D/A 转换器的前置离散预矫正系统

3200103584 蔡松成

摘要：数字信号与模拟信号之间的转换十分重要，本文通过对数模转换方式之一的零阶保持进行频谱分析，发现了频率失真现象，并且着手在频域上构建离散因果系统，对信号在频域补偿，给出了几种切实可行的方法。

关键词：D/A 转换器 频谱失真 Fir 滤波器 零阶保持

一、研究背景

在现实生活中，我们所接触到的信号大多是连续的模拟信号，比如声音、温度等，而计算机只能处理数字信号，因此模数转换(Analog-to-digital)和数模转换(Digital-to-analog)技术使得计算机可以处理现实中的信号，这两种技术对于我们信号处理处理学科来说十分重要。因此数/模转换器 (digital-to-analog converter, DAC) 作为片上系统中连接数字信号与模拟信号的“桥梁”，已经广泛应用于图像处理、无线通信、卫星、测控系统、音频以及多媒体显示等诸多领域。

正如上文所说，数/模转换器的输入是数字信号，但是难以从现实中直接获取。我们在数模转换之前应该先对现实中的模拟信号进行采样，得到采样信号，即离散的数字信号。采样方法有多种，由于不是本文的重点，因此我们仅讨论理想采样后数字信号的 D/A 转换矫正系统。在获取离散信号后，D/A 转换器需要通过保持器来将离散信号转换为连续信号。本文研究的零阶保持器的作用是将采样信号转变为在两个连续采样瞬时之间保持常量的信号。在信号传递过程中，把第 nT 时刻的采样信号值一直保持到第(n+1)T 时刻的前一瞬时，把第(n+1)T 时刻的采样值一直保持到(n+2)T 时刻，依次类推，从而把一个脉冲序列变成一个连续的阶梯信号。零阶保持器的时域和频域表达式如下所示(T 为保持时间)：

$$\mathrm{h(t)} = \mathrm{u(t)} - \mathrm{t(t-T)}$$

$$\mathrm{H_0(j\omega)} = \mathrm{T}\frac{\sin(\omega \mathrm{T}/2)}{\omega \mathrm{T}/2} e^{-j\omega T/2}$$

从上面的频域表达式我们可以看到，零阶保持器是个非理想的低通滤波器，在有效频段内，滤波器增益会随着频率的下降而下降，因此造成了重建信号的频率失真，因此如何有效地对该系统进行频域补偿是本文研究的重点。

二、理想零阶保持与频谱失真现象

1. 理想情况的连续信号采样与信号重建

根据上文所写的理想采样公式 $\mathrm{xp(t)} = \sum_{\mathrm{n}=-\infty}^{\infty} \mathrm{xa(nT)\delta(t-nT)}$(xa 是输入模拟信号，T 是采样时间间隔)，我们可以得到采样信号 xp(t)，其频谱公式是

$$\mathrm{xp(j\omega)} = \frac{1}{\mathrm{T}} \sum_{\mathrm{m}=-\infty}^{\infty} \mathrm{xa(\omega - m\omega s)}$$

上式中$\omega s = \frac{2\pi}{T}$，即采样频率。采样信号的频谱仅仅是以原始输入信号为基带在频域上以ωs 的周期信号。因此我们仅需要一个理想低通滤波器对 xp(t)在频域上进行滤波即可。该理想滤波器频谱特性 H(jω)在$[-\omega s/2, \omega s/2]$上带通且增益为 T。公式如下所示

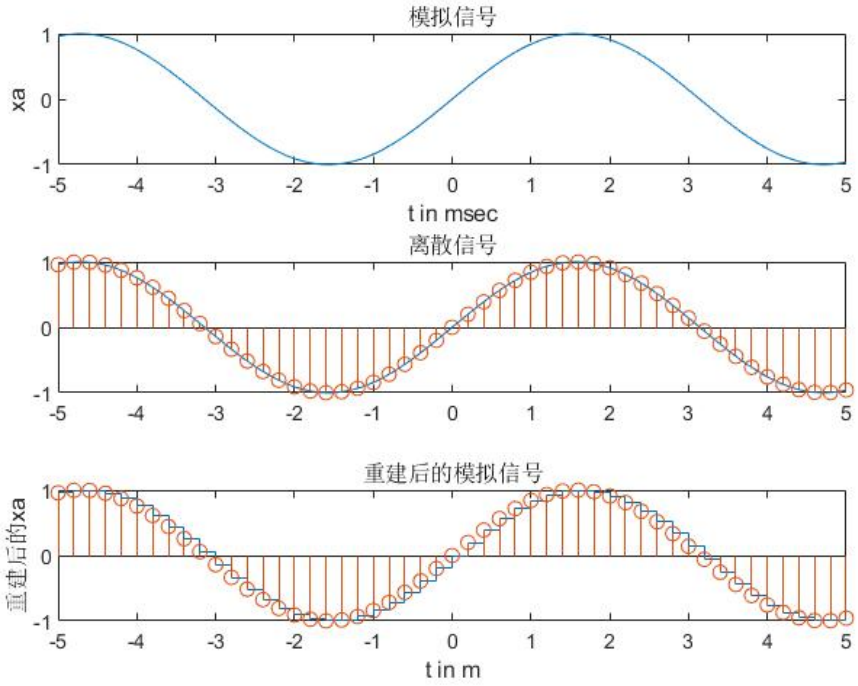
$$xr(j\omega) = xp(j\omega) * H(j\omega) = xa(j\omega)$$

