Design of IIR and FIR Filters

[Objectives]

(1) Master the principles to design IIR filters.

(2) Master the method to design IIR filters with the requirements according to the principles.

(3) Improve students' abilities of self-study, finding, analyzing, and solving problems.

[Topics to discuss]

1. Design a low pass IIR DF with the requirements:

$\Omega_p=0.2613\pi$, $\Omega_s=0.4018\pi$, $A_p=0.75$dB, $A_s=20$dB.

(1) Calculate the minimum order of Butterworth that meets the requirements.

(2) Calculate the minimum order of Chebyshev I that meets the requirements.

(3) Give a bandlimited signal by yourself, let it pass through (1) and (2) above filter systems. Compare the input and output respectively. And discuss the differences between the two filters in the structure and performance.

Firstly, theoretical analysis to calculate the order variance between BW filter and CB I filter:

Converting the digital requirements to analog ones:

$$w_p=\frac{\Omega_\mathrm{p}}{T}=\frac{0.2613\pi}{T}, w_s=\frac{\Omega_\mathrm{s}}{\mathrm{T}}=\frac{0.4018\pi}{T}, A_p\le 0.75dB, As\ge 20dB$$

The minimum order of Butterworth:

$$N\ge ceil\left(\frac{\lg\left(\frac{10^{0.1A_s}-1}{10^{0.1A_p}-1}\right)}{2\lg\left(\frac{w_s}{w_p}\right)}\right)=ceil(7.28)=8\ (7\text{ in MATLAB}) \qquad 1-(1)$$

The minimum order of Chebyshev I:

$$N \geq ceil\left(\frac{\operatorname{arccosh}\frac{\sqrt{10^{0.1A_s}-1}}{\sqrt{10^{0.1A_p}-1}}}{\operatorname{arccosh}\left(\frac{w_s}{w_p}\right)}\right) = ceil(3.84) = 4 \qquad 1-(2)$$

Then, using MATLAB to simulate these two kinds of filter:

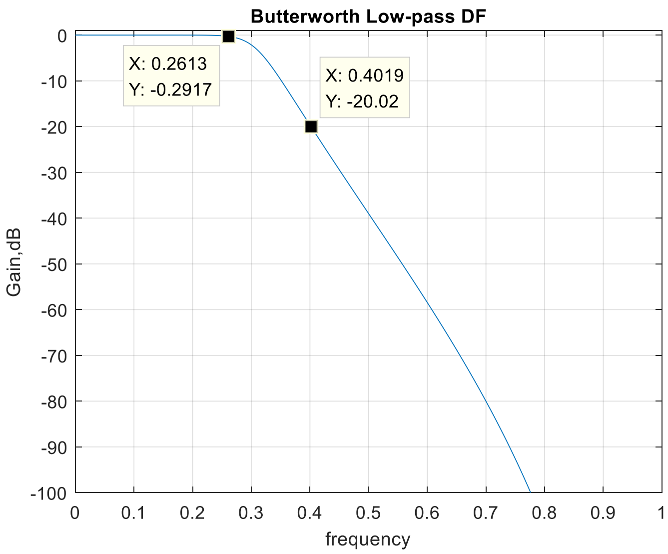


Figure 1 Frequency Response of Butterworth Digital Filter

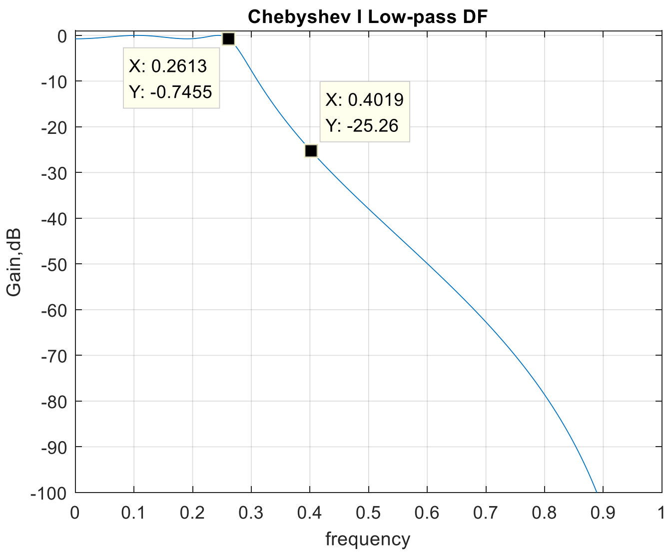


Figure 2 Frequency Response of Chebyshev I Digital Filter

Figures 1 and 2 show the frequency response of Butterworth and Chebyshev-I digital filters. The sampling frequency for both filters is determined as 1K Hz and both filters use the method of Bilinear Transformation. As a consequence, the order and cut-off frequency of the former one are 7 and 1.05K Hz, respectively, whereas the latter one is 4 and 870 Hz. In order to test their performances, we suppose a bandlimited signal shown in figure 3, which will pass through these two filters respectively.

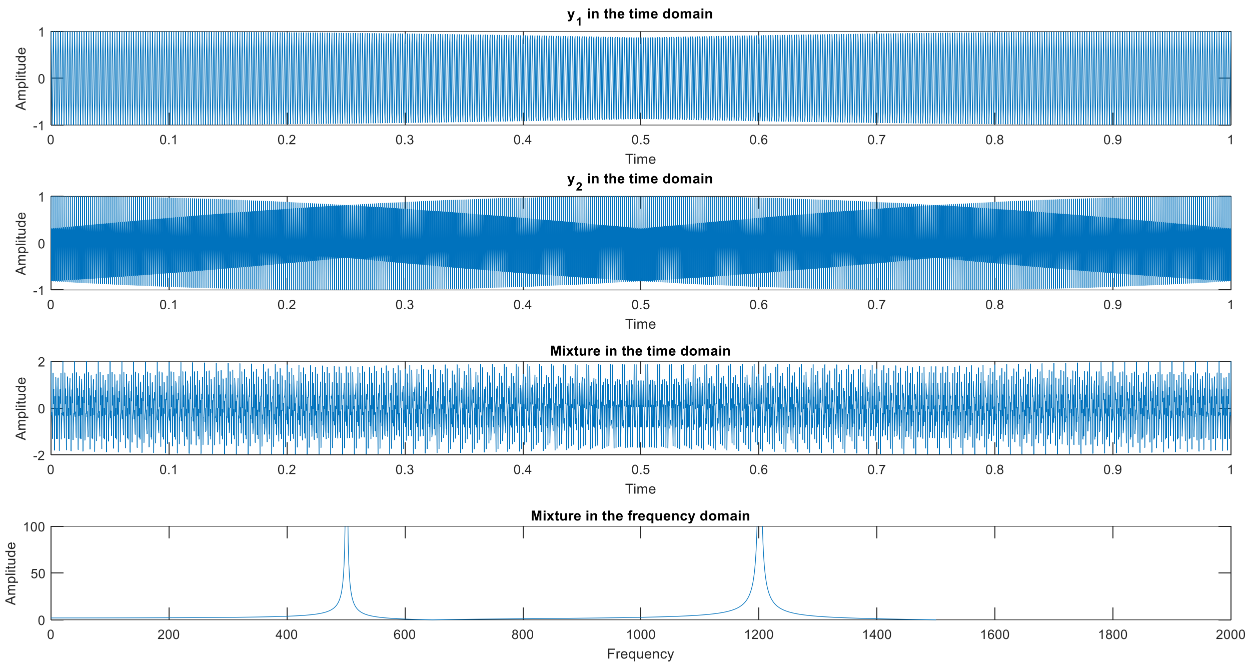


Figure 3 Bandlimited Testing Signal

The bandlimited testing signal in the above figure consists of two various cosine signals with frequencies of 500 Hz and 1200 Hz, and the sampling frequency is 3K Hz.

