filtros

Seguidamente se muestran las rutinas (y los resultados que permiten obtener) correspondientes a los cuatro tipos básicos de filtros.

```
%%%%% FILTRO PASO BAJO        %%%%%

clear all;                    % Elimina variables utilizadas en otras rutinas

% Rango de frecuencias
d1 = 0; d2 = 6; puntos = 10000;
w = logspace(d1, d2, puntos);

% Función de transferencia
R = 1; L = 0.001;
s = i*w;
H = (R/L)./(s + (R/L));

% Cálculos
modulo = abs(H);
atenuacion = -20*log10(modulo);
fase = (180/pi)*unwrap(angle(H));
Hc = max(modulo)/sqrt(2);
bandapaso = find(modulo>=Hc); n = length(bandapaso); wc = w(n);

% Representación gráfica

subplot(3, 1, 1);
semilogx(w, modulo, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('Módulo', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), -(1/4)*max(modulo), (3/2)*max(modulo)]);
set(gca, 'xtick', [w(1) wc w(puntos)], 'FontName', 'Times', 'Fontsize', 12);
set(gca, 'ytick', [0 Hc max(modulo)], 'FontName', 'Times', 'Fontsize', 12);
title ('Filtro paso bajo', 'FontName', 'Times', 'Fontsize', 24);

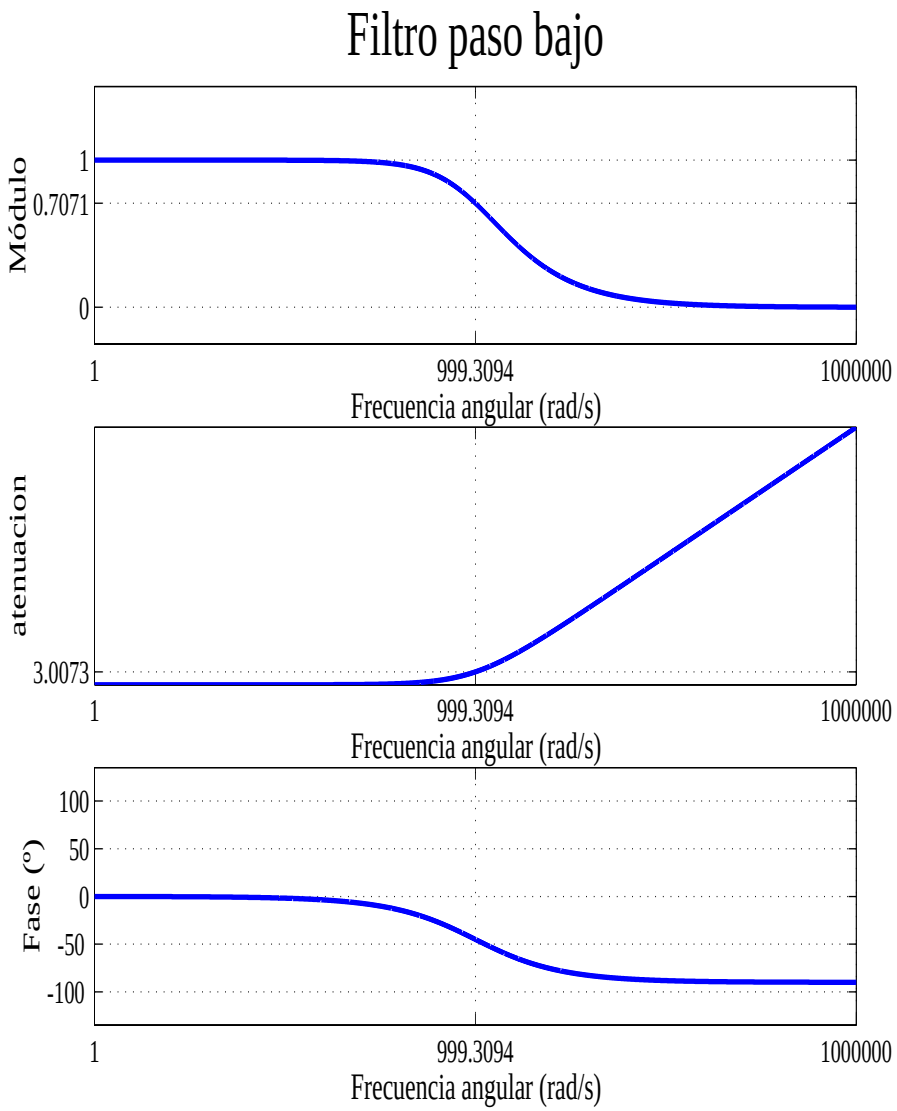
subplot(3, 1, 2);
semilogx(w, atenuacion, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('atenuacion', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), min(atenuacion), max(atenuacion)]);
set(gca, 'xtick', [w(1) wc w(puntos)], 'FontName', 'Times', 'Fontsize', 12);
set(gca, 'ytick', [0 atenuacion(n)], 'FontName', 'Times', 'Fontsize', 12);

subplot(3, 1, 3);
semilogx(w, fase, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('Fase (º)', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), -(3/2)*max(abs(fase)), (3/2)*max(abs(fase))]);
set(gca, 'xtick', [w(1) wc w(puntos)], 'FontName', 'Times', 'Fontsize', 12);

clear all;                    % Elimina las variables utilizadas en esta rutina
```