2. 基于零阶保持的恢复系统

由于理想滤波器在实际系统上无法构建，因此我们考虑现实中的数模转换器，即零阶保持电路。零阶保持电路对信号的作用等效为如下公式：

$$\begin{cases} h(t) = u(t) - t(t-T) \\ xr(t) = xp(t) * h(t) = \sum_{n=-\infty}^{\infty} xa(nT)[u(t-NT) - u(t-(n+1)T) \end{cases}$$

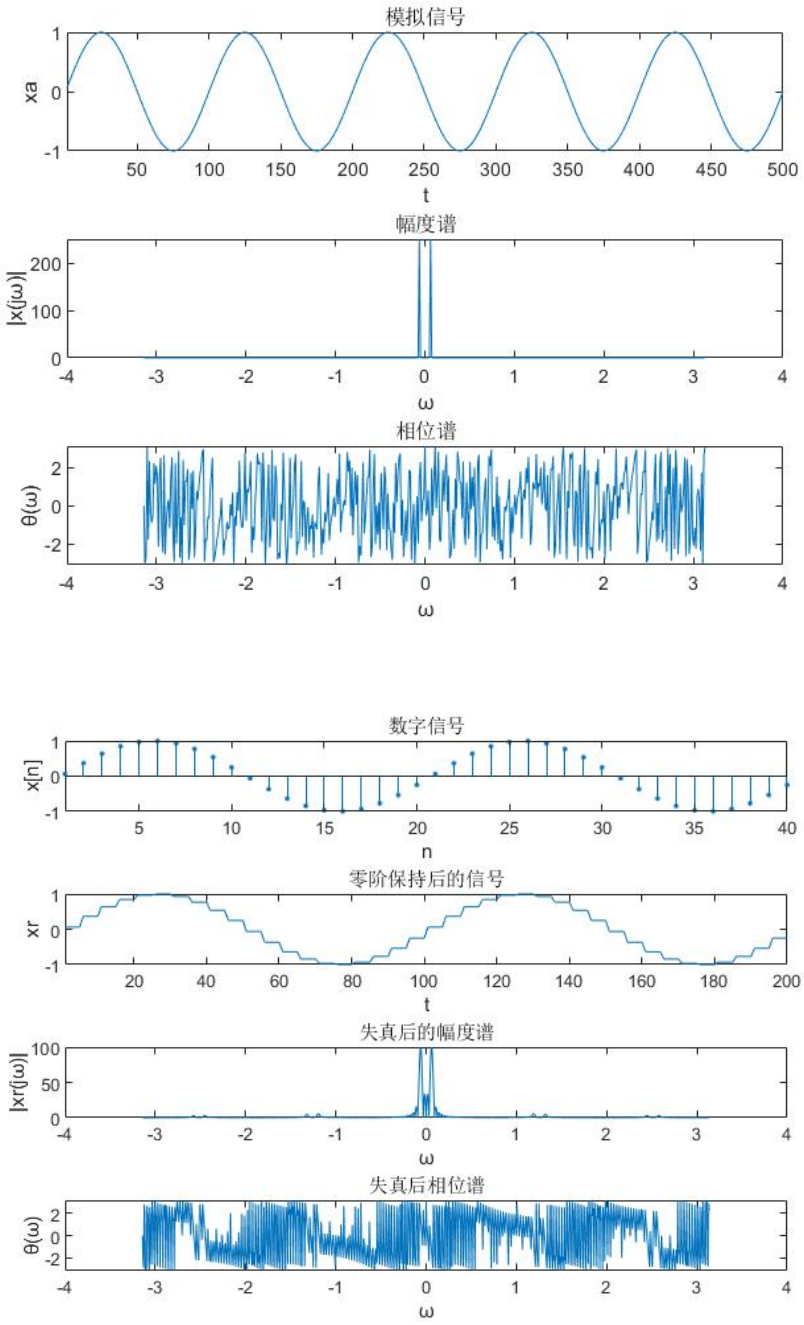
在时域上，我们可以把它理解为一个离散信号在第 n 时刻的采样信号值一直保持到第(n+1)时刻，在离散信号值中间加上一个矩形。它在时域上的近似效果如下图所示(程序 showzero.m)



根据零阶保持的特点，保持时间 T 越小，重建效果越好。可以观察到保持时间 T = 0.2s 的条件下，重建的信号依然是和模拟信号有很大差别。

从频谱上来看，根据研究背景的频域表达式，零阶保持电路构成了一个非理想的低通滤波器，虽然它在高频部分的增益接近于 0，能有效地过滤掉基带信号的镜像和谐波，但是在希望完全恢复的频段，零阶保持器无法正确恢复，从而导致频谱失真，具体效果如下图所示(程序 distortion.m)。

由于 matlab 仿真过程中无法模拟出真实的模拟信号，因此我们用采样频率远大于数字信号的数字信号来仿真模拟信号，因此在放大图后我们观察到模拟信号的幅度谱是两个对称三角形，而数字信号的幅度谱在频域多了些失真的部分，我们的任务是需要通过数学模型构建系统，达到理想滤波器的效果，去除这样的失真，从而达到矫正的目的。



