DSP: Filter Design Assignment

Kalpesh Patil (130040019) Filter Number: 13

Filter Specification

Filter 1

- Nature: Bandpass Filter
- Cutoff Frequencies (in kHz)

  Ω_{s1} = 8700 Ω_{p1} = 10700

  Ω_{p2} = 20700 Ω_{s2} = 22700

Filter 2

- Nature: Bandstop Filter
- Cutoff Frequencies (in kHz)

  Ω_{p1} = 8900 Ω_{s1} = 10900

  Ω_{s2} = 20900 Ω_{p2} = 22900

Filter Design

Filter 1

IIR Implementation

Monotonic in Passband as well as Stopband

Digital Frequencies

$$f_{digital} = \frac{f_{analog}}{2\pi f_{sample}}$$

ω_{s1} = 0.5466 ω_{p1} = 0.6723

ω_{p2} = 1.3006 ω_{s2} = 1.4263

Equivalent Analog Frequencies

$$\Omega = \tan\left(\frac{\omega}{2}\right)$$

Ω_{s1} = 0.2803 Ω_{p1} = 0.3494

Ω_{p2} = 0.7607 Ω_{s2} = 0.8650

Low Pass Equivalent Filter

$$\Omega_l = \frac{\Omega^2 - \Omega_0^2}{B\Omega}$$

Where

$\Omega_0^2 = \Omega_{p1}\Omega_{p2}$ and $B = \Omega_{p2} - \Omega_{p1}$

Stringent of the two values obtained from bandpass specifications is chosen as stop band frequency for low pass filter. Thus specifications of low pass filter needed to be deigned are as follows

$$\Omega_{lp} = 1$$
$$\Omega_{ls} = 1.3561$$

Monotonic response is needed in both stopband as well as passband, hence Butterworth Filter is designed

$$N \geq \frac{\ln \sqrt{\frac{D_2}{D_1}}}{\ln \frac{\Omega_{ls}}{\Omega_{lp}}}$$

Considering the minimum order required, we obtain $N = 8$

Poles of this Butterworth Filter are as follows

$$p_k = \Omega_c e^{i\{\left(\frac{2k+1}{2N}\right)+\frac{\pi}{2}\}}$$

$$gain_{DC} = \Omega_c^N$$

Transfer function is given by

$$H(s) = \frac{1.673}{s^8 + 5.467\, s^7 + 14.94\, s^6 + 26.5\, s^5 + 33.23\, s^4 + 30.14\, s^3 + 19.33\, s^2 + 8.043\, s + 1.673}$$

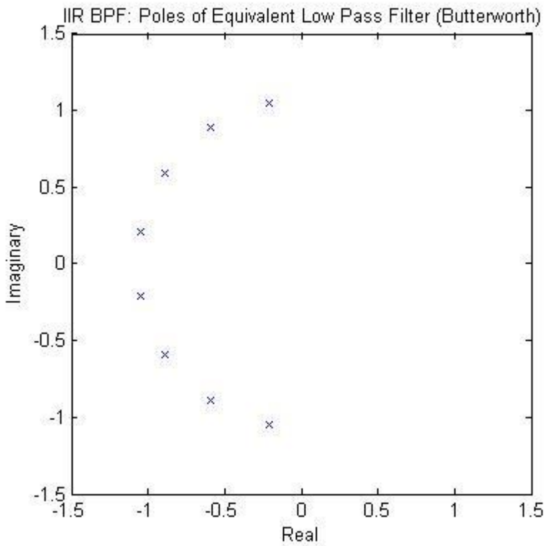

Fig1: Poles of Low Pass Equivalent (Butterworth) of IIR BPF

Analog Bandpass Filter

To obtain analog bandpass filter, following transformation was carried out in the transfer function

$$s_l = \frac{s^2 + \Omega_0^2}{Bs}$$

After transformation, following transfer function was obtained

$$H(s) = \frac{0.00137\, s^8}{\begin{array}{c} s^{16} + 2.248s^{15} + 4.654s^{14} + 6.027s^{13} + 6.96s^{12} + 6.14s^{11} + 4.834s^{10} \\ +3.079\, s^9 + 1.753\, s^8 + 0.8183s^7 + 0.3415s^6 + 0.1153s^5 + 0.03473s^4 \\ +0.007995s^3 + 0.001641s^2 + 0.0002107s + 2.491 * 10^{-5} \end{array}}$$

Digital Filter

To obtain analog bandpass filter, following transformation was carried out in the transfer function

$$s = \frac{1 - z^{-1}}{1 + z^{-1}}$$

After transformation, following transfer function was obtained

$$H(z) = \frac{3.604 * 10^{-5}\, z^{16} - 0.0002883\, z^{14} + 0.001009\, z^{12} - 0.002018\, z^{10} + 0.002523\, z^{8} - 0.002018\, z^{6} + 0.001009\, z^{4} - 0.0002883\, z^{2} + 3.604 * 10^{-5}}{z^{16} - 7.301\, z^{15} + 28.06\, z^{14} - 73.26\, z^{13} + 143.8\, z^{12} - 223.5\, z^{11} + 283\, z^{10} - 297.2\, z^{9} + 260.9\, z^{8} - 192\, z^{7} + 118.2\, z^{6} - 60.22\, z^{5} + 24.98\, z^{4} - 8.195\, z^{3} + 2.02\, z^{2} - 0.3384\, z + 0.03006}$$

Images for the pole-zero plot, magnitude response and phase response obtained using fvtool are shown below