$$a = 1000$$

$$H(s) = \frac{a}{s + a}$$



Obsérvese la forma en la que se ha calculado la frecuencia de corte. También puede notarse que no se etiqueta de forma especial el eje de ordenadas en la representación de la variación de la fase. Dado que aquélla es ignorada a priori, con tal decisión se respeta el numerado automático del eje que efectúa Matlab. Finalmente, obsérvese la forma de la instrucción con la que se obtiene el logaritmo en base 10 en el cálculo de la atenuación; si se omitiera la indicación 10, lo que se obtendría sería el logaritmo neperiano (base e).

```
%%%%% FILTRO PASO ALTO        %%%%%

clear all;                    % Elimina variables utilizadas en otras rutinas

% Rango de frecuencias
d1 = 0; d2 = 6; puntos = 10000;
w = logspace(d1, d2, puntos);

% Función de transferencia
R = 1; L = 0.001;
s = i*w;
H = s./(s + (R/L));

% Cálculos
modulo = abs(H);
atenuacion = -20*log10(modulo);
fase = (180/pi)*unwrap(angle(H));
Hc = max(modulo)/sqrt(2);
bandapaso = find(modulo<=Hc); n = length(bandapaso); wc = w(n);

% Representación gráfica

subplot(3, 1, 1);
semilogx(w, modulo, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('Módulo', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), -(1/4)*max(modulo), (3/2)*max(modulo)]);
set(gca, 'xtick', [w(1) wc w(puntos)], 'FontName', 'Times', 'Fontsize', 12);
set(gca, 'ytick', [0 Hc max(modulo)], 'FontName', 'Times', 'Fontsize', 12);
title ('Filtro paso alto', 'FontName', 'Times', 'Fontsize', 24);

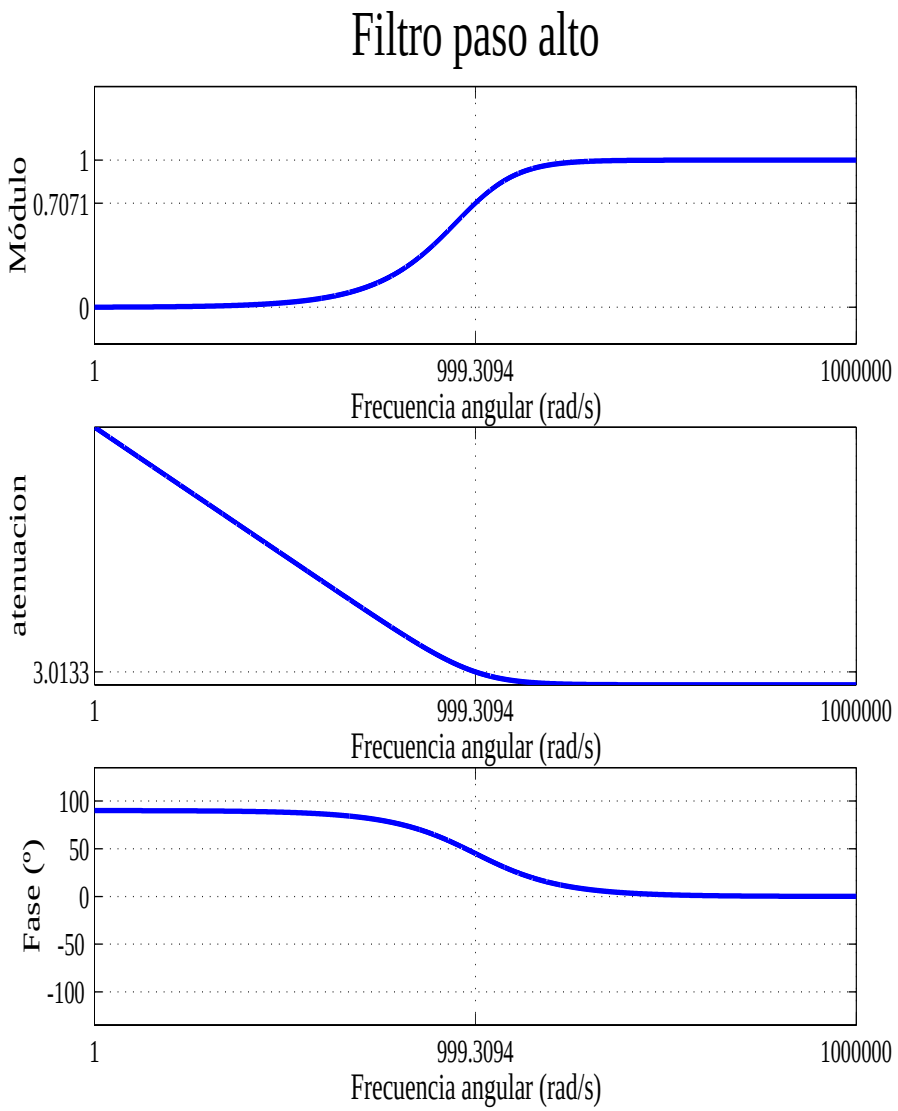
subplot(3, 1, 2);
semilogx(w, atenuacion, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('atenuacion', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), min(atenuacion), max(atenuacion)]);
set(gca, 'xtick', [w(1) wc w(puntos)], 'FontName', 'Times', 'Fontsize', 12);
set(gca, 'ytick', [0 atenuacion(n)], 'FontName', 'Times', 'Fontsize', 12);

subplot(3, 1, 3);
semilogx(w, fase, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('Fase (º)', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), -(3/2)*max(abs(fase)), (3/2)*max(abs(fase))]);
set(gca, 'xtick', [w(1) wc w(puntos)], 'FontName', 'Times', 'Fontsize', 12);

clear all;                    % Elimina las variables utilizadas en esta rutina
```