三、离散预矫正系统的构建

1. 从连续预矫正系统到离散预矫正系统

由于我们处理的信号是 xp(t)（在零阶保持之前），因此我们从连续预矫正系统着手进行如下处理。xp(t)是理想采样信号，Ho(jω)是零阶保持的系统函数，Hr(jω)是我们构建的连续时间系统，xr(t)是重建后的信号。考虑到 xp(t)前面有个$\frac{1}{T}$的系数，因此 Hr(jω)无需再乘$\frac{1}{T}$。具体公式如下：

$$\begin{cases} Ho(j\omega) = T\dfrac{sin(\omega T/2)}{\omega T/2} \\ Hr(j\omega) = \dfrac{\omega T/2}{sin(\omega T/2)}(|\omega| < \omega s) \end{cases}$$

$$\mathrm{H(j\omega)} = \mathrm{Ho(j\omega)} * \mathrm{Hr(j\omega)} = \begin{cases} \mathrm{T} & (|\omega| < \omega s) \\ 0 & \text{其他} \end{cases}$$

从上式可得$Ho(j\omega)$和$Hr(j\omega)$级联后等效为一个理想滤波器, 因此信号可得到精确恢复。但是$Hr(j\omega)$是非因果系统，并且频率响应两边有一定上翘，在实际设计上几乎无法实现。于是我们就对上述系统进行离散化处理，得到如下系统

其中 $\mathrm{x[n] = xa(nT)}$是采样后的离散信号，$\mathrm{Hr}(e^{j\omega}) = \mathrm{Hr(j}\frac{\omega}{\mathrm{T}}) = \frac{\mathrm{w/2}}{\mathrm{sin(wT/2)}}(|\omega| < \omega sT)$,级联后的系统仍是个理想滤波器。

2. $\mathrm{Hr}(e^{j\omega})$系统设计

具有有限时宽的冲激响应的滤波器称为 Fir 滤波器，因果 Fir 滤波器是稳定的，非因果滤波器也能通过延时的方法变成因果 Fir 滤波器。因此我们采用 Fir 滤波器的设计方法来设计数字滤波器 $\mathrm{Hr}(e^{j\omega})$系统。

①频率采样法设计

（1）确定连续频率响应的函数 $\mathrm{Hr(j\omega)}$

$$Hr(j\omega) = \frac{\omega T/2}{sin(\omega T/2)}(|\omega| < \omega s)$$

（2）对$Hr(j\omega)$进行等间隔采样得到

$$\mathrm{Hr(m)} = \mathrm{H}(e^{j\omega})|\omega = \frac{2\pi \mathrm{m}}{\mathrm{TN}} = \frac{\pi \mathrm{m/N}}{\mathrm{sin}(\pi \mathrm{m/N})}$$

（3）根据频域函数进行傅里叶反变换推 h[n]

假设一共采样了 2M+1 个点，且 $2\mathrm{M} \gg 1$ 傅里叶逆变换如下式

$$\mathrm{h[n]} = \frac{1}{2\mathrm{M}} \sum_{\mathrm{m=-M}}^{\mathrm{M}} \frac{\mathrm{m}\pi/2\mathrm{M}}{\mathrm{sin}(\mathrm{m}\pi/2\mathrm{M})} e^{j\omega\pi/M}$$

由于 $H(m) = H(-m)$上式可以改写为

$$h[n] = \frac{1}{M}\left[\frac{1}{2} + \sum_{m=1}^{M} \frac{m\pi/2M}{\sin(m\pi/2M)}\cos(mn\pi/M)\right]$$

（4）对 h[n]进行离散傅里叶变换

利用 $H(e^{j\omega}) = H(j\frac{\omega}{T})$和 $h[n] = h(-n)$的性质，我们可以得到

$$H'r(j\omega) = h(0) + 2\sum_{n=1}^{N} h[n]\cos n\omega T$$

虽然 h[n]是非因果系统，但是从仿真中我们可以看到 h[n]的边上的值接近于 0，因此我们可以截取中间一段的 h[n]，并且在时域上对它进行延时处理，这样的操作对于频谱幅度没有影响，仅仅产生了个线性相位。

②窗函数设计法

（1）给定 $H(e^{j\omega})$，通过离散傅里叶逆变换求 h[n]，具体公式和频率采样法一致

（2）根据允许的过度带宽要求选择窗函数形状以及滤波器长度 N

（3）按照窗函数要求求 hw[n] = h[n]w[n](*w[n]是窗函数的时域表达式)

（4）计算 hw[n]的离散傅里叶变换，验证是否和要求符合

3. Ho(jw)系统设计

Padé近似法是 1892 年法国数学家 Padé提出的一种著名的有理近似方法，后人将其命名为 Padé近似方法,其中 $e^{-Ts} \approx \frac{1-Ts/2+p1(Ts)^2-p2(Ts)^3+\cdots+(-1)^{n+1}(Ts)^{n+1}}{1+Ts/2+p1(Ts)^2+p2(Ts)^3+\cdots+pn(Ts)^{n+1}} \triangleq \frac{ne}{de}$，将式子代入传递函数得