For the Butterworth digital filter, the filtered signal is shown in figure 4, whereas the filtered signal for Chebyshev-1 is shown in figure 5. The comparison between them is given in figure 6

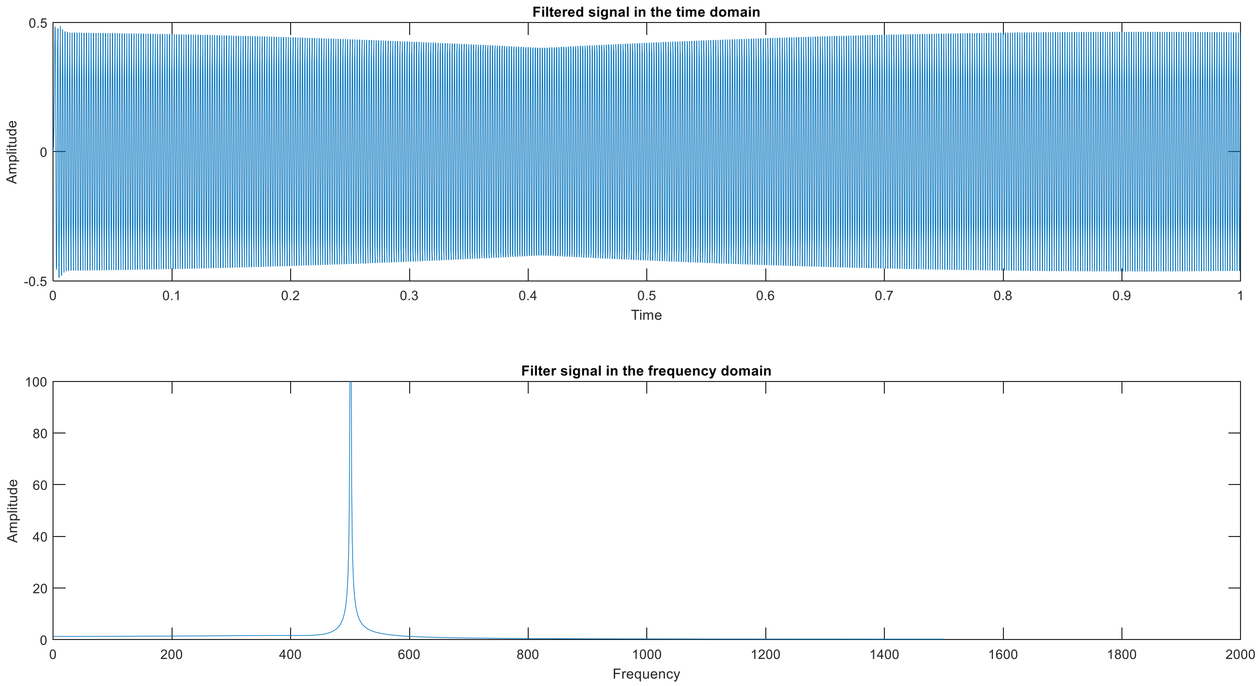


Figure 4 Filtered by Butterworth

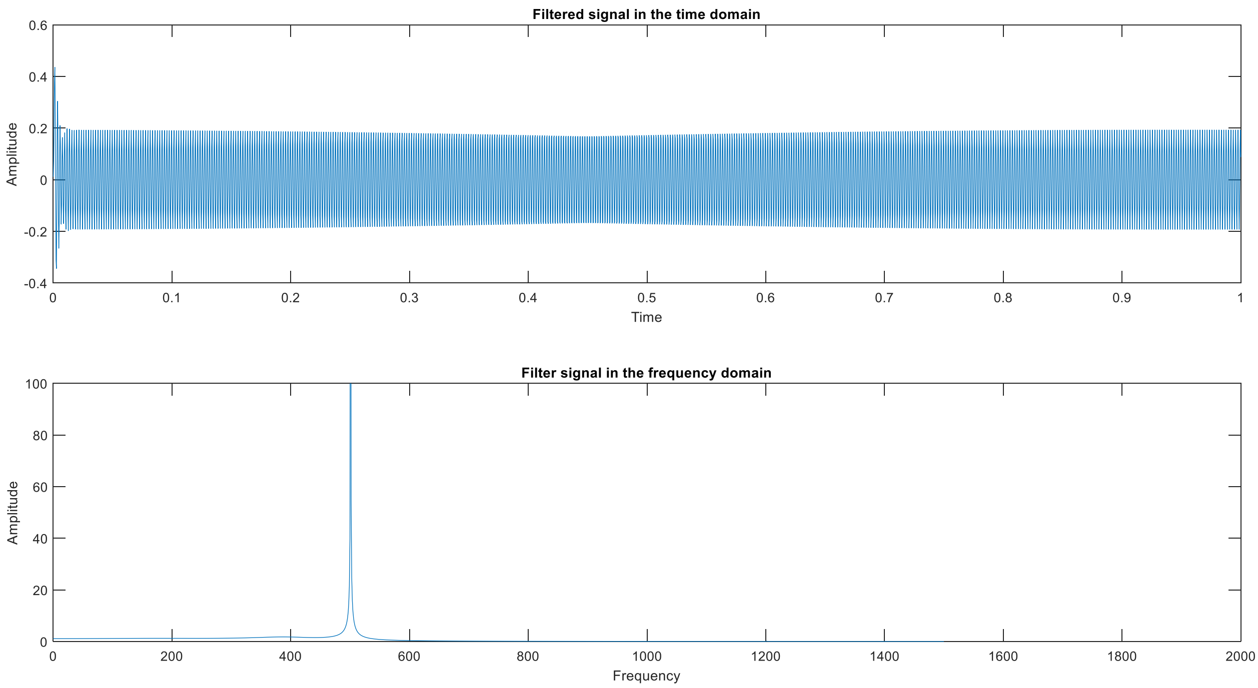


Figure 5 Filtered by Chebyshev-1

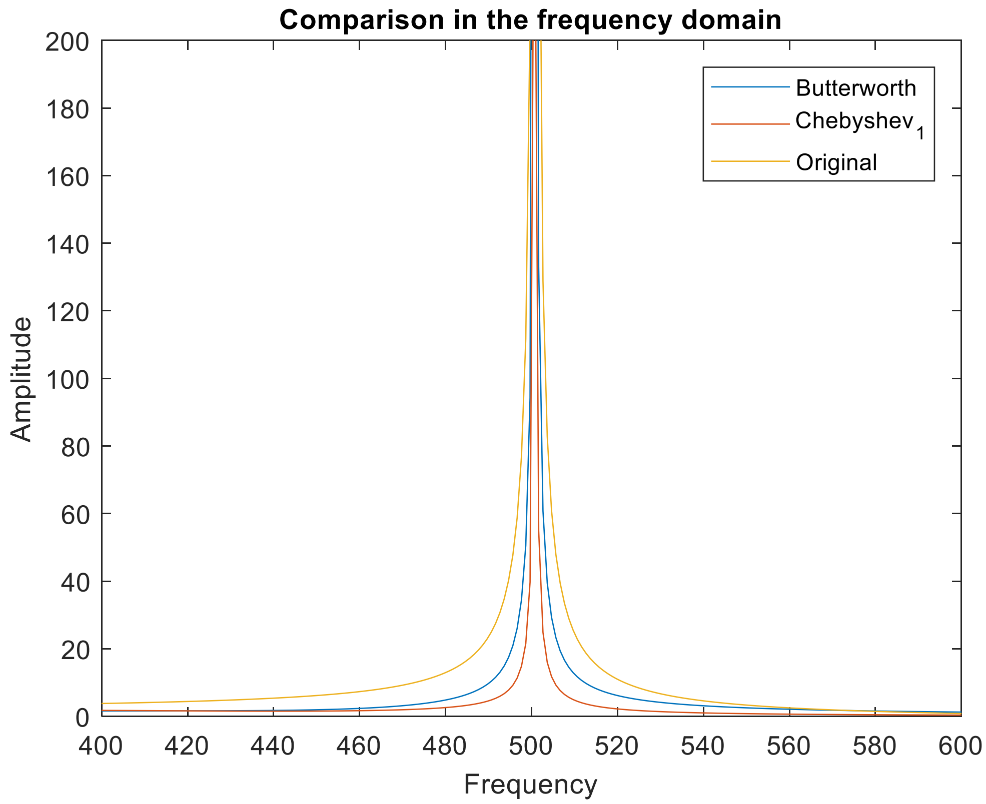


Figure 6 Differences Between the Two Filter

[Methods to design analog IIR filters]

(1) Firstly, transform the requirements of the designed analog filters to the prototype analog low pass filters.

