Trabajo final TDSÑ - Ecualizador de audio

Andrés Herencia, José Ángel Calderón Villar

30/01/2022

Índice

1. Introducción

El objetivo de este trabajo es crear un ecualizador de audio que compense las posibles distorsiones que pueda tener nuestra señal acústica en una sala, ya sea de concierto, de estudio o de cualquier otro tipo. Esta función de transferencia que caracteriza el sonido de la sala no es uniforme en el rango de frecuencias audible (séase, de 20 Hz a 20 KHz), lo que hace que debamos de considerarla como una distorsión y debamos compensar de alguna manera.

Para hacer esto, tal y como vimos en la parte de teoría es medianamente sencillo recuperar el módulo de nuestra señal original en base a la señal ya distorsionada, aprovechando las características de los sistemas LTI causales y estables. Debemos de tener dos sistemas: uno de ellos será $H_d(z)$, es decir, el **sistema distorsionador** (el que represente nuestra sala). El otro será $H_c(z)$, el **sistema compensador**, que será el que compense la distorsión sufrida por el anterior sistema.

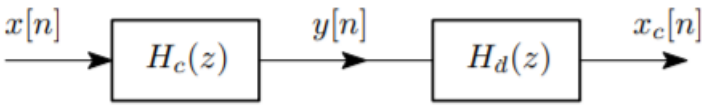

Figura 1: $H_d(z)$ distorsión de la sala, $H_e(z)$ ecualizador.

Para probar que el sistema es correcto, debemos de usar como señal de entrada aquella señal que queramos compensar. Esta pasará por los filtros (el orden es cualquiera, pues ambos sistemas son LTI) y a la salida de

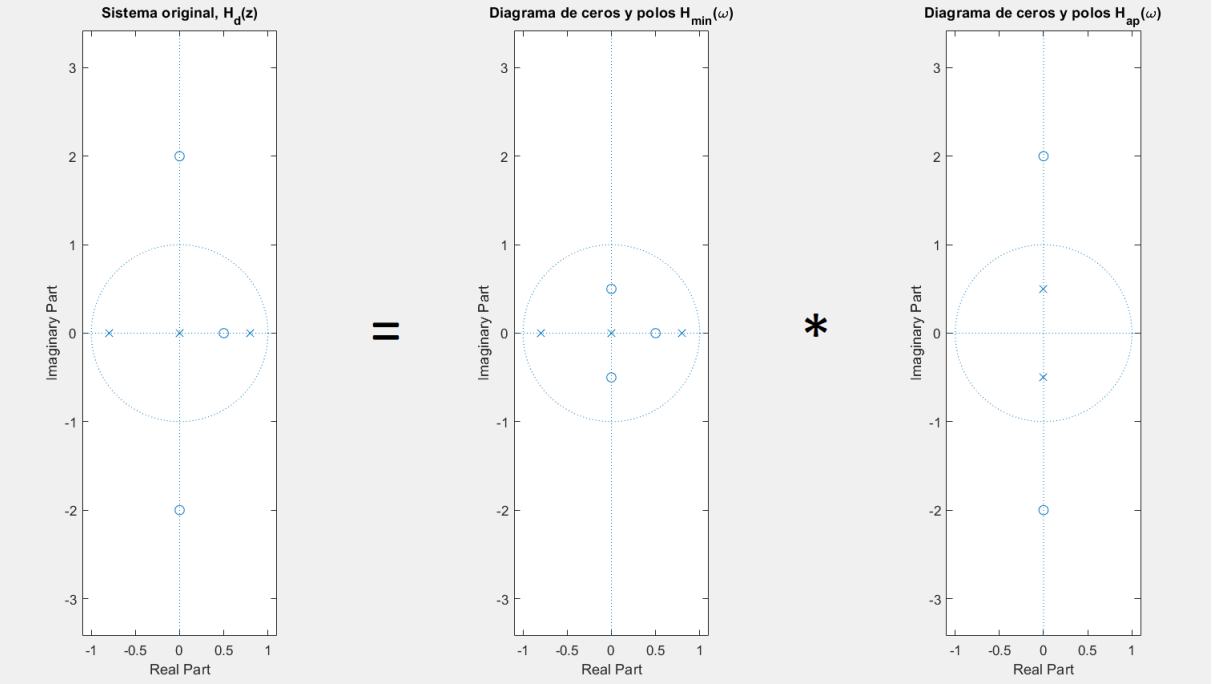

Figura 2: Descomposición de un $H(z)$ genérico en un sistema $H_{min}(z)$ y un sistema $H_{ap}(z)$.

estos dos sistemas, dará la señal compensada. Para probarlo en todo el rango de frecuencias, usaremos una señal *chirp*, que abarca todo el rango de frecuencias a estudiar.