a = 1000

$$H(s) = \frac{s}{s + a}$$



Con relación a esta rutina pueden hacerse observaciones similares a las correspondientes al caso del filtro paso bajo.

```
%%%%% FILTRO PASO BANDA       %%%%%

clear all;                    % Elimina variables utilizadas en otras rutinas

% Rango de frecuencias
d1 = 1; d2 = 5; puntos = 10000;
w = logspace(d1, d2, puntos);

% Función de transferencia
R = 1; L = 0.001; C = 0.001;
s = i*w;
H = (R/L)*s./(s.^2 + (R/L)*s + 1/(L*C));

% Cálculos
modulo = abs(H);
atenuacion = -20*log10(modulo);
fase = (180/pi)*unwrap(angle(H));
Hc = max(modulo)/sqrt(2); bandapaso = find(modulo>=Hc); n = length(bandapaso);
w1 = w(bandapaso(1)); w2 = w(bandapaso(n)); w0 = sqrt(w1*w2);
bandapaso0 = find(w<=w0); n0 = length(bandapaso0);

% Representación gráfica

subplot(3, 1, 1);
semilogx(w, modulo, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('Módulo', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), -(1/4)*max(modulo), (3/2)*max(modulo)]);
set(gca, 'xtick', [w(1) round(w1) round(w0) round(w2) w(puntos)],
'FontName', 'Times', 'Fontsize', 12);
set(gca, 'ytick', [0 Hc max(modulo)], 'FontName', 'Times', 'Fontsize', 12);
title ('Filtro paso banda', 'FontName', 'Times', 'Fontsize', 24);

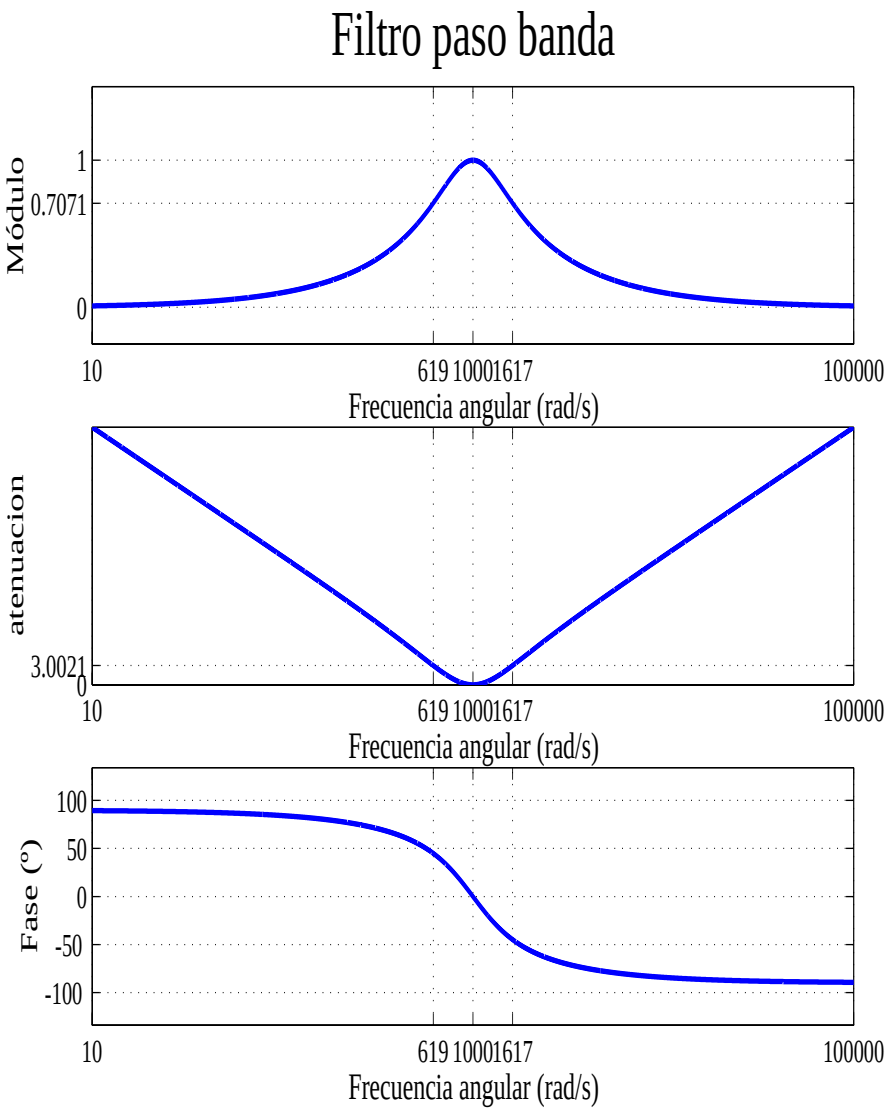
subplot(3, 1, 2);
semilogx(w, atenuacion, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('atenuacion', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), min(atenuacion), max(atenuacion)]);
set(gca, 'xtick', [w(1) round(w1) round(w0) round(w2) w(puntos)],
'FontName', 'Times', 'Fontsize', 12);
set(gca, 'ytick', [0 atenuacion(n0) atenuacion(bandapaso(1))],
'FontName', 'Times', 'Fontsize', 12);

subplot(3, 1, 3);
semilogx(w, fase, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('Fase (º)', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), -(3/2)*max(abs(fase)), (3/2)*max(abs(fase))]);
set(gca, 'xtick', [w(1) round(w1) round(w0) round(w2) w(puntos)],
'FontName', 'Times', 'Fontsize', 12);

clear all;                    % Elimina las variables utilizadas en esta rutina
```