(2) Secondly, design the prototype low pass filters (such as BW, CB, and C).

(3) Lastly, design the desired filters from analog low pass filters through the frequency conversion.

The digital filters of Butterworth and Chebyshev-1 have been shown in the former section. Therefore, this section will only provide the necessary steps for designing an elliptic filter (Cauer filter).

The minimum order of elliptic:

$$N \geq ceil\left(\frac{K\left(\frac{w_p}{w_s}\right)K\left(\sqrt{1-\frac{10^{0.1A_s}-1}{10^{0.1A_p}-1}}\right)}{\mathrm{K}\left(1-\frac{w_p^2}{w_s^2}\right)\mathrm{K}\left(\frac{\sqrt{10^{0.1A_s}-1}}{\sqrt{10^{0.1A_p}-1}}\right)}\right)=3 \qquad 1-(3)$$

The frequency response of this filter is represented by the following figure:

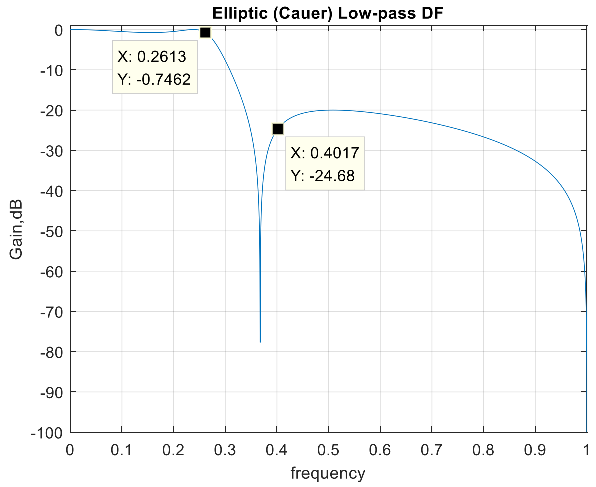


Figure 7 Frequency Response of Elliptic Digital Filter

The signal filtered by the Elliptic digital filter is shown below:

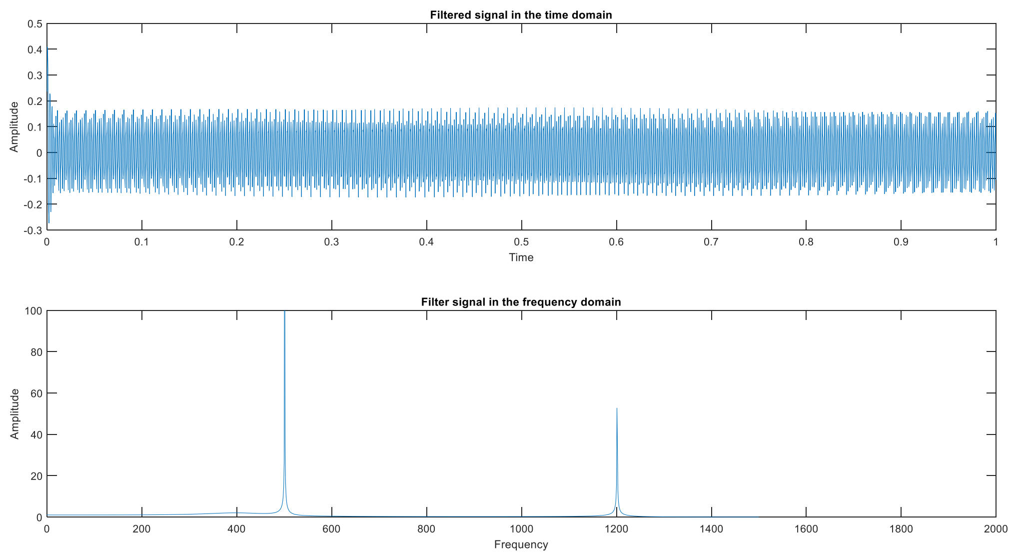


Figure 8 Filtered by Elliptic (Cauer)

The code for Cauer is exclusively shown here:

```
% The Digital requirements
Wp=0.2613*pi; Ws=0.4018*pi; Ap=0.75; As=20;

% Sampling frequency(Hz)
Fs=1000; T=1/Fs;

% Bilinear analogue requirements
wp=2*tan(Wp/2)/T;
ws=2*tan(Ws/2)/T;

% Determine the order of AF filter
% and the 3-dB cutoff frequency
[N,wc]=ellipord(wp,ws,Ap,As,'s');
```

```
% Determine the AF-CBI filter
[numa,dena]=ellip(N,Ap,As,wc,'s');

% Convert AF to DF by bilinear transformation
[numd,dend]=bilinear(numa,dena,Fs);

% Plot the frequency response
w=linspace(0,pi,5000);
h=freqz(numd,dend,w);
plot(w/pi,20*log10(abs(h)));
axis([0 1 -100 1]);grid;
xlabel('frequency'); ylabel('Gain,dB');
title("Elliptic (Cauer) Low-pass DF");
```

[Showing your results using MATLAB]

Butterworth filters:

The frequency response of the Butterworth filter – Figure 1

The input of the Butterworth filter – Figure 3

The output of the Butterworth filter – Figure 4

Chebyshev-I filters:

The frequency response of the Chebyshev-I filter – Figure 2

The input of the Chebyshev-I filter – Figure 3

The output of the Chebyshev-I filter – Figure 5

[Analysis of your results]

Butterworth filters:

Compared to the Chebyshev filter, Butterworth filter shows a flatter pass-band but a poor roll-off rate. According to figure 1, it exhibits a perfectly flat pass-band without any ripples. The roll-off is smooth and monotonic, with low-pass or high-pass roll-off rate of 20 dB/decade (6 dB/octave) for every pole. Thus, a 7th-order Butterworth low-pass filter would have an attenuation rate of 140 dB for every factor of ten increase in frequency beyond the cut-off frequency. It has a reasonably good phase response.

According to figure 4, it has filtered the higher frequency component of the signal and kept the one with lower frequency. From figure 6, it nearly influences nothing relevant to the amplitude of the components located inside the pass-band, due to the flat pass-band; however, the slow roll-off rate makes the remains fatter than the one filtered by the Chebyshev filter.

<u>Chebyshev-I filters:</u>

Compared to a Butterworth filter, a Chebyshev filter can achieve a sharper transition between the passband and the stopband with a lower order filter. According to figure 2, the sharp transition between the passband and the stopband of a Chebyshev filter produces smaller absolute errors and faster execution speeds than a Butterworth filter. Chebyshev filters where the ripple is only allowed in the passband are called type 1 filters. Chebyshev filters have a poor phase response.

According to figure 5, it has filtered the higher frequency component of the signal and kept the one with lower frequency. From figure 6, it nearly influences nothing relevant to the width of the components located inside the pass-band, due to the sharp transition band; however, the ripples in the pass-band make the average amplitude of the remains differ from the original one.