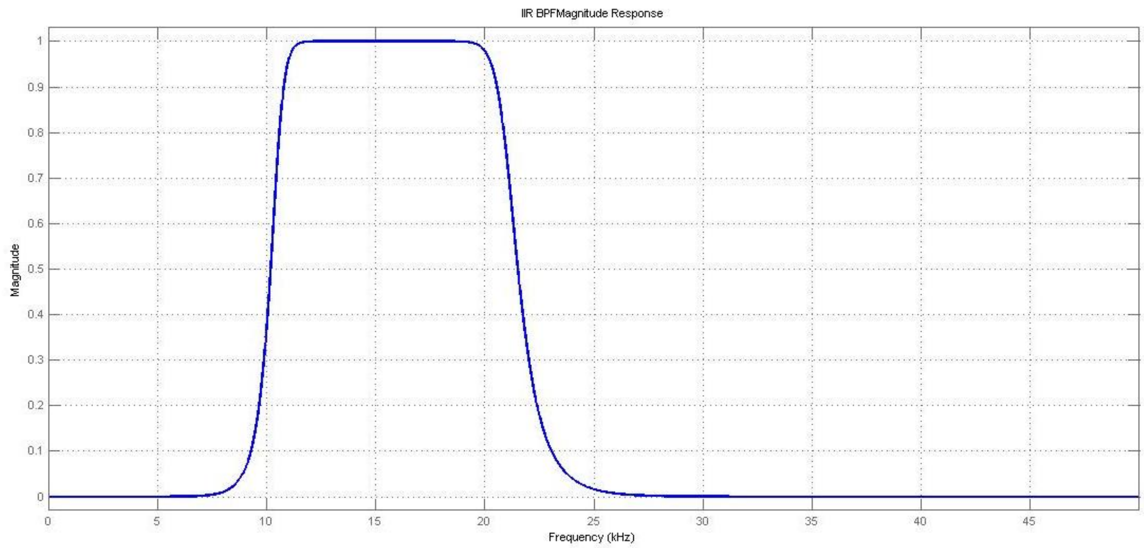

Fig2: Magnitude Response for IIR BPF

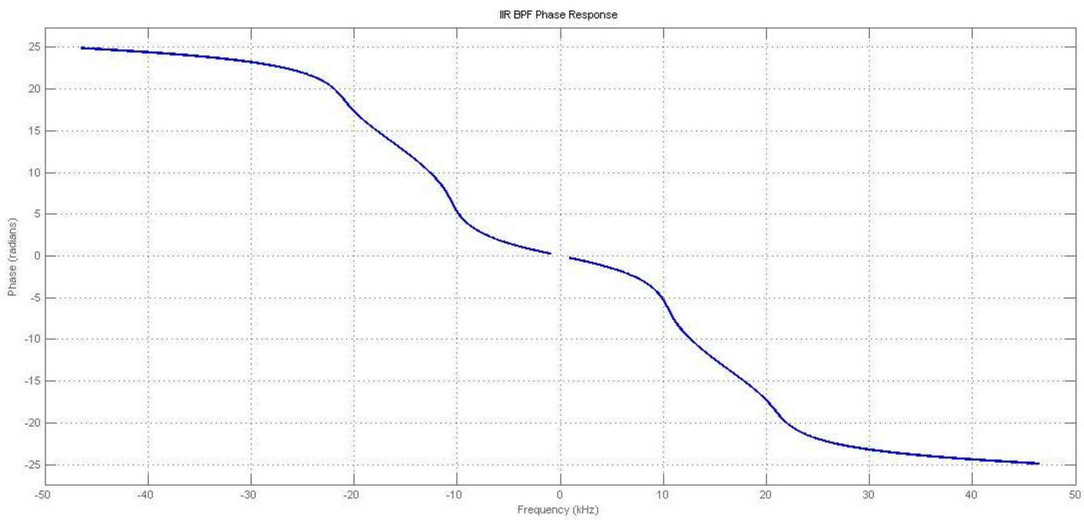

fig3: Phase Response for IIR BPF

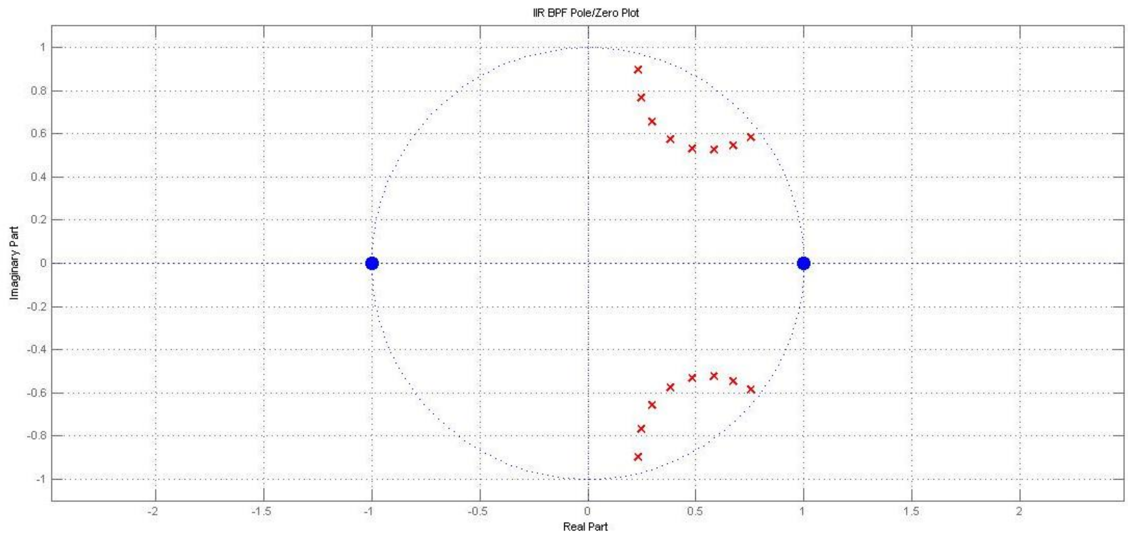

fig4: Pole-Zero Plot for IIR BPF

Direct Form 2

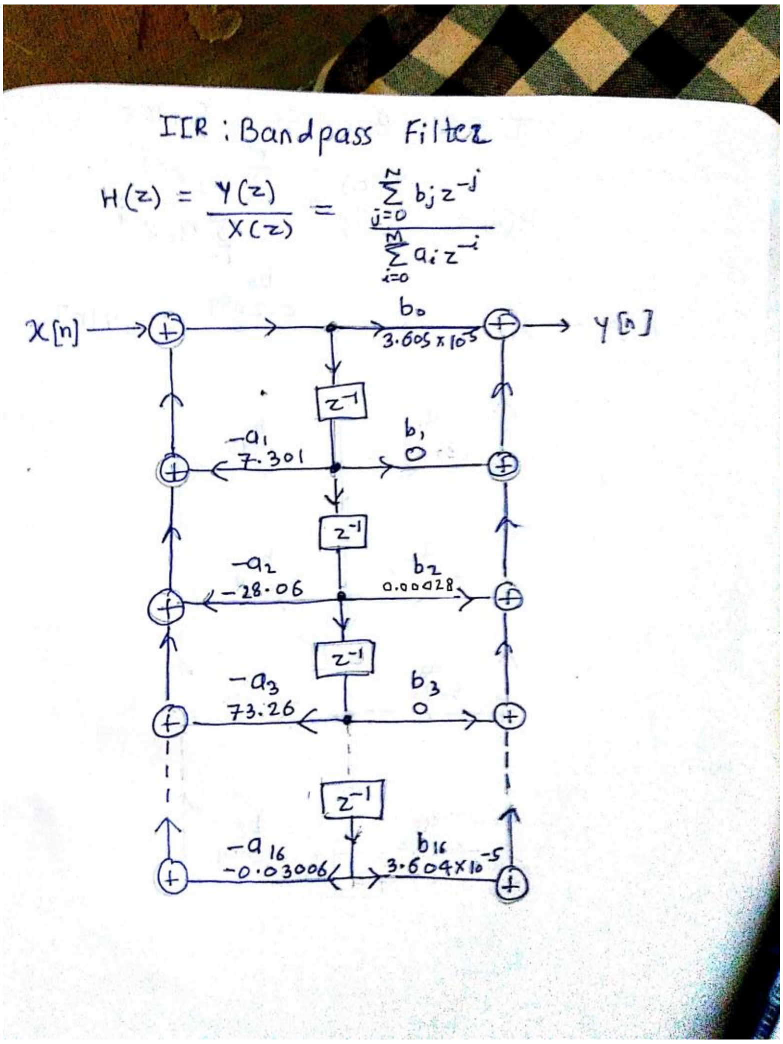

Fig5: Direct form 2 representation of IIR BPF

FIR Implementation

Ideal Impulse Response

$$h_{ideal}[n] = \frac{\sin(k\omega_{c2}) - \sin(k\omega_{c1})}{\pi n} \qquad for\ n \neq 0$$

$$= \frac{\omega_{c2} - \omega_{c1}}{\pi} \qquad for\ n = 0$$

To truncate the ideal response we multiply it by Kaiser window. The length of this window and parameter β is determined by using empirical formulae

Kaiser Window Parameters

$$2N + 1 \geq 1 + \frac{A - 8}{2.285\Delta\omega_{\mathrm{T}}}$$

where $A = -20\log_{10}\delta$ and $\Delta\omega_{\mathrm{T}} = \omega_s - \omega_p$
By substituting values we get, $N = 15, A = 18.0158, \alpha = 0\ and\ \beta = 0$
We take $N = 20$ to match specifications

Filter Response

Filter Coefficients [41] = [0.0193 0.0257 0.0078 -0.0023 0.0099 0.0153 -0.0123 -0.0465 -0.0389 0.0070 0.0339 0.0150 -0.0004 0.0356 0.0764 0.0264 -0.1102 -0.1888 -0.0853 0.1293 0.2400 0.1293 -0.0853 -0.1888 -0.1102 0.0264 0.0764 0.0356 -0.0004 0.0150 0.0339 0.0070 -0.0389 -0.0465 -0.0123 0.0153 0.0099 -0.0023 0.0078 0.0257 0.0193]