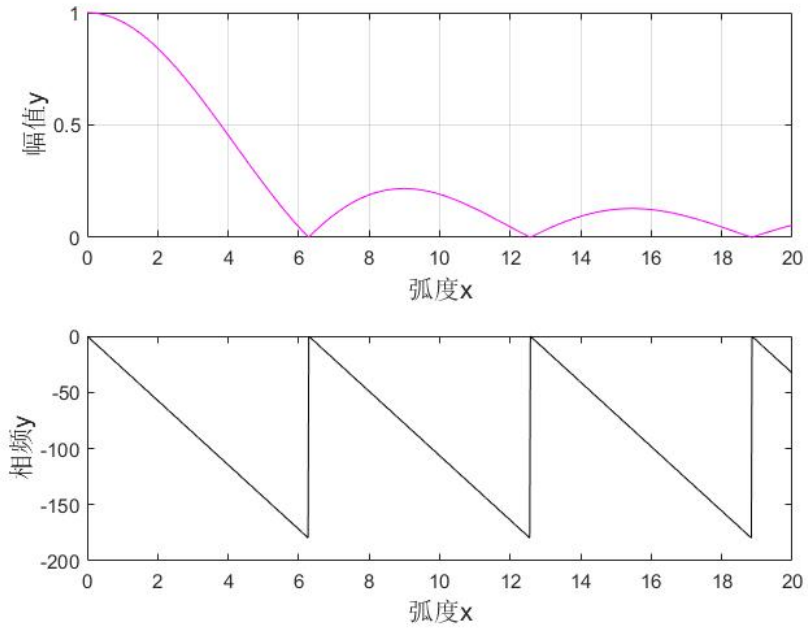
$$Ho(jw) = \frac{de - ne}{sde} \triangleq \frac{num}{den}$$

分母 den 是 s 和 de 的乘积，分子 num 是 de 与 ne 之差。

由于 matlab 只能处理有限长的序列，所以难以计算连续的 h(t)的频谱。因此上式在 matlab 仿真中可用 pade()函数得到较为准确的零阶响应传递函数幅度和频谱。

四、Matlab 仿真

1. 仿真 Ho(jω)的频谱和相位

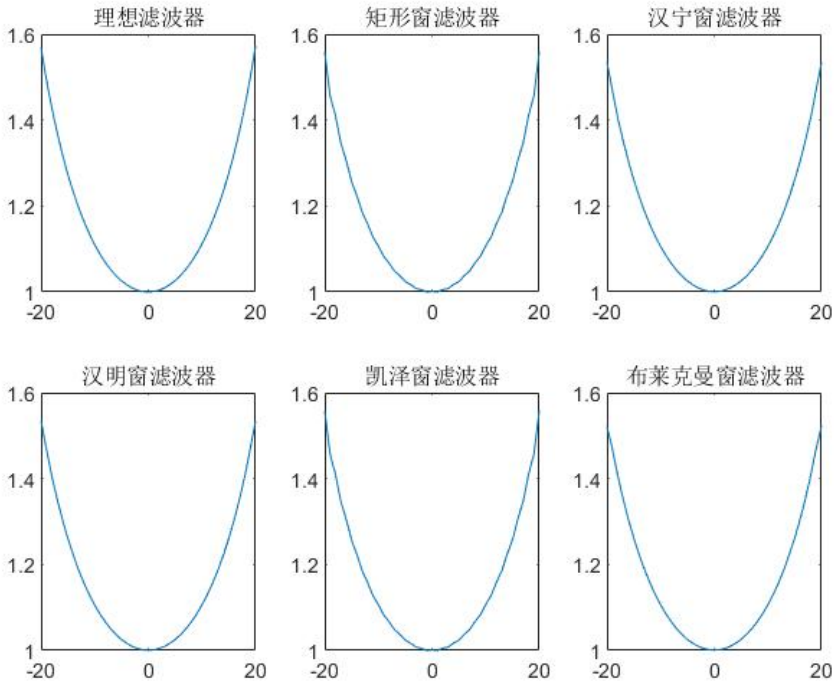


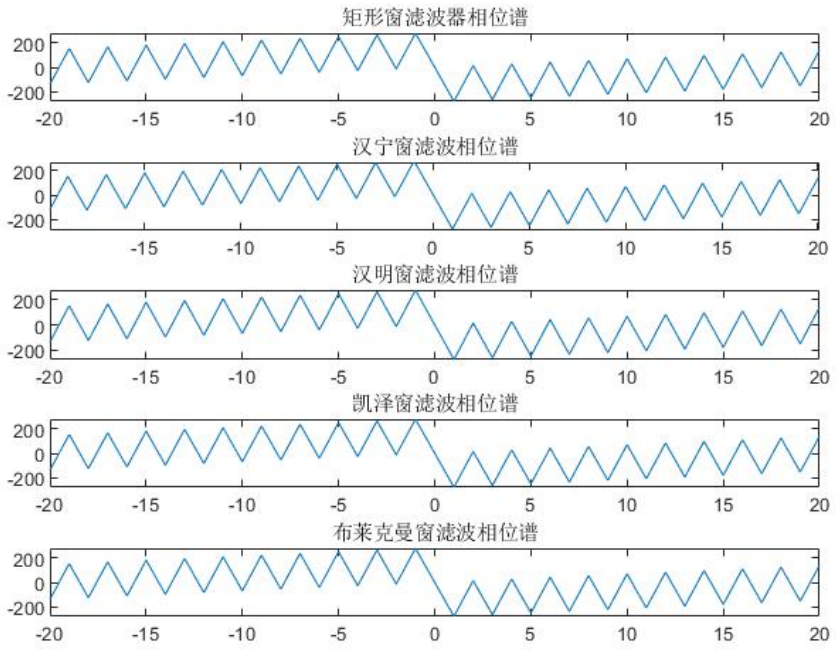
从上图我们观察到，Ho(jω)的频谱幅值随频率的增大衰减速度很快，因此可以认为零阶保持器基本上近似一个低通滤波器；从相频特性来看，零阶保持器存在相角滞后的问题，最大滞后角度为 180°，相角滞后越大，系统的稳定性也就越来越差。