2. Marco Teórico

El sistema distorsionador, $H_d(z)$, por su propia naturaleza (por ser la función de transferencia racional) se puede descomponer en dos filtros: un sistema de fase mínima, $H_{min}(z)$ y un sistema paso todo $H_{ap}(z)$. Teóricamente, el **sistema de fase mínima** ha de tener todos los polos y ceros dentro de la circunferencia unidad. Los ceros que en el sistema original estuvieran fuera de la circunferencia unidad, aparecerán en las posiciones recíprocas conjugadas. Por otro lado, **el sistema paso todo** ha de tener los ceros externos no incluidos en el sistema de fase mínima y los polos en las posiciones recíprocas conjugadas de los mismos. Un ejemplo de este sistema descompuesto sería el siguiente:

Necesitaremos descomponer este sistema para poder compensarlo. Esta compensación vendrá dada por el sistema de dicho nombre, que lo que hará será que el módulo de su función de transferencia se asemeje lo máximo posible al inverso del módulo de la respuesta en frecuencia del sistema de fase mínima.

En la práctica, necesitaremos construir este sistema en base a filtros comerciales. A este nuevo sistema construido con estos filtros lo denominaremos como $H_e(z)$ y tendrá la función del sistema compensador que es irrealizable de manera práctica. Matemáticamente:

$$\mid H_e(z) \mid \approx \mid H_c(z) \mid, \quad con \quad H_c(z) = 1/H_{min}(z) \tag{1}$$

Dicho esto, pasaremos a describir los **filtros de orden 2** cuya expresión general es la siguiente:

$$H_0(z) = \frac{b_0 + b_1 \cdot z^-1 + b_2 \cdot z^-2}{a_0 + a_1 \cdot z^-1 + a_2 \cdot z^-2} \tag{2}$$

Esta expresión general contiene los coeficientes b_i y a_i, que servirán para caracterizar los ceros y polos de cada uno de los filtros, como veremos en la siguiente sección. Los que utilizaremos son en concreto los siguientes:

- El **filtro low shelf** (también conocido como repisa paso bajo) tiene tres parámetros a considerar, la frecuencia de corte, la ganancia y el ancho de banda. La función de este filtro es asignar una ganancia a las frecuencias que estén por debajo de la frecuencia de corte, con un cierto ancho de banda. Tiene como característica especial, que comparte con los otros dos filtros que **las frecuencias a las que no les otorga ganancia** en vez de irse a 0 en unidades lineales **se mantienen en 1** (o en 0 dB en unidades logarítmicas), haciendo que sea un filtro muy estable.
- El **filtro high shelf** (o repisa paso alto) tiene los mismos parámetros a considerar, y la función de este filtro será asignar una ganancia a las frecuencias que estén por encima de la frecuencia central, con un cierto ancho de banda.
- El **filtro peaking EQ** (o pico pasa banda) tiene de parámetros la frecuencia central, la ganancia y el ancho de banda, y asigna una cierta ganancia a una frecuencia central con un cierto ancho de banda.

Cuando conectemos estos filtros en cascada, debemos hacer que se asemejen lo máximo posible a nuestra $H_c(z)$. Lógicamente, cuanto más parecida sea la señal $H_e(z)$ a esta última, mejor será la ecualización. Así, podremos construir nuestro sistema $H_{eq}(z)$, el que usaremos en la "práctica" para compensar la distorsión sufrida por la sala en nuestra señal.

$$\boxed{H_{eq}(z) = H_d(z) * H_e(z)} \tag{3}$$

3. Implementación en MATLAB

3.1. Construcción y obtención de la función de transferencia del sistema compensador perfecto $H_c(z)$

Inicialmente, lo que hacemos es llamar a una función *room*, proporcionada en la propia práctica que lo que hace es simular la distorsión que pueda tener nuestra sala en función de nuestros correos. Esta distorsión estará caracterizada por un sistema racional con polos y ceros en el denominador y numerador respectivamente. Tal como hemos visto en la teoría, debemos descomponer esta distorsión en un sistema de fase mínima y otro de paso todo. Si lo hacemos, obtenemos lo siguiente:

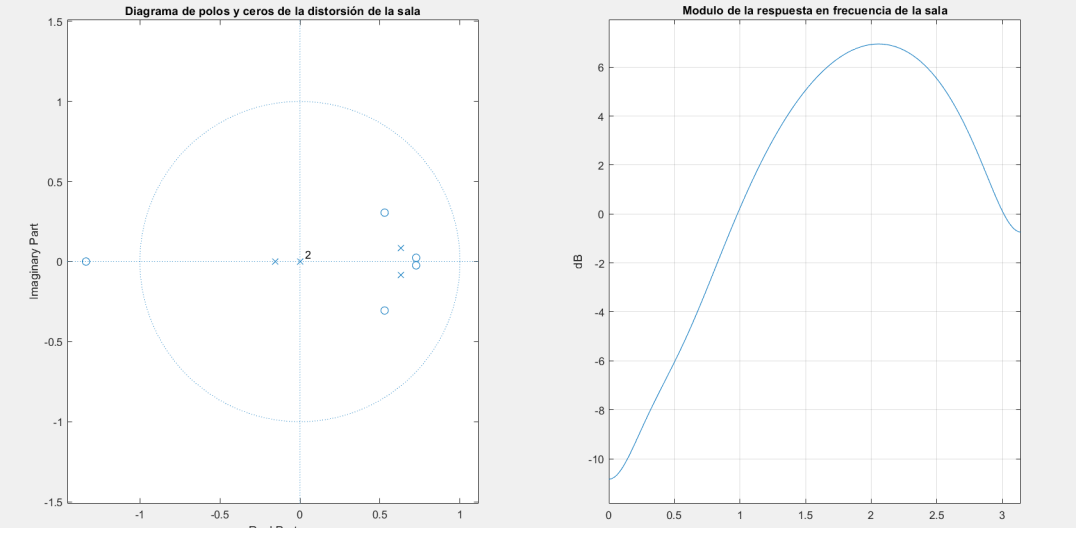

Figura 3: Función de transferencia de la distorsión de la sala $H_d(z)$ y diagrama de polos y ceros de la misma.

Esto se consigue con el siguiente código:

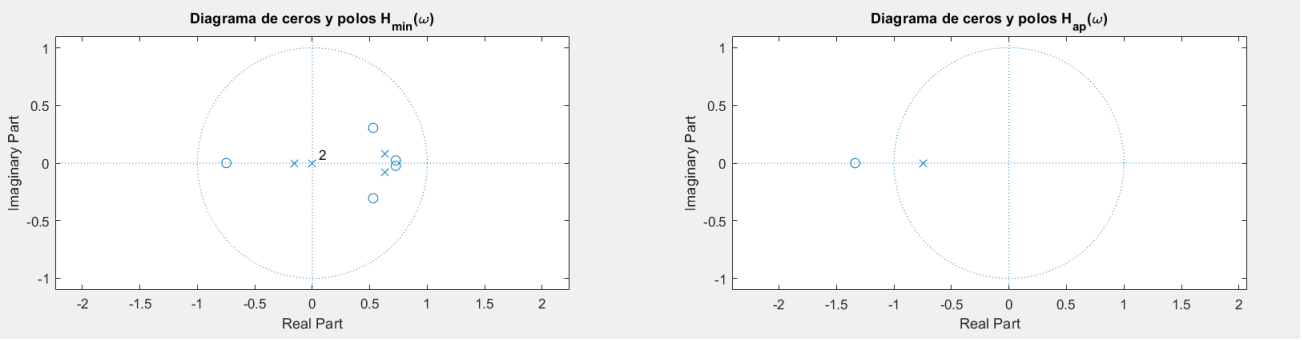

Figura 4: Sistemas de fase mínima y paso todo de $H_d(z)$.

```
%% Hd(z) distorsión creada por la sala

    [Z, P, K] = room('a.herencia@alumnos.upm.es', 'jose.calderon.villar@alumnos.upm.es');
    [B, A] = zp2tf(Z, P, K); % Obtención de constantes B y A
    [H, w] = freqz(B, A, 4096); % Obtención de H y la frecuencia normalizada
    mH = 20*log10(abs(H));
    subplot(1,2,2);
    plot(w, mH);
    title('Modulo de la respuesta en frecuencia de la sala');
    xlabel('\omega'); ylabel('dB'); grid on;
    axis([0, pi, min(mH)-1, max(mH)+1]);
    subplot(1,2,1);
    zplane(Z, P);
    title('Diagrama de polos y ceros de la distorsión de la sala')

% Hmin(z)
    Zmin = Z;
    Pmin = P(abs(P)<=1); % Aquellos polos que estén dentro de la circunferencia unidad
    % los conservamos
    z0 = Z(abs(Z)>1); % Cogemos los ceros que estén fuera de la circunferencia unidad
    zrc = 1/conj(z0); % Calculamos su recíproco conjugado
    Zmin(Zmin==z0)=zrc; % Y sustituimos los nuevos ceros dentro de la c.u. por los que había fuera.
    figure(1);
    subplot(2,2,1);
    zplane(Zmin, Pmin);
    title('Diagrama de ceros y polos H_{min}(\omega)');

% Hap(z)
    Zap = Z(abs(Z)>1);
    Pap = 1/conj(Zap);
    Kap = (1+z0)/(1+zrc); % Si Hap es real, como es el caso,
    podemos calcular su constante
    % de esta manera.
    subplot(2,2,2);
    zplane(Zap, Pap);
    title('Diagrama de ceros y polos H_{ap}(\omega)');
```

Teniendo ya definidos $H_{min}(z)$ y $H_{ap}(z)$, conseguir $H_c(z)$ es inmediato:

```
% Hc(z)
    Zc = Pmin;
    Pc = Zmin;
    Kmin = 1 / Kap;
    subplot(2,2,3);
    zplane(Zc, Pc);
    title('Diagrama de ceros y polos H_{c}(\omega)');
    [Bc, Ac] = zp2tf(Zc, Pc,Kmin);
    [Hc,wc] = freqz(Bc,Ac,4096);
    mHc = 20*log10(abs(Hc));
    subplot(2,2,4);
    plot(w, mHc);
    title('Modulo de Hc(w)');
    xlabel('\omega'); ylabel('dB'); grid on;
    axis([0, pi, min(mHc)-1, max(mHc)+1]);
```

Vemos que la gráfica del módulo es la que tenemos que replicar con $H_e(z)$, cosa que veremos cómo hacerla en la siguiente sección.

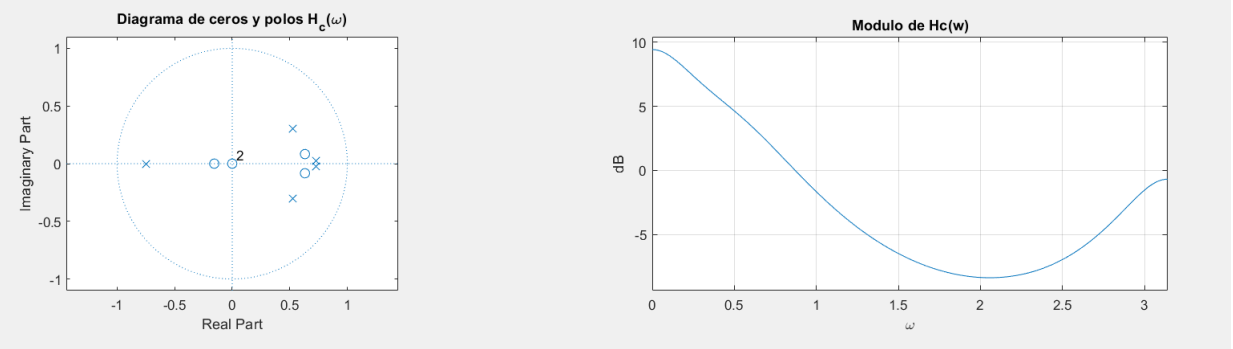

Figura 5: Función de transferencia de la distorsión de la sala $H_d(z)$ y diagrama de polos y ceros de la misma.

3.2. Construcción y obtención de la función de transferencia del sistema ecualizador $H_e(z)$

3.2.1. Definición de los filtros para la construcción del sistema compensador

Como ya hemos dicho, tenemos a nuestra disposición tres tipos de filtros: repisa paso bajo, repisa paso alto y pico paso banda. Estos filtros han de definirse y caracterizarse, cuyos parámetros principales son, aparte de los ya descritos (frecuencia de corte o central, ganancia y ancho de banda), los coeficientes A y B.