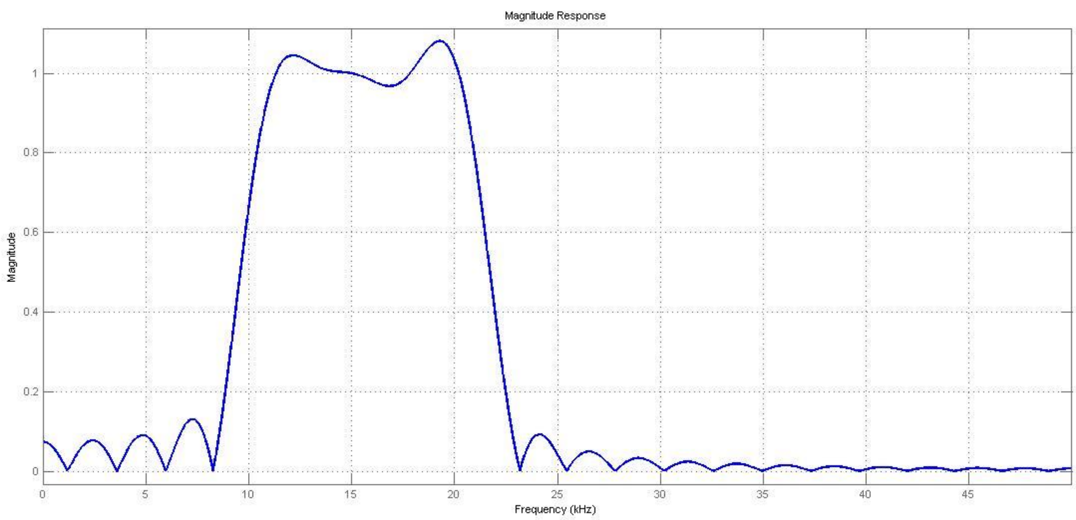

Fig6:Magnitude Response of FIR BPF

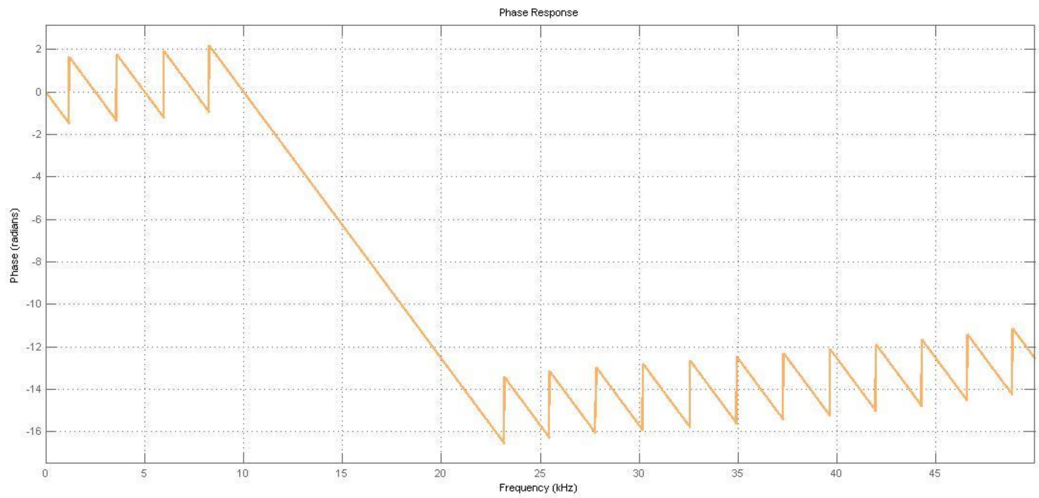

fig7:Phase Response for FIR BPF

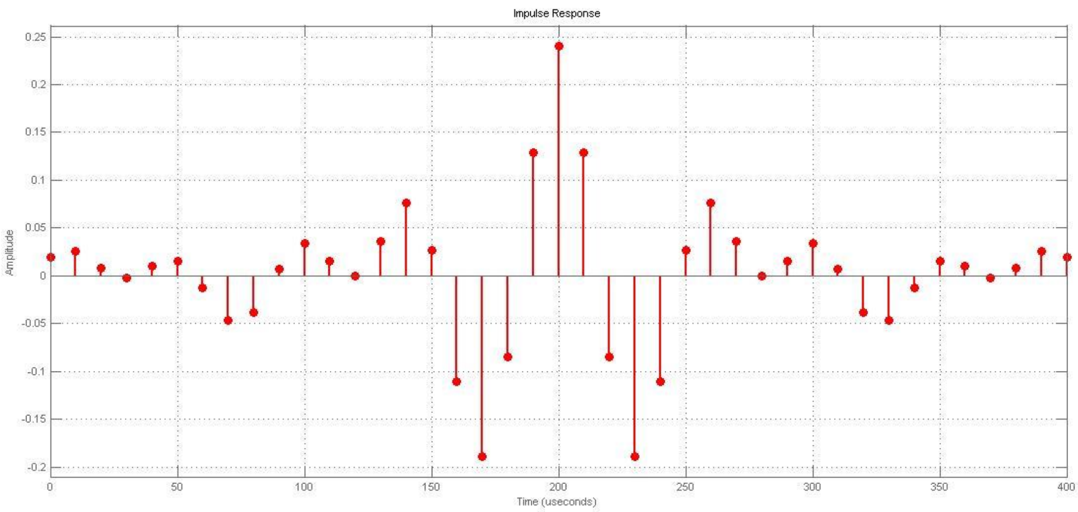

fig 8:Impulse Response for FIR BPF

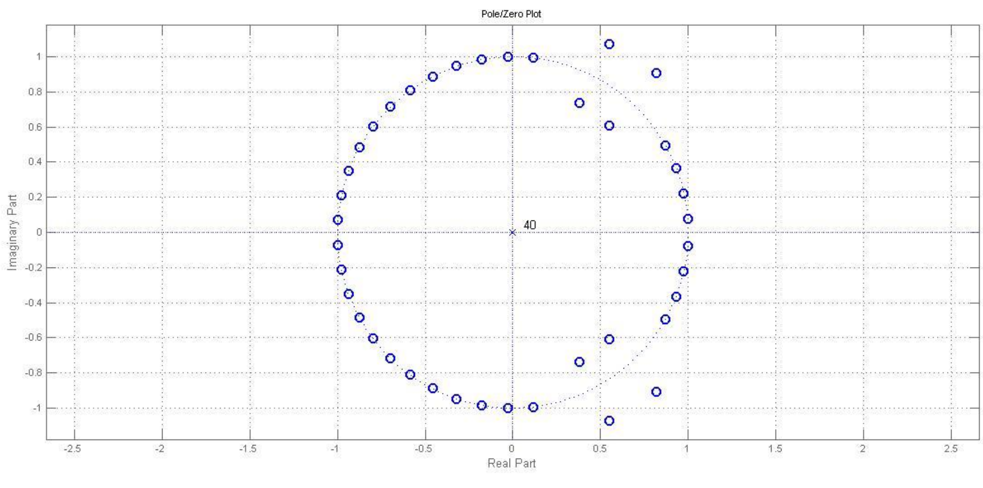

fig9:Zeros of FIR BPF

Filter 2

IIR Implementation

Equiripple in Passband and Monotonic Stopband

Digital Frequencies

$$f_{digital} = \frac{f_{analog}}{2\pi f_{sample}}$$

ω_{p1} = 0.5592 ω_{s1} = 0.6849

ω_{s2} = 1.3132 ω_{p2} = 1.4388

Equivalent Analog Frequencies

$$\Omega = \tan\left(\frac{\omega}{2}\right)$$

Ω_{p1} = 0.2871 Ω_{s1} = 0.3565

Ω_{s2} = 0.7707 Ω_{p2} = 0.8761

Low Pass Equivalent Filter

$$\Omega_l = \frac{B\Omega}{\Omega_0^2 - \Omega^2}$$

Where

$\Omega_0^2 = \Omega_{p1}\Omega_{p2}$ and $B = \Omega_{p2} - \Omega_{p1}$

Stringent of the two values obtained from bandpass specifications is chosen as stop band frequency for low pass filter. Thus specifications of low pass filter needed to be deigned are as follows

$$\Omega_{lp} = 1$$

$$\Omega_{ls} = 1.3256$$

Equiripple response is needed in passband and monotonic in stopband, hence Chebyshev Filter is designed

$$N \geq \frac{\cosh^{-1}\sqrt{\frac{D_2}{D_1}}}{\cosh^{-1}\frac{\Omega_{ls}}{\Omega_{lp}}}$$