[Contents you must study by yourself]

1. Design analog low pass filters using BW, CB, and C filters.
2. Design analog low pass filters using BW, CB, and C filters. using MATLAB.
3. Transform analog low pass filters to the desired filters through frequency conversion.

1. Solution:

<u>Low-pass BW:</u>

Step 1: Determine the order N of an Analogue Filter,

$$N \geq ceil\left(\frac{\lg\left(\frac{10^{0.1A_s}-1}{10^{0.1A_p}-1}\right)}{2\lg\left(\frac{w_s}{w_p}\right)}\right)$$

Step 2: Determine 3dB cut-off frequency w_c,

$$\frac{w_p}{(10^{0.1A_p}-1)^{\frac{1}{2N}}} \leq w_c \leq \frac{w_s}{(10^{0.1A_s}-1)^{\frac{1}{2N}}}$$

Step 3: Determine the poles of the system function of AF,

$$s_k = w_c e^{j\pi(\frac{1}{2}+\frac{2k-1}{2N})}; k = 1,2,\dots,N$$

Step 4: Determine the system function $H_L(s)$ of Low-pass AF.

$$H_L(s) = \prod_{k=1}^{N} \frac{1}{s-s_k}$$

<u>Low-pass CB1:</u>

Step 1: Determine w_c through w_p,

$$w_c = w_p$$

Step 2: Determine ε through A_p,

$$\varepsilon = \sqrt{10^{0.1A_p}-1}$$

Step 3: Determine N through stopband specifications,

$$N \geq ceil\left(\frac{\operatorname{arccosh}\frac{\sqrt{10^{0.1A_s}-1}}{\sqrt{10^{0.1A_p}-1}}}{\operatorname{arccosh}\left(\frac{w_s}{w_p}\right)}\right)$$

Step 4: Determine the poles of ALF through $|H(jw)|^2$,

$$s_k = \sigma_k + jw_k, k = 1,2,\dots,N$$

$$where\ \sigma_k = -\sinh(\beta)\sin\left(\frac{(2k-1)\pi}{2N}\right), w_k = -\cosh(\beta)\cos\left(\frac{(2k-1)\pi}{2N}\right), \beta = \frac{\operatorname{arcsinh}\left(\frac{1}{\varepsilon}\right)}{N}$$

Step 5: Determine $H_L(s)$ through the poles.

$$H_L(s) = \prod_{k=1}^{N} \frac{1}{s - s_k}$$

<u>Low-pass C:</u>

Step 1: Determine w_c through w_p,

$$w_c = w_p$$

Step 2: Step2：Determine ε through A_s,

$$\varepsilon = \frac{1}{\sqrt{10^{0.1A_p} - 1}}$$

Step 3: Determine k through stopband cut-off frequency w_s,

$$k = \frac{w_p}{w_s}$$

Step 4: Determine k_1 through stopband attenuation A_s,

$$k_1 = \frac{\varepsilon}{\sqrt{10^{0.1A_s} - 1}}$$

Step 5: Determine the order N,

$$N \geq ceil\left(\frac{K\left(\frac{w_p}{w_s}\right) K\left(\sqrt{1 - \frac{10^{0.1A_s} - 1}{10^{0.1A_p} - 1}}\right)}{\mathrm{K}\left(1 - \frac{w_p^2}{w_s^2}\right) \mathrm{K}\left(\frac{\sqrt{10^{0.1A_s} - 1}}{\sqrt{10^{0.1A_p} - 1}}\right)}\right)$$

Step 6: Determine k and k_1 to make the equation (5) in balance,

Step 7: Determine $H_L(s)$ through (1) ~ (5).

2. Solution: it has been solved, and the corresponding codes will be provided in the next following section.

3. Solution:

Spectral transformations in the analog domain

Transformation type	Frequency transformation	Complex frequency transformation	Notes
Prototype LP→LP	$\bar{\omega} = \omega / \omega_0$	$\bar{s} = s / \omega_0$	ω_0 is positive.
Prototype LP→HP	$\bar{\omega} = \omega_0 / \omega$	$\bar{s} = \omega_0 / s$	ω_0 is positive.
Prototype LP→BP	$\bar{\omega} = \frac{\omega^2 - \omega_0^2}{B\omega}$	$\bar{s} = \frac{s^2 + \omega_0^2}{Bs}$	$B = \omega_{p2} - \omega_{p1}$ $\omega_0^2 = \omega_{p1}\omega_{p2}$
Prototype LP→BS	$\bar{\omega} = \frac{B\omega}{\omega^2 - \omega_0^2}$	$\bar{s} = \frac{Bs}{s^2 + \omega_0^2}$	$B = \omega_{s2} - \omega_{s1}$ $\omega_0^2 = \omega_{s1}\omega_{s2}$

[References]

1. Sanjit Mitra, Digital Signal Processing, Mc Graw Hill press.2001.

2. Richard G.Lyons, Understanding Digital Signal Processing, Third Edition.2015.

[Finding problems]

Compare the impulse response and the bilinear transform using BW.

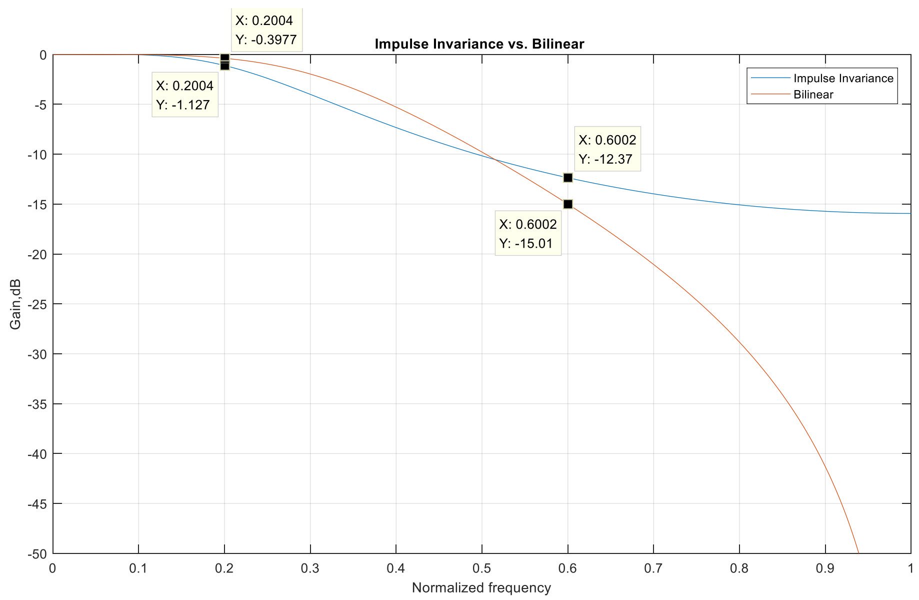


Figure 9 Impulse Invariance vs. Bilinear Using BW

Impulse invariance method is a technique for designing discrete-time infinite-impulse-response (IIR) filters from continuous-time filters in which the impulse response of the continuous-time system is sampled to produce the impulse response of the discrete-time system. The frequency response of the discrete-time system will be a sum of shifted copies of the frequency response of the continuous-time system.

The bilinear transformation method is an alternative to impulse invariance that uses a different mapping that maps the continuous-time system's frequency response, out to infinite frequency, into the range of frequencies up to the Nyquist frequency in the discrete-time case, as opposed to mapping frequencies linearly with circular overlap as impulse invariance does.

Figure 9 shows the comparison between methods of impulse invariance and bilinear to transfer the low-pass Butterworth analog filters to the digital ones. The blue curve of impulse invariance presents a problem that cannot meet the requirement of the stop-band attenuation, whereas the red curve of bilinear does not have this problem.