Para computarlos, hemos seguido la documentación proporcionada *Audio-EQ-Cookbook*, que describe de manera bastante clara cómo obtenerlos. Para cada filtro debemos de calcular los parámetros a_i y b_i, que lo que nos darán serán los coeficientes de los polos y ceros, respectivamente.

Cada filtro lo hemos definido como una nueva función, que después hemos integrado en el script principal, cuyo código es el siguiente:

```
function [Bl, Al] = repisa_paso_bajo(dBgain, f0, BW)
    Fs = 48000;
    A  = sqrt(10^(dBgain/20));
    w0 = 2*pi*f0/Fs;
    alpha = sin(w0)*sinh(log(2)/2 * BW * w0/sin(w0));

    b0 =    A*( (A+1) - (A-1)*cos(w0) + 2*sqrt(A)*alpha );
```

```
    b1 =  2*A*( (A-1) - (A+1)*cos(w0)                   );
    b2 =    A*( (A+1) - (A-1)*cos(w0) - 2*sqrt(A)*alpha );
    a0 =        (A+1) + (A-1)*cos(w0) + 2*sqrt(A)*alpha;
    a1 =   -2*( (A-1) + (A+1)*cos(w0)                   );
    a2 =        (A+1) + (A-1)*cos(w0) - 2*sqrt(A)*alpha;

    Bl = [b0 b1 b2];
    Al = [a0 a1 a2];
end
```

.....

```
function [Bp, Ap] = pico_paso_banda(dBgain, f0, BW)
    Fs=48000;
    A = sqrt(10^(dBgain/20));
    w0 = 2*pi*f0/Fs;
    alpha = sin(w0)*sinh(log(2)/2 * BW * w0/sin(w0));

    b0 =   1 + alpha*A;
    b1 =  -2*cos(w0);
    b2 =   1 - alpha*A;
    a0 =   1 + alpha/A;
    a1 =  -2*cos(w0);
    a2 =   1 - alpha/A;

    Bp = [b0 b1 b2];
    Ap = [a0 a1 a2];
end
```

.....

```
function [Bh, Ah] = repisa_paso_alto(dBgain, f0, BW)
    Fs = 48000;
    A  = sqrt(10^(dBgain/20));
    w0 = 2*pi*f0/Fs;
    alpha = sin(w0)*sinh(log(2)/2 * BW * w0/sin(w0));

    b0 =    A*( (A+1) + (A-1)*cos(w0) + 2*sqrt(A)*alpha );
    b1 = -2*A*( (A-1) + (A+1)*cos(w0)                   );
    b2 =    A*( (A+1) + (A-1)*cos(w0) - 2*sqrt(A)*alpha );
    a0 =        (A+1) - (A-1)*cos(w0) + 2*sqrt(A)*alpha;
    a1 =    2*( (A-1) - (A+1)*cos(w0)                   );
    a2 =        (A+1) - (A-1)*cos(w0) - 2*sqrt(A)*alpha;

    Bh = [b0 b1 b2];
    Ah = [a0 a1 a2];
end
```

.....

3.2.2. Utilización y representación de los filtros

Para representar y ajustar la respuesta en frecuencia de cada filtro seguiremos en todos una estructura similar:

1. Tantear y probar "a mano" cada uno de los parámetros del filtro para que la combinación de las respuestas en frecuencia de todos los filtros de la señal buscada.
2. Calcular y representar para cada caso la respuesta en frecuencia del filtro.

Para sacar los parámetros de entrada de cada filtro, lo que hemos hecho ha sido ir probando poco a poco de qué manera encajaba cada filtro con el siguiente para que cada uno afectase lo menos posible al otro. Esto se observa en el bajo ancho de banda ocupado por cada filtro y en las ganancias tan específicas de los mismos. Hay que recordar que las frecuencias están estandarizadas y siguen la norma ISO 266, cosa respetada en todo momento. Al final lo que buscábamos era mucha precisión: que el filtro construido diferiese en, como mucho, 0.5 dB del original.