Considering the minimum order required, we obtain $N = 4$

$$A_k = \frac{(2k+1)\pi}{2N}$$

$$B_k = \frac{\sinh^{-1}\frac{1}{\epsilon}}{N}$$

Poles are given by

$$p_k = \Omega_p sinA_k sinhB_k + i\Omega_p cosA_k coshB_k$$

Transfer function is given by

$$H(s) = \frac{0.2017}{s^4 + 0.8342\, s^3 + 1.348\, s^2 + 0.6243\, s + 0.2373}$$

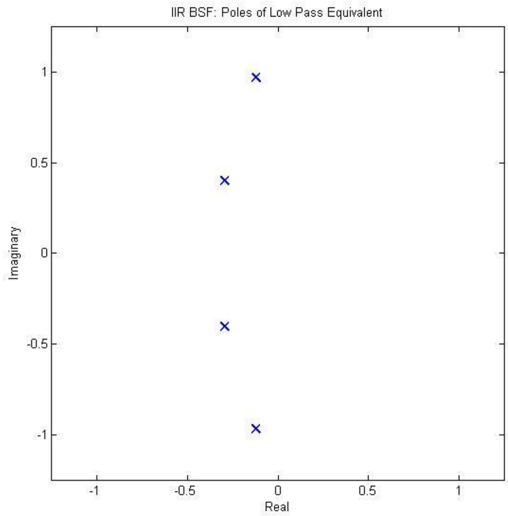

Fig10: Poles of Low Pass Equivalent of IIR BSF (Chebyshev)

Analog Bandpass Filter

To obtain analog bandstop filter, following transformation was carried out in the transfer function

$$s_l = \frac{Bs}{s^2 + \Omega_0^2}$$

After transformation, following transfer function was obtained

$$H(s) = \frac{0.85\, s^8 + 0.8552\, s^6 + 0.3227\, s^4 + 0.05411\, s^2 + 0.003403}{s^8 + 1.549\, s^7 + 2.976\, s^6 + 1.887\, s^5 + 1.878\, s^4 + 0.4747\, s^3 + 0.1883\, s^2 + 0.02466\, s + 0.004003}$$

Digital Filter

To obtain analog bandpass filter, following transformation was carried out in the transfer function

$$s = \frac{1 - z^{-1}}{1 + z^{-1}}$$

After transformation, following transfer function was obtained

$$H(z) = \frac{0.2089\, z^8 - 0.9995\, z^7 + 2.629\, z^6 - 4.428\, z^5 + 5.267\, z^4 - 4.428\, z^3 + 2.629\, z^2 - 0.9995\, z + 0.2089}{z^8 - 3.115\, z^7 + 5.066\, z^6 - 5.76\, z^5 + 4.999\, z^4 - 3.181\, z^3 + 1.598\, z^2 - 0.716\, z + 0.2114}$$

Images for the pole-zero plot, magnitude response and phase response obtained using fvtool are shown below