$a = 1000,\ b = 10^6$

$$H(s) = \frac{as}{s^2 + as + b}$$



Obsérvese la forma particular en la que se especifican los puntos del eje de abscisas que han de ser marcados especialmente; se utiliza la instrucción round (que redondea el argumento al entero más próximo) con objeto de que los valores de las frecuencias particulares estén lo suficientemente separados como para ser leídos. Nótese también cómo se calculan las frecuencias que definen la banda de paso del filtro.

```
%%%%% FILTRO DE BANDA RECHAZADA           %%%%%

clear all;                    % Elimina variables utilizadas en otras rutinas

% Rango de frecuencias
d1 = 1; d2 = 5; puntos = 10000;
w = logspace(d1, d2, puntos);

% Función de transferencia
R = 1; L = 0.001; C = 0.001;
s = i*w;
H = (s.^2 + 1/(L*C))./(s.^2 + (R/L)*s + 1/(L*C));

% Cálculos
modulo = abs(H);
atenuacion = -20*log10(modulo);
fase = (180/pi)*unwrap(angle(H));
Hc = max(modulo)/sqrt(2); bandapaso = find(modulo<=Hc); n = length(bandapaso);
w1 = w(bandapaso(1)); w2 = w(bandapaso(n)); w0 = sqrt(w1*w2);
bandapaso0 = find(w<=w0); n0 = length(bandapaso0);

% Representación gráfica

subplot(3, 1, 1);
semilogx(w, modulo, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('Módulo', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), -(3/2)*min(modulo), (3/2)*max(modulo)]);
set(gca, 'xtick', [w(1) round(w1) round(w0) round(w2) w(puntos)],
'FontName', 'Times', 'Fontsize', 12);
set(gca, 'ytick', [0 min(modulo) Hc max(modulo)],
'FontName', 'Times', 'Fontsize', 12);
title ('Filtro de banda rechazada', 'FontName', 'Times', 'Fontsize', 24);

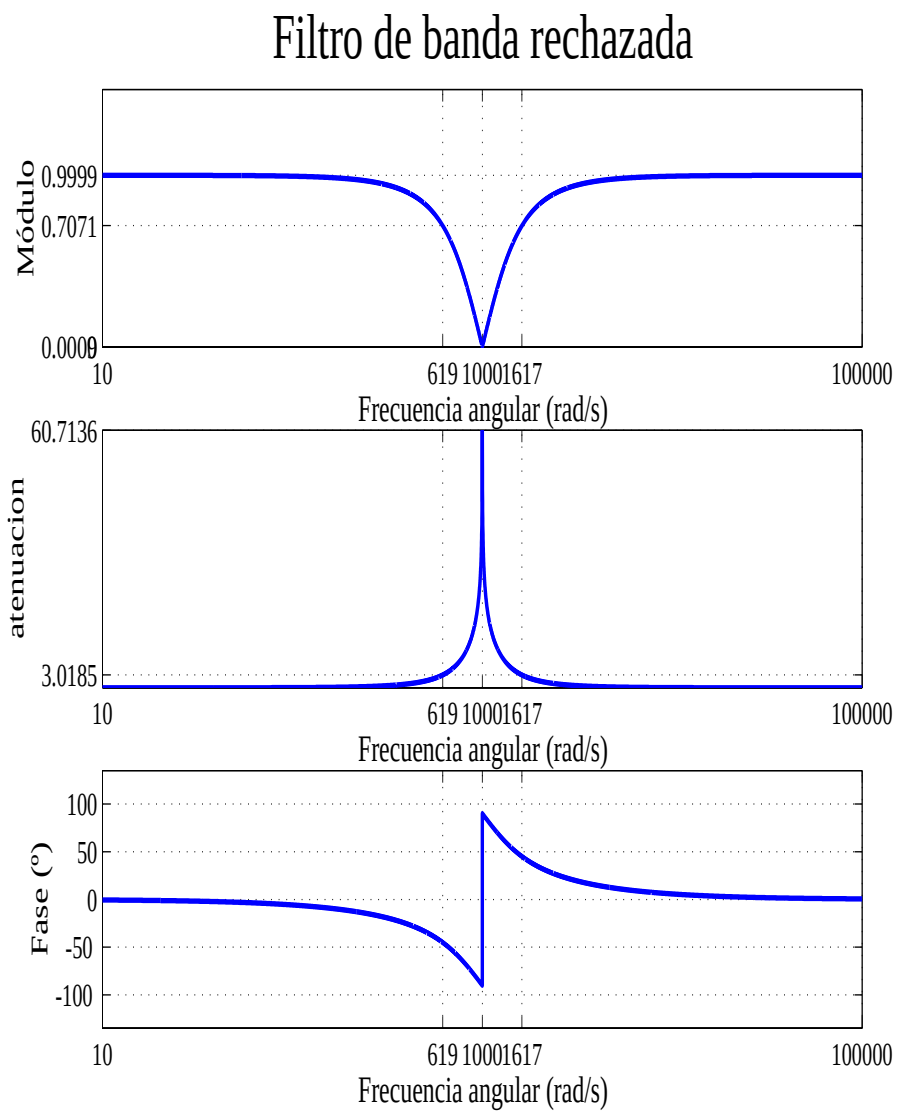
subplot(3, 1, 2);
semilogx(w, atenuacion, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('atenuacion', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), min(atenuacion), max(atenuacion)]);
set(gca, 'xtick', [w(1) round(w1) round(w0) round(w2) w(puntos)],
'FontName', 'Times', 'Fontsize', 12);
set(gca, 'ytick', [0  atenuacion(bandapaso(1)) atenuacion(n0)],
'FontName', 'Times', 'Fontsize', 12);

subplot(3, 1, 3);
semilogx(w, fase, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel('Fase (º)', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), -(3/2)*max(abs(fase)), (3/2)*max(abs(fase))]);
set(gca, 'xtick', [w(1) round(w1) round(w0) round(w2) w(puntos)],
'FontName', 'Times', 'Fontsize', 12);

clear all;                    % Elimina las variables utilizadas en esta rutina
```