Al final, el código y los resultados que obtenemos son los siguientes:

```
% Filtro repisa paso bajo
    [Bl,Al] = repisa_paso_bajo(9.2,1600,2);
    [Hl, wl] = freqz(Bl, Al, 4096);
    mHl = 20*log10(abs(Hl));
    figure(2);
    subplot(3,2,1);
    plot(wl, mHl);
    title('Módulo de la respuesta en frecuencia de H_{l}(z)');
    xlabel('\omega'); ylabel('dB'); grid on;
    axis([0, pi, min(mHl)-1, max(mHl)+1]);

% Filtro repisa paso alto
    [Bh,Ah] = repisa_paso_alto(-1.1,20000,0.3);
    [Hh, wh] = freqz(Bh, Ah, 4096);
    mHh = 20*log10(abs(Hh));
    subplot(3,2,2);
    plot(wh, mHh);
    title('Módulo de la respuesta en frecuencia de H_{h}(z)');
    xlabel('\omega'); ylabel('dB'); grid on;
    axis([0, pi, min(mHh)-1, max(mHh)+1]);

% Filtro pico paso banda 1
    [Bp1,Ap1]=pico_paso_banda(5.6,3150, 2.2);
    [Hp1, wp1]= freqz(Bp1, Ap1, 4096);
    mHp1= 20*log10(abs(Hp1));
    subplot(3,2,3);
    plot(wp1, mHp1);
    title('Módulo de la respuesta en frecuencia de H_{p1}(\omega)');
    xlabel('\omega'); ylabel('dB'); grid on;
    axis([0, pi, min(mHp1)-1, max(mHp1)+1]);

% Filtro pico paso banda 2
    [Bp2,Ap2]=pico_paso_banda(-1.6,10000,0.8);
    [Hp2, wp2]= freqz(Bp2, Ap2, 4096);
    mHp2= 20*log10(abs(Hp2));
    subplot(3,2,4);
    plot(wp2, mHp2);
    title('Módulo de la respuesta en frecuencia de H_{p2}(\omega)');
    xlabel('\omega'); ylabel('dB'); grid on;
    axis([0, pi, min(mHp2)-1, max(mHp2)+1]);

% Filtro pico paso banda 3
    [Bp3,Ap3]=pico_paso_banda(-8.6,16000,1.1);
    [Hp3, wp3]= freqz(Bp3, Ap3, 4096);
    mHp3= 20*log10(abs(Hp3));
    subplot(3,2,5);
    plot(wp3, mHp3);
```

```
title('Módulo de la respuesta en frecuencia de H_{p3}(\omega)');
xlabel('\omega'); ylabel('dB'); grid on;
axis([0, pi, min(mHp3)-1, max(mHp3)+1]);
```

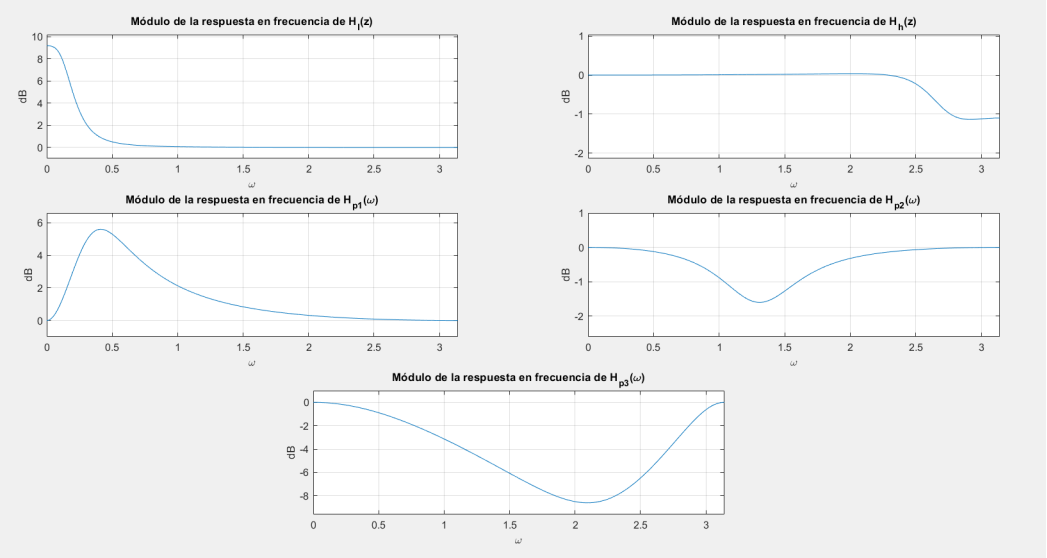

Figura 6: Función de transferencia de cada uno de los filtros usados.