The impulse invariance method suffers from aliasing and is not used often. The bilinear transformation does not suffer from aliasing and is by far more popular than the impulse invariance method. The frequency relationship from the s-plane to the z-plane is non-linear, and one needs to compensate by pre-processing the critical frequencies such that after the transformation the desired response is realized. Stability is maintained in this transformation since the left-half s-plane maps onto the interior of the unit circle.

[MATLAB Codes]

Code 1 Butterworth Digital Filter Design Using Bilinear Transformation

```
% The Digital requirements
Wp=0.2613*pi; Ws=0.4018*pi; Ap=0.75; As=20;

% Sampling frequency(Hz)
Fs=1000; T=1/Fs;

% Bilinear analogue requirements
wp=2*tan(Wp/2)/T;
ws=2*tan(Ws/2)/T;

% Determine the order of AF filter
% and the 3-dB cutoff frequency
[N,wc]=buttord(wp,ws,Ap,As,'s');

% Determine the AF-BW filter
[numa,dena]=butter(N,wc,'s');

% Convert AF to DF by bilinear transformation
[numd,dend]=bilinear(numa,dena,Fs);

% Plot the frequency response
w=linspace(0,pi,5000);
h=freqz(numd,dend,w);
plot(w/pi,20*log10(abs(h)));
axis([0 1 -100 1]);grid;
xlabel('frequency'); ylabel('Gain,dB');
title("Butterworth Low-pass DF");
```

Code 2 Chebyshev-1 Digital Filter Design Using Bilinear Transformation

```
% The Digital requirements
Wp=0.2613*pi; Ws=0.4018*pi; Ap=0.75; As=20;

% Sampling frequency(Hz)
Fs=1000; T=1/Fs;

% Bilinear analogue requirements
wp=2*tan(Wp/2)/T;
ws=2*tan(Ws/2)/T;

% Determine the order of AF filter
% and the 3-dB cutoff frequency
[N,wc]=cheb1ord(wp,ws,Ap,As,'s');

% Determine the AF-CBI filter
[numa,dena]=cheby1(N,Ap,wc,'s');

% Convert AF to DF by bilinear transformation
[numd,dend]=bilinear(numa,dena,Fs);

% Plot the frequency response
w=linspace(0,pi,5000);
h=freqz(numd,dend,w);
plot(w/pi,20*log10(abs(h)));
axis([0 1 -100 1]);grid;
xlabel('frequency'); ylabel('Gain,dB');
title("Chebyshev I Low-pass DF");
```

Code 3 Signal Passes Through Various Filters

```
f1=500;% frequency inside the roll-off
f2=1200;% frequency outside the roll-off
fs=3000;% sampling frequency
T=1;% sampling interval
n = T*fs;% number of sampling points
t = linspace(0,T,n);
```

```
xlabel('Time');ylabel('Amplitude');
subplot(4,1,3);
plot(t,y);
title('Mixture in the time domain');
xlabel('Time');ylabel('Amplitude');
subplot(4,1,4);
```

```
% Input signal
y1 = cos(2*pi*f1*t);
y2 = cos(2*pi*f2*t);
y = y1 + y2;

% Input signal in frequency domain
fft_y=fftshift(fft(y));
f=linspace(-fs/2,fs/2,n);

% Filter the input signal
y_filter=filter(numd,dend,y);
y_filter_freq=fftshift(fft(y_filter));

% Input signal figures
figure(1);
subplot(4,1,1);
plot(t,y1);
title('y_1 in the time domain');
xlabel('Time');ylabel('Amplitude');
subplot(4,1,2);
plot(t,y2);
title('y_2 in the time domain');
plot(f,abs(fft_y));
title('Mixture in the frequency domain');
xlabel('Frequency');ylabel('Amplitude');
axis([0 2000 0 100]);

% Output signal figures
figure(2);
subplot(2,1,1);
plot(t,y_filter);
title('Filtered signal in the time domain');
xlabel('Time');ylabel('Amplitude');
subplot(2,1,2);
plot(f,abs(y_filter_freq));
xlabel('Frequency');ylabel('Amplitude');
title('Filter signal in the frequency domain');
axis([0 2000 0 100]);

figure(3);
plot(f,abs(y_filter_freq));
hold on;
xlabel('Frequency');ylabel('Amplitude');
title('Comparison in the frequency domain');
axis([400 600 0 200]);
```

Code 4 Low-Pass Butterworth to Band-Stop

```
% The Analogue requirements
wp1=10; wp2=30; ws1=19; ws2=21; Ap=1; As=20;

% parameter transformation
ws = 1;
[wp,B,w0] = LP2BStrans(ws1,ws2,wp1,wp2);

% BW Low-pass filter
[N,Wc]=buttord(wp,ws,Ap,As,'s');
[num,den] = butter(N,Wc,'s');
% LP to BS
[numt,dent] = lp2bs(num,den,w0,B);

% plot
w=linspace(5,35,1000);
h=freqs(numt,dent,w);
plot(w,20*log10(abs(h)));
w=[wp1 ws1 ws2 wp2];
set(gca,'xtick',w);grid;
h=freqs(numt,dent,w);A=-20*log10(abs(h))
```

```
function [wp,B,w0] =
LP2BStrans(ws1,ws2,wp1,wp2)
%LP2BSTRANS is to transfer LP's parameters
to BP
% w_bar=(w^2-w0^2)/(B*w), where B=wp2-wp1
and w0=wp1wp2
B = ws2-ws1;
w0 = sqrt(ws1*ws2);
wp1_bar = (B*wp1)/(wp1^2-w0^2);
wp2_bar = (B*wp2)/(wp2^2-w0^2);
wp = max(abs(wp1_bar),abs(wp2_bar));
end
```

Code 5 Low-Pass Butterworth to Band-Pass

```
% The Analogue requirements
wp1=6; wp2=8; ws1=4; ws2=11; Ap=1; As=32;

% parameter transformation
wp = 1;
[ws,B,w0] = LP2BPtrans(ws1,ws2,wp1,wp2);

% BW Low-pass filter
[N,Wc]=buttord(wp,ws,Ap,As,'s');
[num,den] = butter(N,Wc,'s');
```

```
% LP to BP
[numt,dent] = lp2bp(num,den,w0,B);

% plot
w=linspace(2,12,1000);
h=freqs(numt,dent,w);
plot(w,20*log10(abs(h))) ; grid ;
xlabel('Frequency in rad/s');
ylabel('Gain in dB')
```

```
function [ws,B,w0] =
LP2BPtrans(ws1,ws2,wp1,wp2)
%LP2BPTRANS is to transfer LP's parameters
to BP
% w_bar=(w^2-w0^2)/(B*w), where B=wp2-wp1
and w0=wp1wp2
```

```
B = wp2-wp1;
w0 = sqrt(wp1*wp2);
ws1_bar = (ws1^2-w0^2)/(B*ws1);
ws2_bar = (ws2^2-w0^2)/(B*ws2);
ws = min(abs(ws1_bar),abs(ws2_bar));
end
```

Code 6 Comparison Between Impulse Response and The Bilinear Transform

```
% DF BW LP specfication
Wp=0.2*pi; Ws=0.6*pi; Ap=2; As=15;
T=1;Fs=1/T; % Sampling frequency(Hz)

% Use impulse invariance
wp_ii=Wp*Fs; ws_ii=Ws*Fs;
% Use bilinear invariance
wp_bi=2*tan(Wp/2)/T;ws_bi=2*tan(Ws/2)/T;

%determine the order of AF filter
[N_ii,wc_ii]=buttord(wp_ii,ws_ii,Ap,As,'s');
[N_bi,wc_bi]=buttord(wp_bi,ws_bi,Ap,As,'s');

%determine the AF-BW filter
[numa_ii,dena_ii]=butter(N_ii,wc_ii,'s');
[numa_bi,dena_bi]=butter(N_bi,wc_bi,'s');
```

```
%determine the DF filter
[numd_ii,dend_ii]=impinvar(numa_ii,dena_ii,Fs);
[numd_bi,dend_bi]=bilinear(numa_bi,dena_bi,Fs);

%plot the frequency response
w=linspace(0,pi,1024);
h_ii=freqz(numd_ii,dend_ii,w);
h_bi=freqz(numd_bi,dend_bi,w);
plot(w/pi,20*log10(abs(h_ii/max(abs(h_ii)))));
hold on; grid on;
plot(w/pi,20*log10(abs(h_bi)));
axis([0 1 -50 0]);
xlabel('Normalized
frequency');ylabel('Gain,dB');
legend("Impulse Invariance","Bilinear");
title('Impulse Invariance vs. Bilinear ');
```

[Objectives Topics to discuss]

(1) Master the principles to design FIR filters.