$a = 1000,\ b = 10^6$

$$H(s) = \frac{s^2 + b}{s^2 + as + b}$$



Con respecto a esta rutina pueden hacerse observaciones similares a las correspondientes a la rutina relativa al filtro paso banda.

4.2 Diseño de filtros

El diseño de filtros con respuestas Butterworth o Chebyshev (los únicos considerados en este manual) comienza con el diseño de un prototipo de filtro paso bajo normalizado. Tal diseño se efectúa en función de las especificaciones del usuario sobre rangos de frecuencias, atenuaciones a frecuencias dadas, y resistencias de generador y carga; naturalmente, también es preciso especificar el tipo de filtro deseado (paso bajo, paso alto, paso banda y de banda rechazada). Los resultados del diseño del prototipo son el número de elementos que contiene aquél y los valores de tales elementos. El diseño conduce a resultados distintos según que la respuesta deseada sea Butterworth o Chebyshev.

Seguidamente se muestra una rutina que permite obtener el prototipo de filtro normalizado. Como resultado intermedio, la rutina muestra la variación de la atenuación con la frecuencia. La observación de esta figura permite al usuario asegurar que se trata del tipo de filtro en el que está interesado y que cumple las especificaciones sobre atenuación

```
%%%%% PROTOTIPO DE FILTRO NORMALIZADO (Butterworth)         %%%%%

clear all;                    % Elimina variables utilizadas en otras rutinas

tipo = 1;   % Tipo de filtro (1: bajo; 2: alto; 3: banda; 4: banda rechazada)

% Datos de frecuencias (rad/s), atenuaciones (dB) y resistencias (Ω)

wc = 1000;          % Frecuencia de corte (no necesaria si tipo = 3 o tipo = 4)
ws = 5000;          % Frecuencia fuera de la banda de paso (> w0 para tipo = 3, 4)
w0 = 100000;        % Frecuencia central (no necesaria si tipo = 1 o tipo = 2)
bwrel = 0.1;        % Ancho de banda relativo (no necesario si tipo = 1 o tipo = 2)
Amax = 3;           % Atenuación máxima en la banda de paso (valor fijo)
Amin = 30;          % Atenuación mínima exigida para w = ws
Rg = 50;            % Resistencia de generador
Rl = 50;            % Resistencia de carga

% Rango de frecuencias

inicial = 0; final = 6; puntos = 10000;
w = logspace(inicial, final, puntos);

% Normalización de frecuencias

     if tipo == 1
            wnorm = ws/wc; wn = w/wc;
     elseif tipo == 2
            wnorm = wc/ws; wn = wc./w;
     elseif tipo == 3
            wnorm = ((ws/w0) - (w0/ws))/bwrel; wn = ((w/w0) - (w0./w))/bwrel;
     else
            wnorm = bwrel/((ws/w0) - (w0/ws)); wn = bwrel./((w/w0) - (w0./w));
     end
```

```
% Realizabilidad del filtro

if wnorm <= 1
     disp('El filtro no es realizable con las especificaciones indicadas');
else

% Número de elementos del prototipo

amax = 10^(Amax/10) - 1; amin = 10^(Amin/10) - 1;
n = ceil(log10(amin/amax)/(2*log10(wnorm)));

% Representación gráfica de la atenuación

A = 10*log10(1 + wn.^(2*n));
semilogx (w, A, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel ('Atenuación (dB)', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), min(A), (3/2)*max(A)]);
hold on;

% Cálculo del prototipo

     if Rg >= Rl                    % Estructura del filtro
            disp('El filtro empieza con un elemento en paralelo')
     else
            disp('El filtro empieza con un elemento en serie')
     end

     if Rg == Rl                    % Caso particular (fórmulas de Bennett)
      i = 1;
            while i<=n
                  g(i) = 2*sin((2*i -1)*pi/(2*n));
                  i = i + 1;
            end
     else                           % Caso general (fórmulas de Bossé)
      K = 4*Rg*Rl/((Rg + Rl)^2); alfa = (1 - K)^(1/(2*n));
      i = 1;
      b(i) = 1 + alfa^2 - 2*alfa*cos(i*pi/n);
      x(i) = sin((2*i -1)*pi/(2*n));
      g(i) = 2*x(1)/(1 - alfa);
      i = 2;
            while i<=n
                  b(i) = 1 + alfa^2 - 2*alfa*cos(i*pi/n);
                  x(i) = sin((2*i -1)*pi/(2*n));
                  g(i) = 4*x(i-1)*x(i)/(b(i-1)*g(i-1));
                  i = i + 1;
            end
     end

disp('i ='); disp(1:n); disp('g = '); disp(g);

end

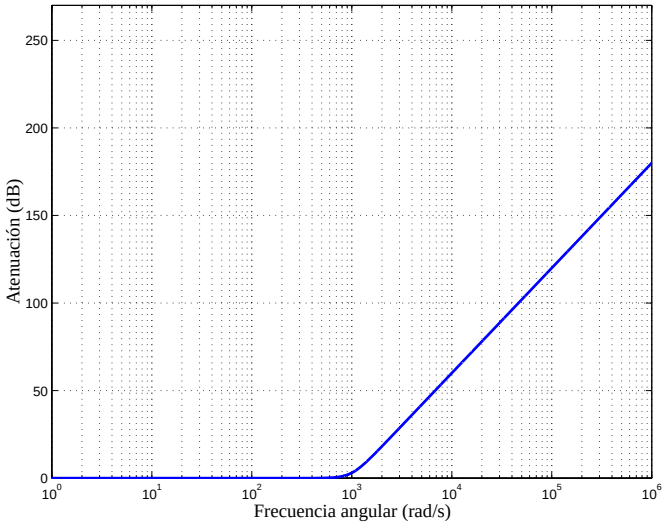
clear all;                    % Elimina las variables utilizadas en esta rutina
```

Obsérvese que, para presentar resultados en la ventana de comandos, en esta rutina se utilizan las instrucciones

```
disp('texto')          % Presenta el texto utilizado como argumento
disp(y)                % Presenta el valor de y
```

La rutina genera los resultados que se presentan seguidamente.