2.基于频率采样法和窗函数设计法的 Fir 滤波器设计

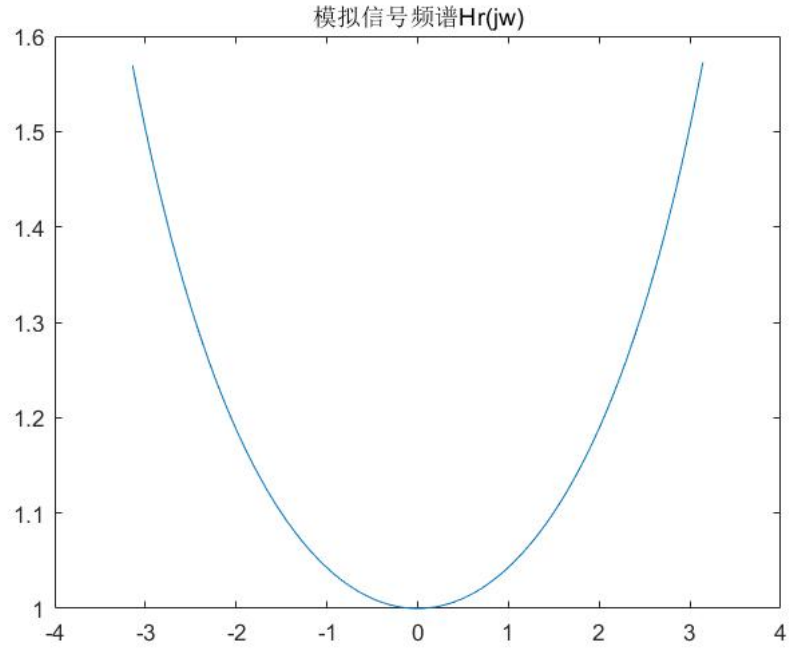
①窗函数法：

Matlab 提供了几种窗函数的函数，例如 hanning(),hamming(),kaiser()等等。具体实现在程序 windows_filter.m 中。





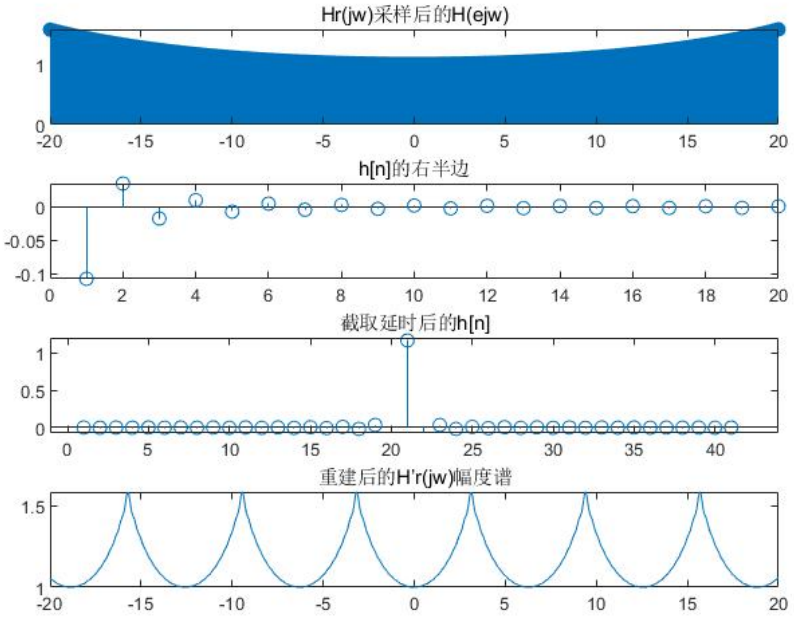
在程序中，我们设定每个窗函数的长度为 41，在求得 h[n]的基础上进行加窗，完成了预期的效果，与模拟信号 Hr(j ω)的频谱相比近似，同时看到它们的相位是线性的，对最后的信号重建没有影响。模拟信号频谱图如下图所示