(2) Master the method to design FIR filters with the requirements according to the principles.

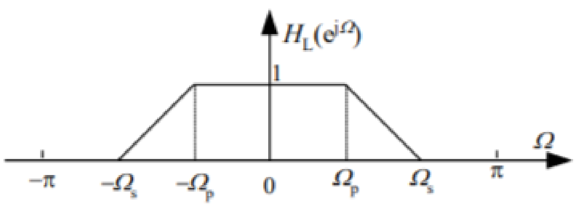
(3) Improve students' abilities of self-study, finding, analyzing, and solving problems.

[Topics to discuss]

(1) Design a linear phase FIR low pass filter using the Blackman and Kaiser window respectively.

Ω_p=0.4π, A_p=0.5dB, Ω_s=0.6π, A_s=55dB.

(2) During design in order to improve the Gibbs phenomenon at the cutoff frequency point, a gradually decaying window function is employed. As shown in the figure below, the frequency response of an ideal low-pass filter with a linear transition band is used to design an FIR filter approaching the frequency response using the window method.



[Methods to design FIR DF]

There are mainly two methods to design the FIR DF, which are windowed FS and frequency sample. Using the Blackman and Kaiser windows to truncate belongs to the first method.

<u>For windowed Fs:</u>

Step 1: Determine the category of DF (LP, HP, BP or BS, etc.),

Step 2: Determine the type of linear phase FIR (Type-I, II, III or IV),

Step 3: Determine the amplitude function $A_d(\Omega)$, Step 4: Determine the amplitude function $\Phi_d(\Omega)$,

$$\Phi_d(\Omega) = -0.5M\Omega + \beta,$$

$$\text{where } \beta = 0 \text{ if type I or II, and } \beta = \frac{\pi}{2} \text{ if type III or IV}$$

Step 5: Compute $h_d[k]$ through IDTFT,

$$h_d[k] = \frac{1}{2\pi}\int_{-\pi}^{\pi} A_d(\Omega)e^{j\Phi_d(\Omega)}e^{jk\Omega}\,d\Omega$$

Step 6: Truncate $h_d[k]$ to $h[k]$.

$$h[k] = \{\begin{matrix} h_d[k] .* W_N[k], & 0 \le k \le M \\ 0, & otherwise \end{matrix}$$

Where,

$$W_N[k] = 0.42 - 0.5\cos\left(\frac{2\pi k}{M}\right) + 0.08\cos\left(\frac{4\pi k}{M}\right), 0 \le k \le M \; for\; Blackman$$

$$W_N[k] = \frac{I_0\left(\beta\sqrt{1-\left(1-\frac{2k}{M}\right)^2}\right)}{I_0(\beta)}, 0 \le k \le M \; for\; Kaiser$$

<u>For frequency sample:</u>

Step 1: Determine the type of DF,

Step 2: Obtain an M+1 sample $H_d[m]$ in $H_d[e^{j\Omega}]$ in the range $\Omega\epsilon[0,2\pi)$. And make $H_d[m]$ satisfy linear phase condition,

$$H_d[m] = H_d\left[e^{j\Omega}\right]\Big|_{\Omega=\frac{2\pi}{M+1}m} = e^{j\beta}e^{-j\frac{M\pi}{M+1}m}A_d\left(\frac{2\pi m}{M+1}\right) \rightarrow A_d\left(\frac{2\pi m}{M+1}\right) = A_d[m]$$

Step 3: Obtain $h[k] = IDFT\{H_d[m]\}$.

$$h[k] = \frac{1}{M+1}\sum_{m=0}^{M} H_d[m]W_{M+1}^{-mk}$$

We will use the first method to solve the question (1) above.

[Showing your results using MATLAB]