3.2.3. Análisis y construcción del ecualizador paramétrico $H_e(z)$

Con estas funciones, ya podemos armar nuestra función ecualizadora. Simplemente, multiplicando en el dominio de la frecuencia cada función de transferencia correspondiente a los anteriores filtros, podemos obtener el sistema ecualizador. Esto es:

$$\boxed{H_e(z) = H_l(z) \cdot H_{p1}(z) \cdot H_{p2}(z) \cdot H_{p3}(z) \cdot H_h(z)} \tag{4}$$

Como resultado, obtenemos la siguiente gráfica. Esta gráfica intenta asemejarse lo máximo posible al módulo de la función $H_c(\omega)$, de tal manera que cuando la multipliquemos por $H_d(\omega)$ resulte en el sistema compensador. Hay que recordar que las pequeñas variaciones en el módulo dependen de que los filtros no son perfectos ni se amoldan de manera exacta a la forma de la señal, si no que solo se aproximan con el error antes definido. En la siguiente gráfica se puede ver que las variaciones de la función de transferencia están contenidas en ese rango.

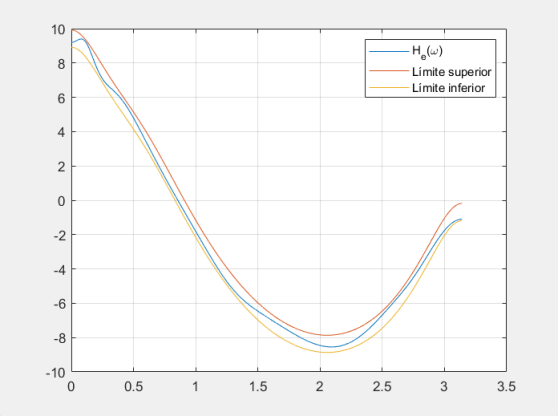

Figura 7: Módulo de la función de transferencia del sistema ecualizador $H_e(\omega)$

Su implementación en código es la siguiente:

```
% Filtro ecualizador parametrico discreto
    He = Hl  .* Hp1 .* Hp2 .*Hp3 .* Hh;
    mHe = 20*log10(abs(He));
    subplot(3,2,6);
    plot(wp3,mHe);
    title('Módulo de la respuesta en frecuencia de H_{e}(\omega)');
    xlabel('\omega'); ylabel('dB'); grid on;
    axis([0, pi, min(mHe)-1, max(mHe)+1]);
```

3.2.4. Implementación del ecualizador paramétrico $H_{eq}(z)$ sobre el sistema final

El filtro equalizador final se consigue multiplicando la función de transferencia de la distorsión de la sala por el sistema ecualizador.

$$\boxed{H_{eq}(z) = H_d(z) \cdot H_e z)} \tag{5}$$

El resultado de esta operación da lugar a una función de transferencia cuyo módulo es el siguiente:

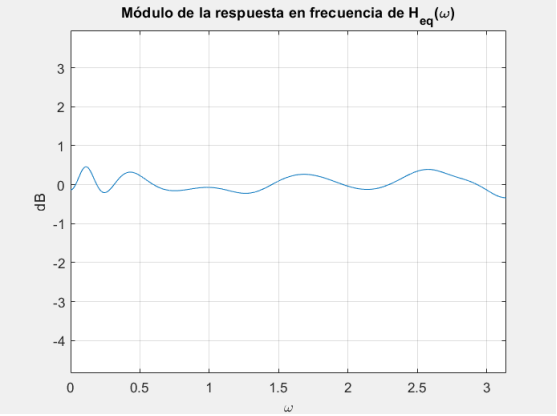

Figura 8: Módulo de la respuesta en frecuencia de $H_{eq}(\omega)$.