El filtro empieza con un elemento en paralelo
i = 1 2 3
g = 1 2 1



La rutina equivalente para el caso de respuesta Chebyshev es la que se muestra seguidamente.

```
%%%%% PROTOTIPO DE FILTRO NORMALIZADO (Chebyshev)          %%%%%

clear all;                    % Elimina variables utilizadas en otras rutinas

tipo = 1;   % Tipo de filtro (1: bajo; 2: alto; 3: banda; 4: banda rechazada)

% Datos de frecuencias (rad/s), atenuaciones (dB) y resistencias (Ω)

wc = 1000;          % Frecuencia de corte (no necesaria si tipo = 3 o tipo = 4)
ws = 5000;          % Frecuencia fuera de la banda de paso (> w0 para tipo = 3, 4)
w0 = 100000;        % Frecuencia central (no necesaria si tipo = 1 o tipo = 2)
bwrel = 3;          % Ancho de banda relativo (no necesario si tipo = 1 o tipo = 2)
Amax = 3;           % Atenuación máxima en la banda de paso (valor fijo)
Amin = 50;          % Atenuación mínima exigida para w = ws
Rg = 50;            % Resistencia de generador
Rl = 75;            % Resistencia de carga

% Rango de frecuencias

inicial = 0; final = 6; puntos = 10000;
w = logspace(inicial, final, puntos);
```

```
% Normalización de frecuencias

     if tipo == 1
            wnorm = ws/wc; wn = w/wc;
     elseif tipo == 2
            wnorm = wc/ws; wn = wc./w;
     elseif tipo == 3
            wnorm = ((ws/w0) - (w0/ws))/bwrel; wn = ((w/w0) - (w0./w))/bwrel;
     else
            wnorm = bwrel/((ws/w0) - (w0/ws)); wn = bwrel./((w/w0) - (w0./w));
     end

% Realizabilidad del filtro

if wnorm <= 1
     disp('El filtro no es realizable con las especificaciones indicadas');
else

% Número de elementos del prototipo

amax = 10^(Amax/10) - 1; amin = 10^(Amin/10) - 1;
n = ceil((acosh(sqrt(amin/amax)))/(acosh(wnorm)));

% Representación gráfica de la atenuación

rango1 = find(wn<=1);
A(rango1) = 10*log10(1 + amax*cos(n*acos(wn(rango1))).^2);
rango2 = find(wn>1);
A(rango2) = 10*log10(1 + amax*cosh(n*acosh(wn(rango2))).^2);
semilogx (w, A, 'b', 'LineWidth', 2);
grid on;
xlabel('Frecuencia angular (rad/s)', 'FontName', 'Times', 'Fontsize', 14);