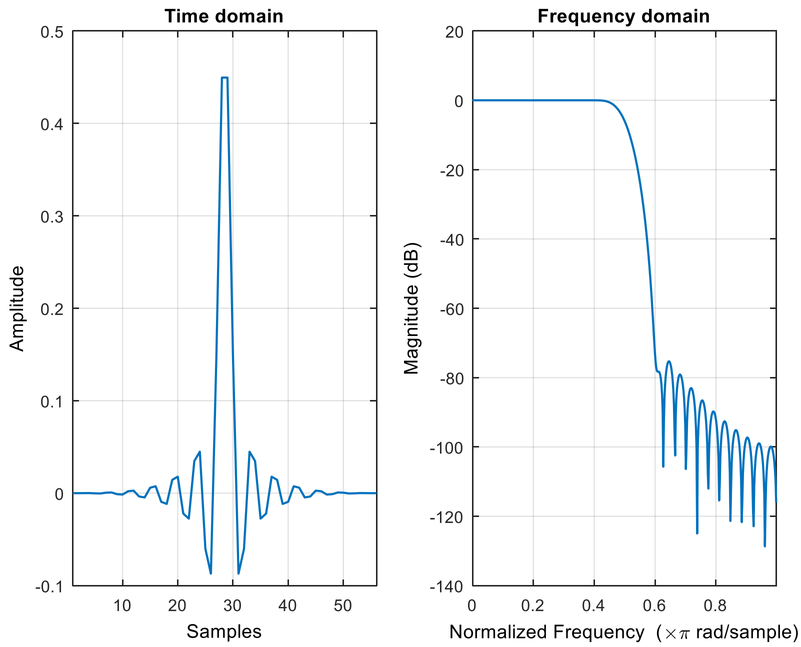


Figure 10 wvtool of Blackman

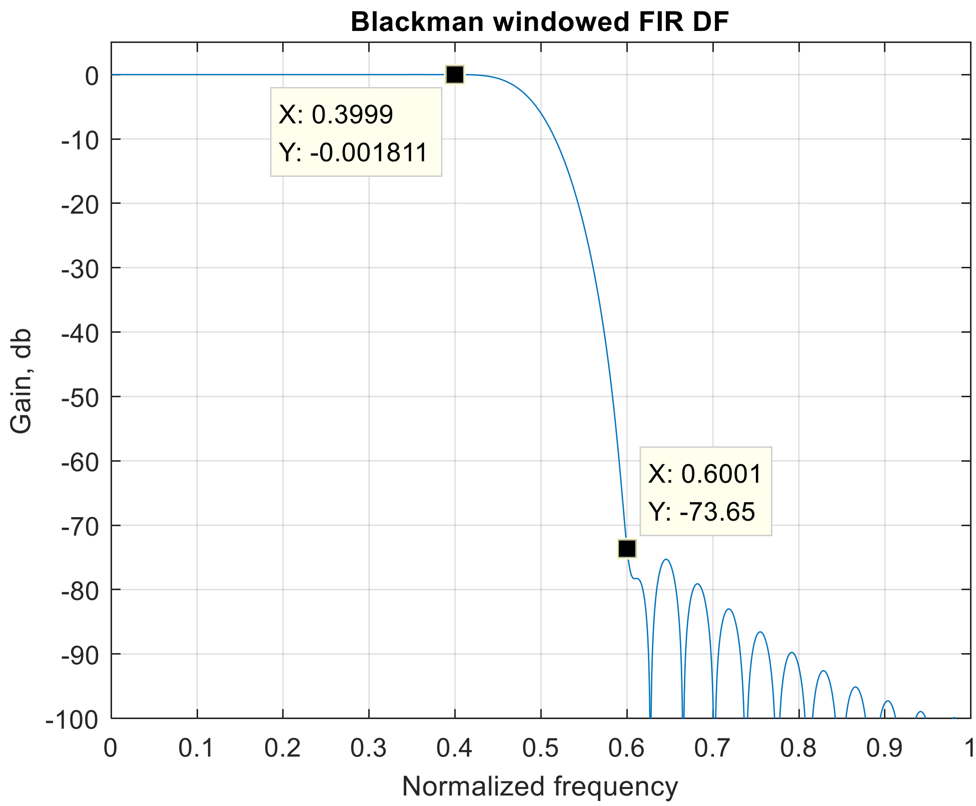


Figure 11 Blackman windowed FIR DF

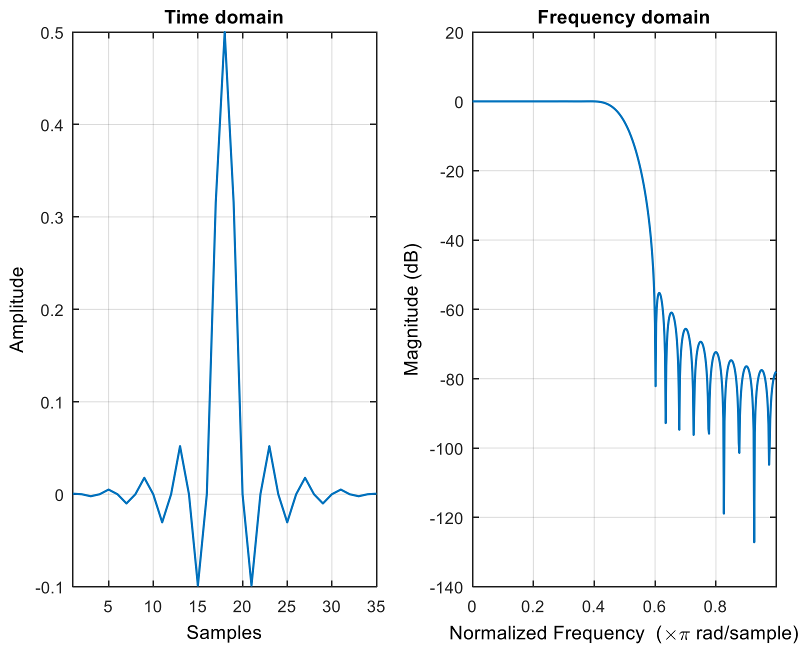


Figure 12 wvtool of Kaiser

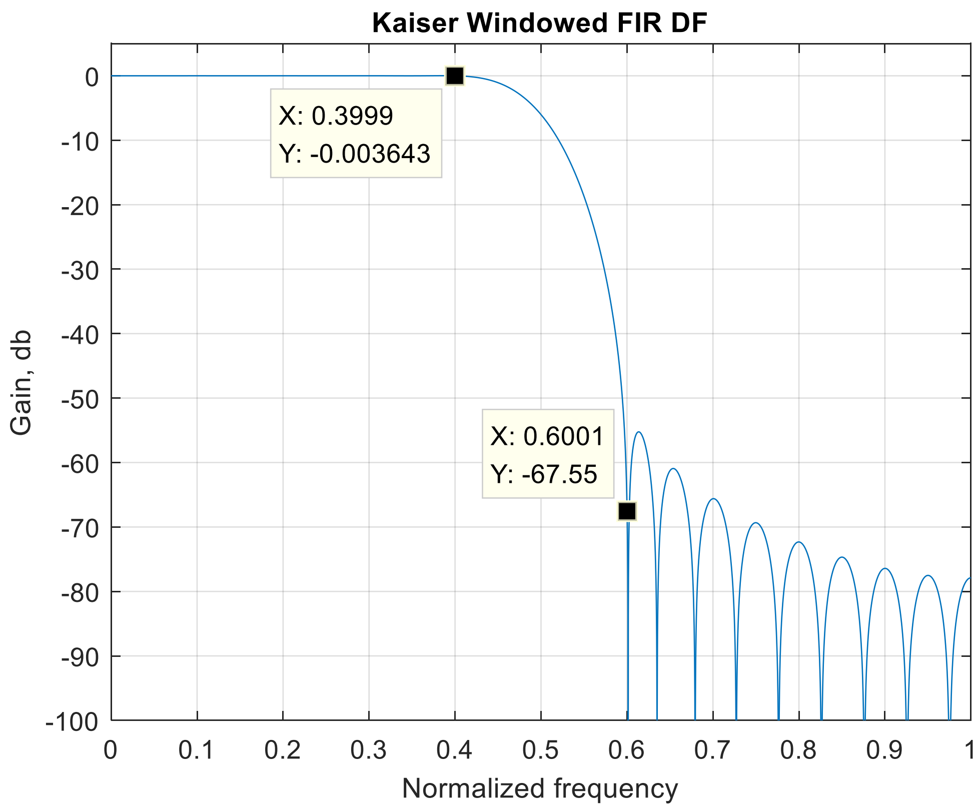


Figure 13 Blackman windowed FIR DF

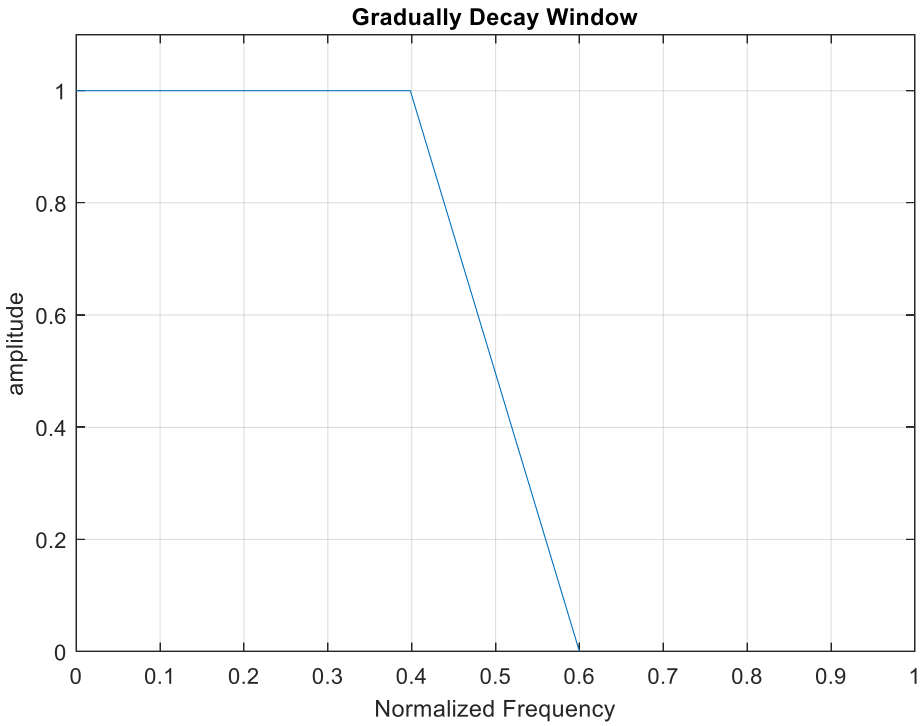


Figure 14 Gradually Decay Window

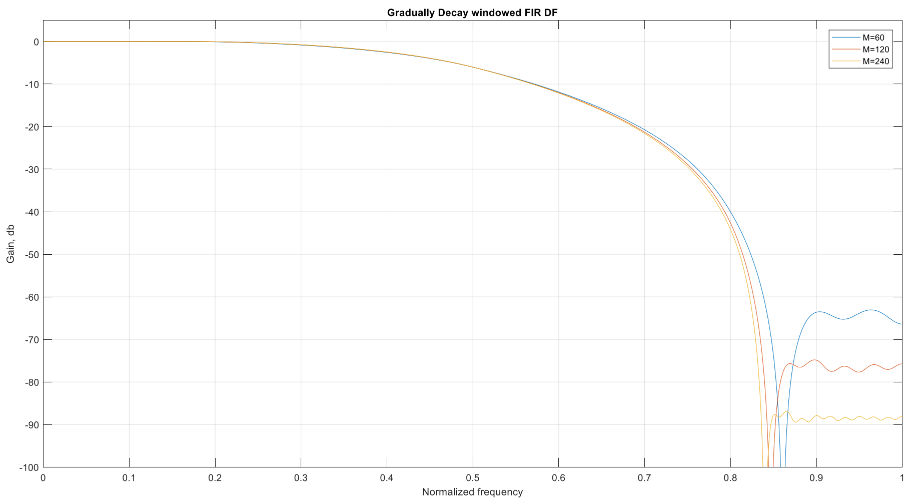


Figure 15 Gradually Decay Windowed FIR DF

[Analysis of your results]

Compared to the Blackman windowed method with Hamming, it has an extra cosine term, reducing the side lobes. Besides, for FIR low-pass filter design, there exist fewer side lobes using a Blackman window to truncate the signal. However, the equivalent noise bandwidth is greater in the Blackman window than the Hamming window, while the ENBW is smaller for the Blackman window.

In the Kaiser window, the passband and stopband ripple size are not affected much by the change in the window length (when compared to other windows). It is affected by how the coefficients roll-off (the shape factor). The window length mainly affects the transition band. So, keeping the window length constant, we can adjust the shape factor, to design for the passband and stopband ripples. This is a huge advantage over other windows, where the window length, ripple size, and transition bandwidth have a three-way tradeoff. The only disadvantage with the Kaiser window is that we design for the same ripple size in both passband and stopband. So, there would be some overdesign in either of the bands

According to Figures 11 and 13, both windows truncating methods accomplished the requirements quite well, but the Kaiser window uses a much smaller window length, which is the advantage of the Kaiser window.

Gradually Decay Windowed method to design the FIR digital filters can curtail the Gibbs phenomenon, to some extent, but it sacrifices the pass-band and stop-band attenuation. According to figure 15, with the incremental of window length, the Gibbs phenomenon has been improved.

[References]

1. Sanjit Mitra, Digital Signal Processing, Mc Graw Hill press.2001.
2. Richard G.Lyons, Understanding Digital Signal Processing, Third Edition.2015

[Finding problems]

How to compromise between the stopband attenuation and the transition bandwidth.

Table 1 Properties of commonly used windows (L=M+1)

Window Name	Side Lobe level (dB)	Approx.△w	Exact △w	δp=δs	Ap (dB)	As (dB)
Rectangular	-13	4π/L	1.8π/L	0.09	0.75	21
Bartlett	-25	8π/L	6.1π/L	0.05	0.25	26
Hann	-31	8π/L	6.2π/L	0.0063	0.055	44
Hamming	-41	8π/L	6.6π/L	0.0022	0.019	53
Blackman	-51	12π/L	11π/L	0.0002	0.0017	74