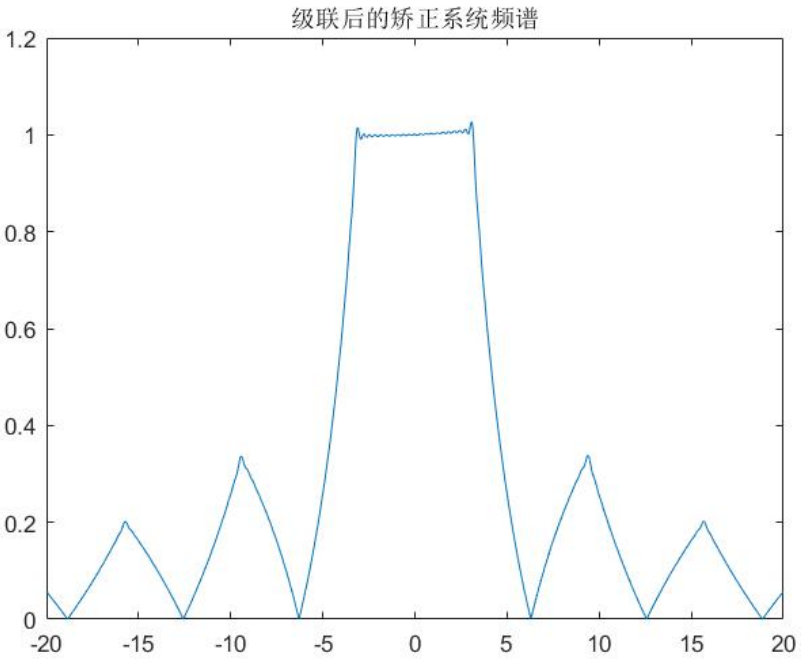
②频率采样法：

根据上文分析的公式，我们先对已知的 Hr(jw)进行采样获得了下图第一张图的频谱序列 $H(e^{j\omega})$，然后在对该序列进行傅里叶逆变换。由于得到的时域离散信号在离 0 越远的的地方无限接近于 0，我们对该时域信号进行截取和延时，在进行傅里叶变换后，根据

$H(e^{j\omega}) = Hr(j\omega)$的性质获得重建后的频谱即我们需要设计的 H'r(jω)。



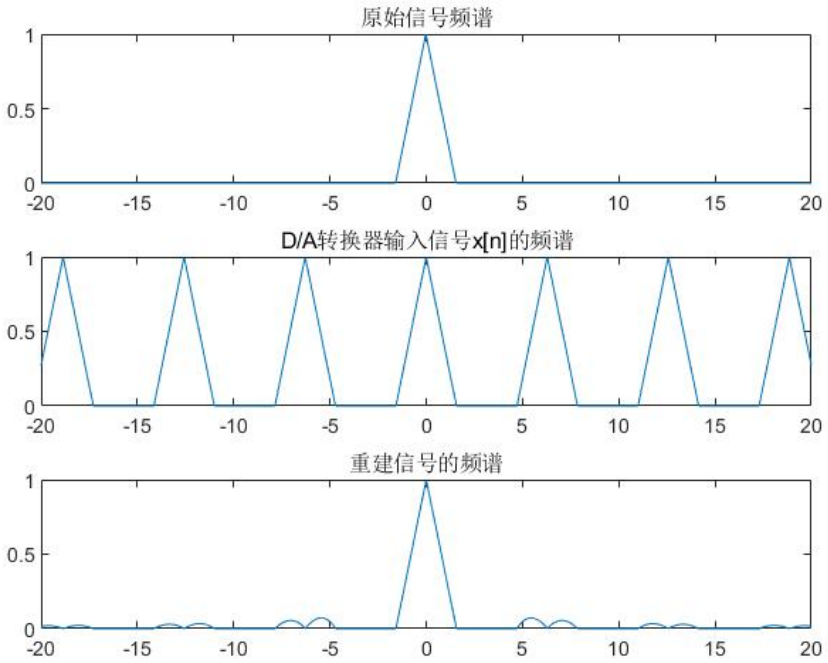
4. 离散预矫正系统频域仿真



我们取保持时间 T = 1，因此$\omega s = \frac{2\pi}{T} = 2\pi$，所以信号的基带信号频域区间为$[-\pi, \pi]$，仿真出来的系统函数在$[-\pi, \pi]$接近于 1，在其他频段都呈现衰减效果明显，我们可以近似为理想低通滤波器，在$-\pi$和π附近频域值上下波动较为剧烈，是因为吉布斯现象产生的。

5. 验证矫正系统效果

为了验证系统输出效果，我们用频谱为 $x(jw) = 1 - \frac{2}{\pi}|\omega|$的信号作为输入，在通过与系统函数相乘得到的频谱与原信号极为接近，证明了该系统的可行性。



五、Matlab 程序

1. 程序 showzero

```
% Analog signal
Dt = 0.00005;
t = - 0.005:Dt:0.005;
xa = sin(1000*t);

subplot(3,1,1);
plot(1000*t,xa);
title('模拟信号');
xlabel('t in msec');
ylabel('xa');

%Fs = 5000,Ts = 0.0002
% Discrete-time signal
Ts = 0.0002;
Fs = 1/Ts;
n = -25:25;
nTs = n*Ts;
x = sin(1000*nTs);
```

```
subplot(3,1,2)
plot(1000*t,xa);
hold on
stem(n*Ts*1000,x);
title('离散信号');
hold off

% Analog signal reconstruction
subplot(3,1,3);
stairs(nTs*1000,x);
title('重建后的模拟信号');
xlabel('t in m');
ylabel('重建后的 xa');
hold on
stem(n*Ts*1000,x)
hold off
```