ylabel ('Atenuación (dB)', 'FontName', 'Times', 'Fontsize', 14);
axis([w(1), w(puntos), min(A), (3/2)*max(A)]);
hold on;

% Cálculo del prototipo (fórmulas de Takahasi)

     if((rem(n,2) == 0) & (Rg == Rl))     % Realizabilidad del filtro
            disp('No es posible obtener un filtro Chebyshev de orden par');
            disp('en el que las resistencias de generador y de carga son iguales');

     else
            if Rg > Rl                                % Estructura del filtro
            disp('El filtro empieza con un elemento en paralelo')
            else
            disp('El filtro empieza con un elemento en serie')
            end

            K = 4*Rg*Rl/((Rg + Rl)^2);          % Elementos del prototipo
            alfa = asinh(sqrt(1/amax))/n;
            beta = asinh(sqrt((1 - K)/amax))/n;
            i = 1;
            b(i) = (sinh(alfa))^2 + (sinh(beta))^2 + (sin(i*pi/n))^2 -
            - 2*sinh(alfa)*sinh(beta)*cos(i*pi/n);
            x(i) = sin((2*i -1)*pi/(2*n));
            g(i) = 2*x(i)/(sinh(alfa) - sinh(beta));
            i = 2;
```

```
            while i<=n
                  b(i) = (sinh(alfa))^2 + (sinh(beta))^2 + (sin(i*pi/n))^2 -
                  - 2*sinh(alfa)*sinh(beta)*cos(i*pi/n);
                  x(i) = sin((2*i -1)*pi/(2*n));
                  g(i) = 4*x(i-1)*x(i)/(b(i-1)*g(i-1));
                  i = i + 1;
            end

            disp('i ='); disp(1:n); disp('g = '); disp(g);
      end

end

clear all;                    % Elimina las variables utilizadas en esta rutina
```

En esta rutina se utiliza la instrucción

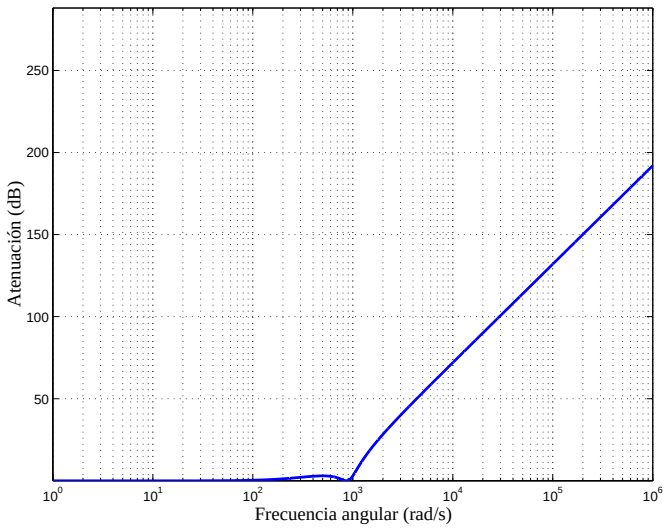
```
rem(x, y)
```
% Proporciona el resto de la división de x entre y

En la forma empleada en la rutina permite determinar si el número de elementos del prototipo es par (resto nulo) o impar.

Con esta rutina se obtienen resultados como los mostrados a continuación.

El filtro empieza con un elemento en serie

i = 1	2	3
g = 4.3069	0.5637	4.1090



En las rutinas precedentes el usuario puede modificar los datos correspondientes para que el cálculo se ajuste a lo que realmente desea.