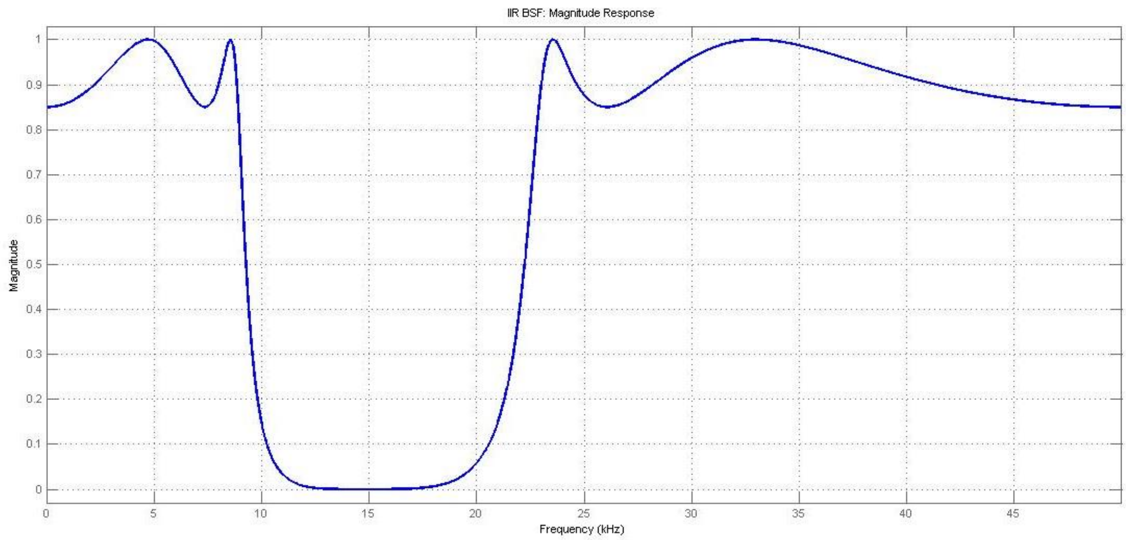

fig11: Magnitude Response of IIR BSF

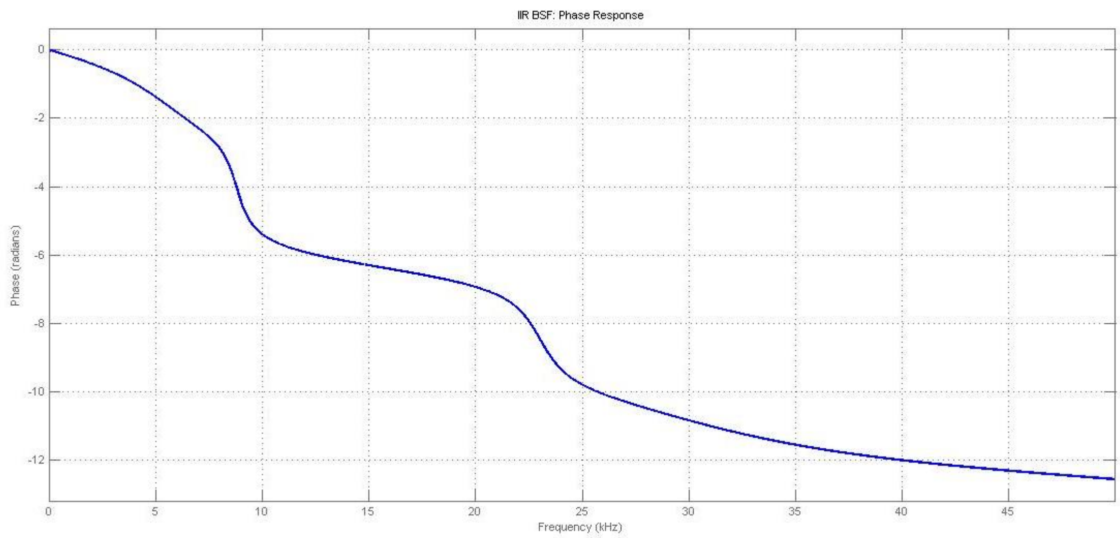

fig12: Phase Response of IIR BSF

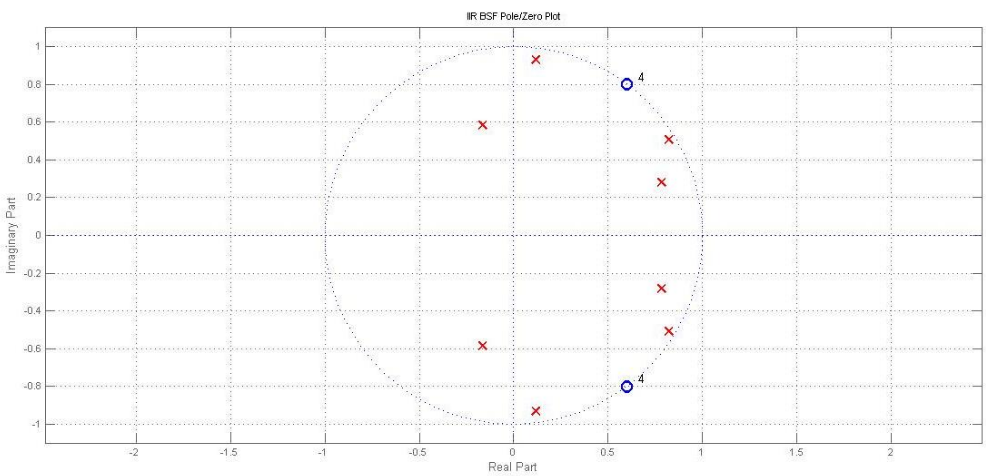

fig13: Pole-Zero Plot of IIR BSF

Direct Form 2

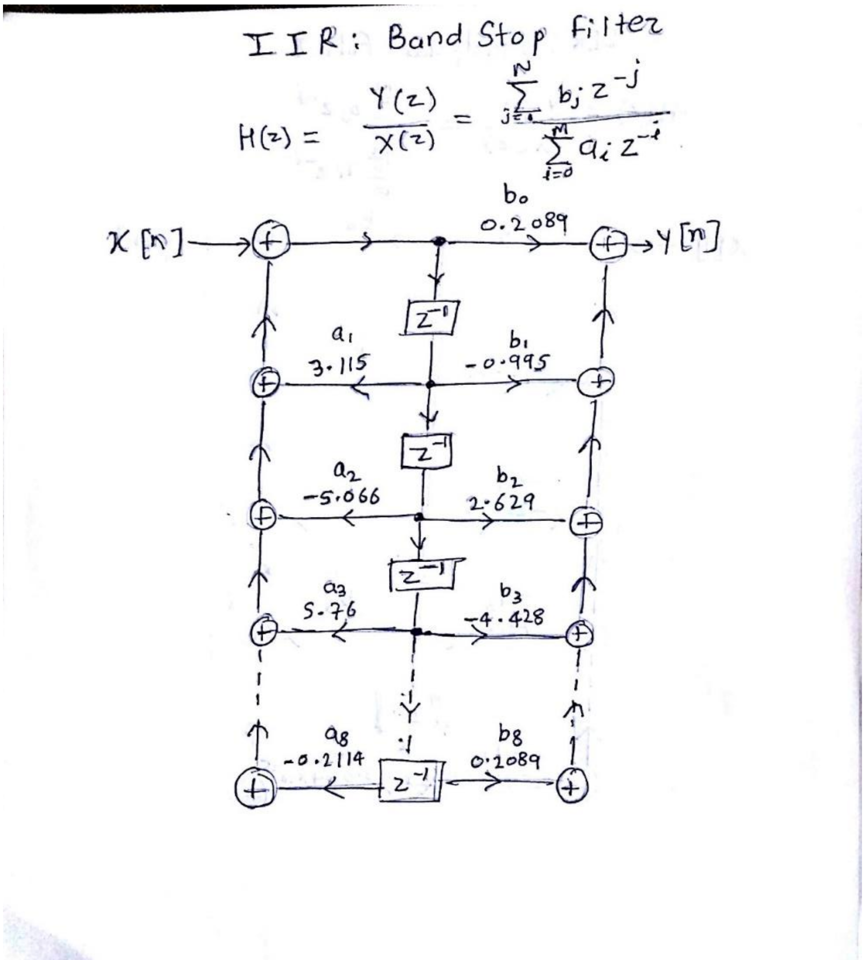

fig14: Direct Form 2 Representation of IIR BSF

FIR Implementation

Ideal Impulse Response

$$h_{ideal}[n] = \frac{\sin(k\omega_{c1}) - \sin(k\omega_{c2})}{\pi n} \qquad for\ n \neq 0$$