2. 程序 distortion

```
clear;
f0=10000; %用来模拟模拟信号的数字信号的采样频率 fs<<f0
f=[100];%f 是模拟信号的频率表 max(f)<250;
fs=2000; %信号的采样频率
N=200;%数字信号的样点数
%模拟信号的生成
s=signal_generate(f,f0,N);
figure(1)
subplot(3,1,1);plot(s);axis([1 N min(s) max(s)]);title('模拟信号
');xlabel('t');ylabel('xa');
[w,y] = mycalculateDiscreteFourierTransform(s,2*pi);
subplot(3,1,2);plot(w,abs(y)),title('幅度谱');xlabel('ω
');ylabel('|x(jω)|');
subplot(3,1,3);plot(w,angle(y)),title('相位谱');xlabel('ω
');ylabel('θ(ω)');
%采样点数，间隔的计算
deltaN = f0/fs;
Ns = N/deltaN;
%采样
for i=1:Ns
sd(i)=s((i-1)*deltaN+1);
end

Np =1000;
%恢复出方波信号
```

```
sp=[];
for i=1:Ns
 sp=[sp sd(i)*ones(1,deltaN)];
end
figure(2);
subplot(4,1,1);stem(sd,'.');axis([1 Ns min(s) max(s)]);title('数字信号');xlabel('n');ylabel('x[n]');
subplot(4,1,2);plot(sp);axis([1 N min(s) max(s)]);title('零阶保持后的信号');xlabel('t');ylabel('xr');
sp1 = [sp,zeros(1, Np-length(sp))];
[w1,y1] = mycalculateDiscreteFourierTransform(sp1,2*pi);
subplot(4,1,3);plot(w1,abs(y1)),title('失真后的幅度谱');xlabel('ω');ylabel('|xr(jω)|');
subplot(4,1,4);plot(w1,angle(y1)),title('失真后相位谱');xlabel('ω');ylabel('θ(ω)');
```

3. 程序 zero

```
clc;
clear;
M=1024;
m = 2*M+1;
dw = 40/m;
w = 0:dw:20;
T = 1;%保持时间
[ne,de] = pade(T,15);%求传递函数系数
num = de -ne;
den = conv([1 0],de);%卷积
g = freqs(num,den,w);%求频谱
%求幅度和相角
x = abs(g);
y = angle(g);
subplot(2,1,1);plot(w,x,'m-');
xlabel('弧度 x','fontsize',12);ylabel('幅值 y','fontsize',12);
grid;subplot(2,1,2);
plot(w,y*180/pi,'k-');
xlabel('弧度 x','fontsize',12);ylabel('相频 y','fontsize',12);
```

4. 程序 windows_filter

```
M=1024;
m = 2*M+1;
dw = 40/m;
```

```
w = -20:dw:20;
for i = 1:m+1
    if i == floor(m/2)
        hd(i) = 1;
    else
        hd(i) = (pi*(i-M)/m)/(sin(pi*(i-M)/m));
    end
end
subplot(2,3,1);
plot(w,hd);
title('理想滤波器');

n = 20;
N = -20:1:20;
hrl = zeros(1,n);
hrr = zeros(1,n);
for i = 1:n
    hrl(i) = 1/(2*M);
    for j = 1:M
        hrl(i) = hrl(i)+ (1/M)*(j*pi/(2*M))*cos(j*i*pi/M)/
sin(j*pi/(2*M));
    end
end

for i = 1:n
    hrr(i) = hrl(n-i+1);
end
hmid = 1/(2*M);
for i = 1:M
    hmid = hmid +1/M*(i*pi/(2*M))/sin((i*pi/(2*M)));
end
hr = [hrr hmid hrl];

%加矩形窗
rewindow = ones(1,length(N));
h = hr.*rewindow;
[w1,y1] = mycalculateDiscreteFourierTransform(h,length(w));
subplot(2,3,2);
plot(N,abs(y1));
title('矩形窗滤波器幅度谱')

%加汉宁窗
hannwindow = hanning(length(N))';
h = hr.*hannwindow;
```

```
[w1,y2] = mycalculateDiscreteFourierTransform(h,length(w));
subplot(2,3,3);
plot(N,abs(y2));
title('汉宁窗滤波幅度谱');

%加汉明窗
hammwindow = hamming(length(N))';
h = hr.*hammwindow;
[w1,y3] = mycalculateDiscreteFourierTransform(h,length(w));
subplot(2,3,4);
plot(N,abs(y3));
title('汉明窗滤波器幅度谱');

%加凯泽窗
kawindow = kaiser(length(N))';
h = hr.* kawindow;
[w1,y4] = mycalculateDiscreteFourierTransform(h,length(w));
subplot(2,3,5);
plot(N,abs(y4));
title('凯泽窗滤波器幅度谱');

%加布莱克曼窗
bawindow = blackman(length(N))';
h = hr.* bawindow;
[w1,y5] = mycalculateDiscreteFourierTransform(h,length(w));
subplot(2,3,6);
plot(N,abs(y5));
title('布莱克曼窗滤波器幅度谱');
%画相位谱
figure(2);
subplot(5,1,1);
plot(N,90*angle(y1));
title('矩形窗滤波器相位谱')
subplot(5,1,2);
plot(N,90*angle(y2));
title('汉宁窗滤波相位谱');
subplot(5,1,3);
plot(N,90*angle(y3));
title('汉明窗滤波相位谱');
subplot(5,1,4);
plot(N,90*angle(y4));
title('凯泽窗滤波相位谱');
subplot(5,1,5);
plot(N,90*angle(y4));
```

```
title('布莱克曼窗滤波相位谱');
```