Una vez calculados los elementos del prototipo normalizado (sea cual sea la respuesta deseada), lo que procede es obtener el filtro real; es decir, obtener los valores de las inductancias y capacidades que realmente constituyen en el filtro. La rutina que sigue permite realizar tal cálculo.

```
%%%%% CÁLCULO DE LOS ELEMENTOS DE UN FILTRO          %%%%%

clear all;                    % Elimina variables utilizadas en otras rutinas

tipo = 3;   % Tipo de filtro (1: bajo; 2: alto; 3: banda; 4: banda rechazada)

% Datos de frecuencias (rad/s), resistencias (Ω) y estructura
wc = 2000;          % Frecuencia de corte (no necesaria si tipo = 3 o tipo = 4)
w0 = 3000;          % Frecuencia central (no necesaria si tipo = 1 o tipo = 2)
bwrel = 0.1;        % Ancho de banda relativo (no necesario si tipo = 1 o tipo = 2)
Rg = 60;            % Resistencia de generador
Rl = 50;            % Resistencia de carga
est = 1;            % Estructura (1: empieza en paralelo; 2: empieza en serie)

% Datos del prototipo de filtro normalizado
n = 3;                                % Número de elementos del prototipo
i = 1:n;                              % Indicativo de los elementos del protipo
g = [3.7274 0.6457 3.6479];           % Elementos del prototipo

% Inicialización
     for i = 1:n
            L(i) = 0; C(i) = 0;
     end

% Cálculo de los elementos
     if tipo == 1
      i = 1;
            if est == 1
                  while i<=n
                        C(i) = g(i)/(Rg*wc);
                        if i<n
                              L(i+1) = Rg*g(i+1)/wc;
                        end
                        i = i + 2;
                  end
            else
                  while i<=n
                        L(i) = Rg*g(i)/wc;
                        if i<n
                              C(i+1) = g(i+1)/(Rg*wc);
                        end
                        i = i + 2;
                  end
            end
     end

     if tipo == 2
      i = 1;
            if est == 1
                  while i<=n
                        L(i) = Rg/(g(i)*wc);
                        if i<n
                              C(i+1) = 1/(Rg*g(i+1)*wc);
                        end
                        i = i + 2;
                  end
            else
                  while i<=n
```

```
                    C(i) = 1/(Rg*g(i)*wc);
                    if i<n
                          L(i+1) = Rg/(g(i+1)*wc);
                    end
                    i = i + 2;
              end
        end
end

if tipo == 3
 i = 1;
        if est == 1
              while i<=n
                    L(i) = Rg*bwrel/(g(i)*w0); C(i) = g(i)/(bwrel*Rg*w0);
                    if i<n
                          L(i+1) = Rg*g(i+1)/(bwrel*w0);
                          C(i+1) = bwrel/(Rg*g(i+1)*w0);
                    end
                    i = i + 2;
              end
        else
              while i<=n
                    L(i) = Rg*g(i)/(bwrel*w0); C(i) = bwrel/(Rg*g(i)*w0);
                    if i<n
                          L(i+1) = Rg*bwrel/(g(i+1)*w0);
                          C(i+1) = g(i+1)/(bwrel*Rg*w0);
                    end
                    i = i + 2;
              end
        end
end

if tipo == 4
 i = 1;
        if est == 1
              while i<=n
                    L(i) = Rg/(bwrel*g(i)*w0); C(i) = bwrel*g(i)/(Rg*w0);
                    if i<n
                          L(i+1) = bwrel*Rg*g(i+1)/w0;
                          C(i+1) = 1/(bwrel*Rg*g(i+1)*w0);
                    end
                    i = i + 2;
              end
        else
              while i<=n
                    L(i) = bwrel*Rg*g(i)/w0; C(i) = 1/(bwrel*Rg*g(i)*w0);
                    if i<n
                          L(i+1) = Rg/(bwrel*g(i+1)*w0);
                          C(i+1) = bwrel*g(i+1)/(Rg*w0);
                    end
                    i = i + 2;
              end
        end
end
```

```
% Resultados
     if est == 1
            disp('El filtro empieza con un elemento en paralelo')
     else
            disp('El filtro empieza con un elemento en serie')
     end

disp('i ='); disp(1:n); disp('g ='); disp(g);
disp('L ='); disp(L); disp('C ='); disp(C);

clear all;                    % Elimina las variables utilizadas en esta rutina
```

Los resultados que produce esta rutina son como los que se muestran seguidamente.

```
El filtro empieza con un elemento en paralelo
       i =           1       2       3
   g =           3.7274    0.6457    3.6479
   L =           0.0005    0.1291    0.0005
   C = 1.0e-03 * 0.2071    0.0009    0.2027
```

4.3 Ejercicios propuestos

Ejercicio 13

Determinar el tipo (paso alto, paso bajo, paso banda, de banda eliminada) del filtro cuya función de transferencia es

$$H(s) = \frac{\frac{s}{RC}\left(s^2 + s\frac{R}{L} + \frac{1}{LC}\right)}{s^4 + s^3\left(\frac{R}{L} + \frac{1}{RC}\right) + \frac{4s^2}{LC} + \frac{s}{LC}\left(\frac{R}{L} + \frac{1}{RC}\right) + \frac{1}{L^2C^2}}$$

en la que R = 1 Ω, L = 1 μH y C = 1 μF.

Obsérvese la influencia de R en la frecuencia a la que se obtiene el máximo del módulo de la función de transferencia y en la anchura de la banda de paso (considérense, por ejemplo, los valores 1, 3, 5 y 7 Ω).

Ejercicio 14

Diseñar un filtro paso banda con respuesta Chebyshev que debe ser insertado entre sendas resistencias de 50 (generador) y 75 Ω (carga). La frecuencia central del filtro será de 10000 rad/s, y el ancho de banda del 10 %. Dentro de la banda de paso, la atenuación máxima será de 3 dB. Para una frecuencia de 30000 rad/s, el filtro ha de presentar una atenuación mínima de 70 dB.