$$= 1 - \frac{\omega_{c2} - \omega_{c1}}{\pi} \qquad for\ n = 0$$

To truncate the ideal response we multiply it by Kaiser window. The length of this window and parameter β is determined by using empirical formulae

Kaiser Window Parameters

$$2N + 1 \geq 1 + \frac{A - 8}{2.285\Delta\omega_T}$$

where $A = -20\log_{10}\delta$ and $\Delta\omega_T = \omega_s - \omega_p$

By substituting values we get, $N = 18, A = 18.0158, \alpha = 0\ and\ \beta = 0$

Filter Response

Filter Coefficients[41] =

[-0.0129 -0.0256 -0.0110 0.0014 -0.0095 -0.0187 0.0058 0.0439 0.0436 -0.0003 -0.0316 -0.0160 0.0014 -0.0332 -0.0784 -0.0338 0.1043 0.1900 0.0903 -0.1268 0.7600 -0.1268 0.0903 0.1900 0.1043 -0.0338 -0.0784 -0.0332 0.0014 -0.0160 -0.0316 -0.0003 0.0436 0.0439 0.0058 -0.0187 -0.0095 0.0014 -0.0110 -0.0256 -0.0129]

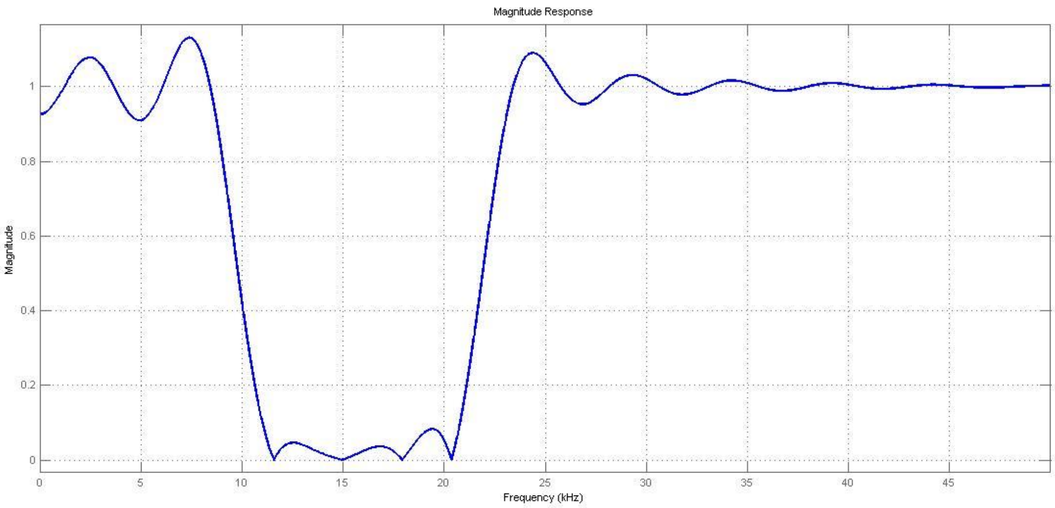

Fig15: Magnitude Response of FIR BSF

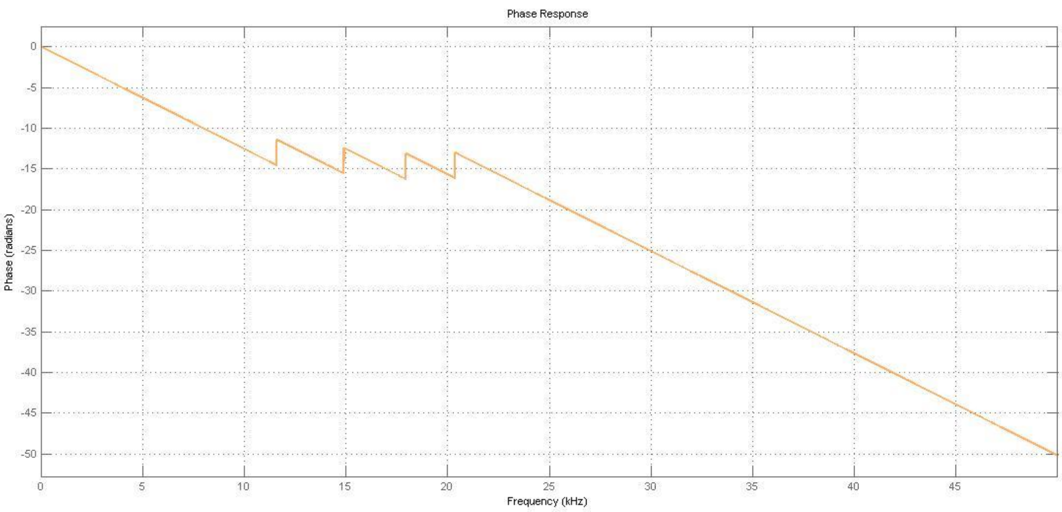

Fig16: Phase Response of FIR BSF

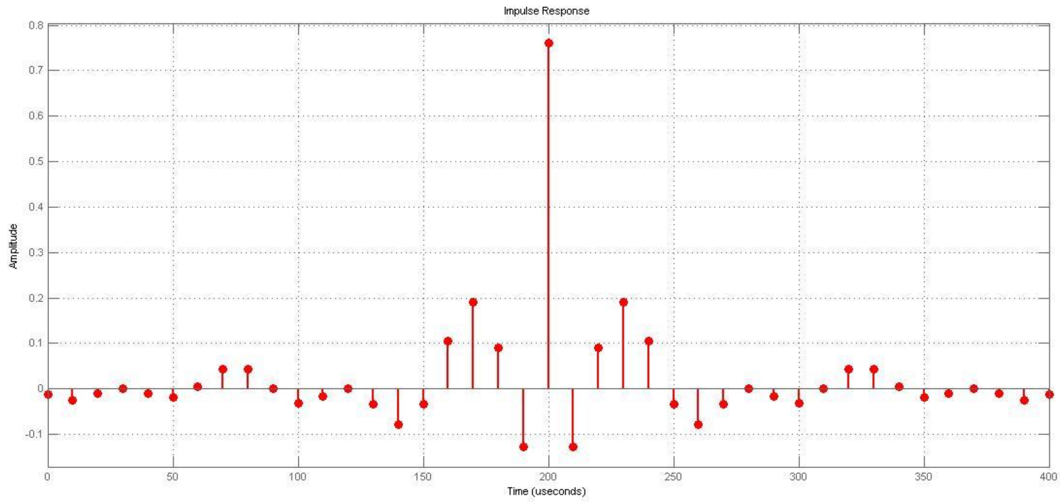

fig17: Impulse Response of FIR BSF

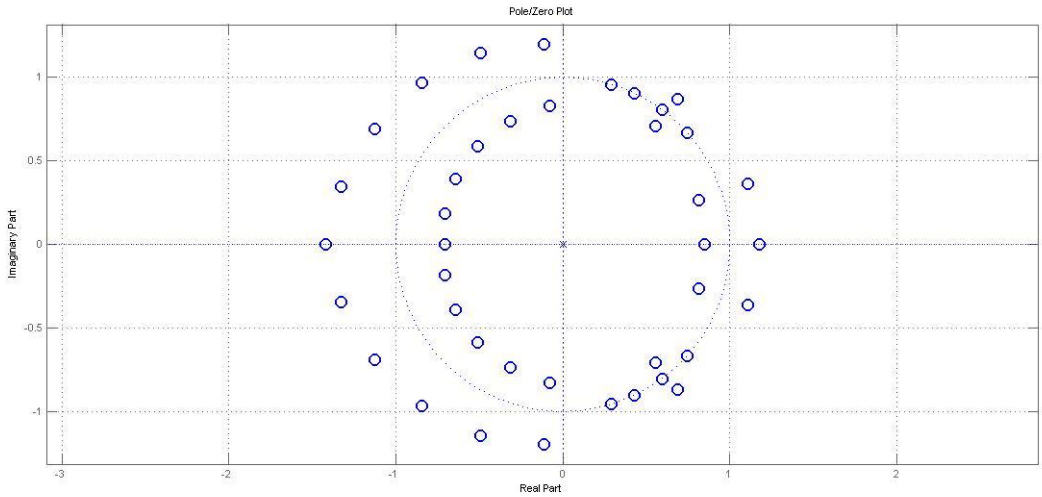

Fig18: Zeros of FIR BSF

Codes

IIR Bandpass Filter (Using Butterworth)

```
% % Kalpesh Patil - 130040019
% % filter number 13
% % BPF IIR
% % Butterworth

%finding cutoff frequencies
m                           =   13;
q_m                         =   floor(0.1*m);
r_m                         =   m - 10*q_m;
Bl_m                        =   4 + 0.7*q_m + 2*r_m;
Bh_m                        =   Bl_m + 10;
```

```
delta_1                 =   0.15;
delta_2                 =   0.15;
f_sample                =   100*1000;