Y la implementación del código es simple:

```
%% Filtro final

    Heq = H .* He;
    mHeq = 20 * log10(abs(Heq));
    plot(w, mHeq);
    figure(4);
    title('Módulo de la respuesta en frecuencia de H_{eq}(\omega)');
    xlabel('\omega'); ylabel('dB'); grid on;
    axis([0, pi, min(mHeq)-3, max(mHeq)+5]);

    figure(3)
    plot(wp3, mHe);
    hold on
    grid on;
    plot(w, mHc+0.5);
    plot(w, mHc-0.5);
    legend('H_{e}(\omega)', 'Límite superior', 'Límite inferior')
    hold off
```

3.3. Sistema final y pruebas sobre la señal chirp

Por último, creamos una señal de prueba "chirp" que hará un barrido de las frecuencias deseadas. Para generarla en Matlab definiremos sus parámetros de la siguiente manera:

```
%% Señal de prueba chirp y montaje del sistema final

    fs = 48000; f0 = 20; f1 = 20000; t1 = 10; t = 0:1/fs:t1; % Parámetros de la señal chirp
    x = chirp(t, f0, t1, f1,'logarithmic'); % Creación de la señal de prueba
```

Se podrá comprobar que el sistema funciona correctamente cuando pasamos la señal de prueba "x" por los filtros de acuerdo a la figura 1. Por tanto, primero se pasará por el ecualizador paramétrico, es decir, los filtros repisa y los pico paso banda y luego, a través del sistema distorsionador.

```
    x1 = filter(Bl, Al, x);
    x2 = filter(Bp1, Ap1, x1);
    x3 = filter(Bp2, Ap2, x2);
    x4 = filter(Bp3, Ap3, x3);

    y = filter(Bh, Ah, x4);

    xc = filter(B, A, y);
```

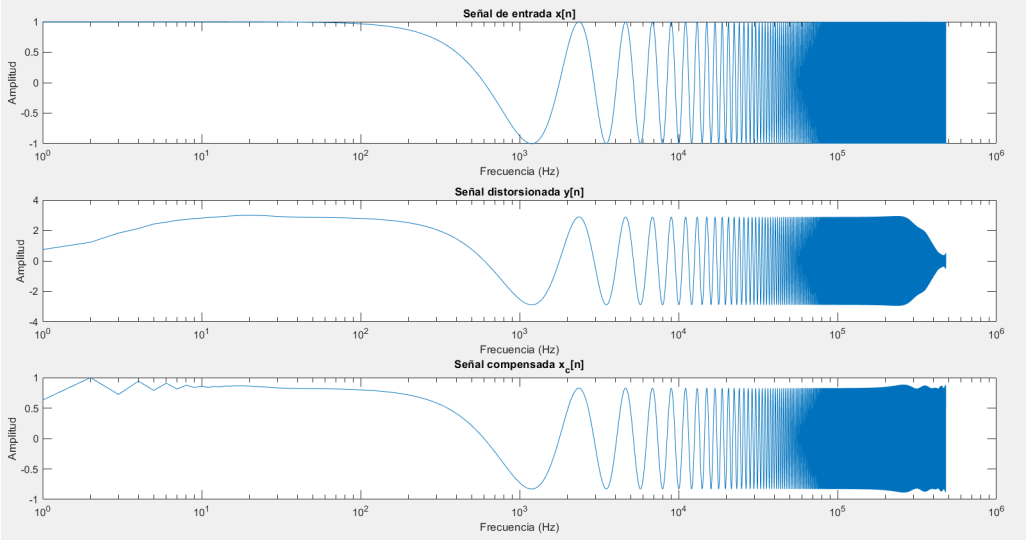

Figura 9: Señal de entrada, señal distorsionada y señal compensada

Las pequeñas variaciones de la señal compensada se deben a que el ecualizador no es perfecto, y su función de transferencia varía entre 0.5 y -0.5 dB de su valor original. A pesar de que es una aproximación muy buena, se puede apreciar una pérdida de calidad ínfima.

4. Conclusiones

En este trabajo hemos visto cómo diseñar un ecualizador paramétrico de audio que compense las distorsiones introducidas en una sala. Aunque en la realidad estas distorsiones sean difíciles de caracterizar y provengan de distintas fuentes, la aproximación con la que trabajamos nos permite analizarlos e idear un sistema compensatorio, como en el que en un futuro tendremos que crear.

También, nos permite conocer el funcionamiento de otros filtros no tan comunes y la función de transferencia resultante de la unión de estos filtros. Así como también la elección de sus parámetros de la mejor manera posible para que esta función sea lo más parecida a una de compensación ideal.