5. 程序 continuous_filter

```
dw = 2*pi/(2*1000+1);%采样频率取极大值模拟模拟信号
w = -pi:dw:pi;%区间
w0 = floor(length(w)/2);
hr = zeros(1,length(w));
for i = 1:length(w)
    if i == w0
        hr(i) = 1;
    else
        hr(i) = (1/2*(i*dw-pi))/sin(1/2*(i*dw-pi));%求 hr(jw)
    end
end
plot(w,hr);
title('模拟信号频谱 Hr(jw)');
```

6．程序 Digital_filter

```
clear;
M=1024;%为了能简化公式，M 取较大值
m = 2*M+1;%长度
dw = 40/m;%间距
w = -20:dw:20;
for i = 1:m+1
    if i == floor(m/2)
        hd(i) = 1;
    else
        hd(i) = (pi*(i-M)/m)/(sin(pi*(i-M)/m));
    end
end
subplot(4,1,1);
stem(w,hd);
title('Hr(jw)采样后的 H(ejw)')

n = 20;
N = -20:1:20;%h[n]的区间
hrl = zeros(1,n);
hrr = zeros(1,n);
for i = 1:n
    hrl(i) = 1/(2*M);
    for j = 1:M
```

```
       hrl(i) = hrl(i)+ (1/M)*(j*pi/(2*M))*cos(j*i*pi/M)/
sin(j*pi/(2*M));%傅里叶逆变换的近似公式
    end
end
subplot(4,1,2);
stem(hrl);
title('h[n]的右半边');
for i = 1:n
    hrr(i) = hrl(n-i+1);
end
hmid = 1/(2*M);
for i = 1:M
    hmid = hmid +1/M*(i*pi/(2*M))/sin((i*pi/(2*M)));
end
hr = [hrr hmid hrl];%获得双边的 h[n]
subplot(4,1,3);
stem(hr);
title('截取延时后的 h[n]');

T = 1;
for i = 1:length(w)
   y(i) = hr(n+1);
   for j = 1:n
      y(i) = y(i)+ 2*hr(j+n+1)*cos(j*(i*dw-20)*T);%离散傅里叶变换的近
似公式
   end
end
subplot(4,1,4);plot(w,abs(y));title('重建后的 H’r(jw)幅度谱');
```

7. 程序 cascade

```
M=1024;
m = 2*M+1;
dw = 40/m;
w = 0:dw:20;
T = 1;%保持时间
%获取 Ho(jw)
[ne,de] = pade(T,15);
num = de -ne;
den = conv([1 0],de);
g = freqs(num,den,w);
g = [fliplr(g) g];
w1 = -20:dw:20;
```

```
x = abs(g);
y = angle(g);
%获取 Hr(jw)
for i = 1:m+1
    if i == floor(m/2)
        hd(i) = 1;
    else
        hd(i) = (pi*(i-M)/m)/(sin(pi*(i-M)/m));
    end
end
n = 20;
N = -20:1:20;
hrl = zeros(1,n);
hrr = zeros(1,n);
for i = 1:n
    hrl(i) = 1/(2*M);
    for j = 1:M
        hrl(i) = hrl(i)+ (1/M)*(j*pi/(2*M))*cos(j*i*pi/M)/
sin(j*pi/(2*M));
    end
end
for i = 1:n
    hrr(i) = hrl(n-i+1);
end
hmid = 1/(2*M);
for i = 1:M
    hmid = hmid +1/M*(i*pi/(2*M))/sin((i*pi/(2*M)));
end
hr = [hrr hmid hrl];
T = 1;
for i = 1:length(w1)
   y(i) = hr(n+1);
   for j = 1:n
      y(i) = y(i)+ 2*hr(j+n+1)*cos(j*(i*dw-20)*T);
   end
end
%级联
z = y.*x;
plot(w1,z);
title('级联后的矫正系统频谱');
```

8. 程序 example

```
clear;
```

```
M=1024;
m = 2*M+1;
dw = 40/m;
w = -20:dw:20;%区间
lenw = length(w);
T = 1;%保持时间
origin = zeros(1,lenw);
for i = 1:lenw
    if i*dw-20<pi/2 && i*dw-20>-pi/2
    origin(i) = 1-2/pi*abs(i*dw-20); %构建原始信号频谱
    end
end
subplot(3,1,1);
plot(w,origin);
title('原始信号频谱');
time = floor(20/pi);%区间内重复次数
j = 1;
for i = 1:lenw
   if  i*dw-20<pi && i*dw-20>-pi
        trans(j) = origin(i);
        j = j+1;
   end
end
left = (lenw - 5*length(trans))/2;
trans1 = [trans(end-219:end) trans trans trans trans trans
trans(1:220)];%获得离散信号频谱
subplot(3,1,2);
plot(w,trans1);
title('D/A 转换器输入信号 x[n]的频谱');
z = Pre_correction_system();%产生系统函数
result = z.* trans1;%频域相乘，时域卷积
subplot(3,1,3);
plot(w,result);
title('重建信号的频谱');
```