The passband and stopband ripples and the width of the transition band are determined by the running-integral of the amplitude window function. Furthermore, to a very good approximation, the passband ripple δp and stopband attenuation δs are equal and independent of M; they can be changed only by changing the shape of the window. The width of the transition band, which is controlled by the width of the main lobe of the amplitude window function, can be reduced by increasing the order of the filter. Table 1 summarizes the properties of the windows. These windows thus are termed “fixed” windows since their stopband attenuation is independent of window length. A careful inspection of the table shows that the Hamming window provides the best choice for filter design since it gives the best compromise between a narrow transition width and a high stopband attenuation. However, in reality, the choices are always decided by the specific requirements.

[MATLAB Codes]

Code 7 Blackman windowed FIR DF

```
% filter specification
wp=0.4*pi; ws=0.6*pi; Ap=0.5; As=55;

% estimate M of Blackman window
df=abs(ws-wp);
M = ceil(11*pi/df);

% build the Blackman windowed DF
w = blackman(M+1); %blackman window
alpha=M/2; k=0:M; wc=(wp+ws)/2;
hd=(wc/pi)*sinc((wc/pi)*(k-alpha));
h=hd.*w';
```

```
% plot the figures
figure(1);
wvtool(h);
figure(2);
omega=linspace(0,pi,5000);
mag=freqz(h,[1],omega);
magdb=20*log10(abs(mag));
plot(omega/pi,magdb);axis([0 1 -100 5]);grid on;
title('Blackman windowed FIR DF');
xlabel('Normalized frequency');ylabel('Gain, db')
```

Code 8 Kaiser windowed FIR DF

```
% filter specification
wp=0.4*pi; ws=0.6*pi; Ap=0.5; As=55;

% estimate M and beta of Kaiser window
delta_p = 1-10^(-0.05*Ap);
delta_s = 10^(-0.05*As);
A = -20*log10(min(delta_p,delta_s));
M = ceil((A-7.95)/(2.285*abs(wp-ws)));
M = mod(M,2)+M;

if As > 50
    beta = 0.1102*(As-8.7);
elseif As>=21 && As<=50
    beta=0.5842*(As-21)^0.4+0.07886*(As-21);
else
    beta = 0;
end
```

```
% build the Kaiser windowed DF
w=kaiser(M+1,beta); %Kaiser window
alpha=M/2; k=0:M; wc=(wp+ws)/2;
hd=(wc/pi)*sinc((wc/pi)*(k-alpha));
h=hd.*w';

% plot the figures
figure(1);
wvtool(h);
figure(2);
omega=linspace(0,pi,5000);
mag=freqz(h,[1],omega);
magdb=20*log10(abs(mag));
plot(omega/pi,magdb);axis([0 1 -100 5]);grid on;
title('Kaiser Windowed FIR DF');
xlabel('Normalized frequency');ylabel('Gain, db')
```

Code 9 Gradually Decay windowed FIR DF

```
% filter specification
wp=0.4*pi; ws=0.6*pi; Ap=0.5; As=55;

% gradually decaying window design
```

```
% plot the figures
figure(1);
omegaW = linspace(-ws,ws,M);
plot(omegaW/pi,W);axis([0 1 0 1.1]);grid on;
```

```
Lt = 40; Lp = 4*Lt; M=Lt*2+Lp;
W = ones(1,M);
W(1:Lt)=(0:1:Lt-1)/Lt;
W(Lt+Lp+1:Lt*2+Lp)=fliplr(W(1:Lt));

% build the Blackman windowed DF
w = abs(ifftshift(ifft(W)));
w = (w-min(w))/(max(w)-min(w));
alpha=M/2; k=0:M-1; wc=(wp+ws)/2;
hd = (wc/pi)*sinc((wc/pi)*(k-alpha));
h = hd.*w;
```

```
xlabel('Normalized
Frequency');ylabel("amplitude");
title('Gradually Decay Window');
figure(2)
omega = linspace(0,pi,5000);
mag = freqz(h,[1],omega);
mag = abs(mag);
magnorm = (mag-min(mag))/(max(mag)-min(mag));
magdb = 20*log10(magnorm);
plot(omega/pi,magdb);axis([0 1 -100 5]);grid
on;
title('Gradually Decay windowed FIR DF');
xlabel('Normalized frequency');ylabel('Gain,
db');
hold on;
```

END