omega_s1                =   (Bl_m-2)*1000;
omega_p1                =   (Bl_m)*1000;
omega_p2                =   (Bh_m)*1000;
omega_s2                =   (Bh_m+2)*1000;

f_analog_array          =   [omega_s1 omega_p1 omega_p2 omega_s2];

%normalized specs
f_digital_array         =   f_analog_array.*2*pi/f_sample;
f_eqv_analog            =   tan(f_digital_array/2);

%analog LPF
omega_not               =   sqrt(f_eqv_analog(2)*f_eqv_analog(3));
B                       =   f_eqv_analog(3) - f_eqv_analog(2);

%frequency transformation
f_eqv_analog_LPF        =   (f_eqv_analog.^2 -
omega_not^2)./(B*f_eqv_analog);


%designing this LPF
D1                      =   (1/(1-delta_1)^2)-1;
D2                      =   (1/delta_2^2)-1;
mod_f_eqv_analog_LPF    =   abs(f_eqv_analog_LPF);
stringent_omega_s       =
min(mod_f_eqv_analog_LPF(1),mod_f_eqv_analog_LPF(4));
temp                    =
log(sqrt(D2)/sqrt(D1))/log(stringent_omega_s/f_eqv_analog_LPF(3));
N                       =   ceil(temp);

%finding poles of equivalent low pass filter
omega_p                 =   f_eqv_analog_LPF(3);
omega_c                 =
((omega_p/(D1^(1/(2*N))))+(stringent_omega_s/(D2^(1/(2*N)))))/2;
thetas                  =   (2*[1:2*N] - 1)*(pi)/(2*N);

gain_LF                 =   omega_c^N;
poles                   =   1i*omega_c*exp(1i*thetas);
poles_LF                =   poles(1:N);
zeros_LF                =   [];
[num_LF,den_LF]         =   zp2tf(zeros_LF,poles_LF,gain_LF);
tf_LF                   =   tf(num_LF,den_LF);

%converting back to band pass filter
poles_BF                =   zeros(1,2*N);
poles_BF(1:N)           =   (B/2).*(poles_LF-sqrt(poles_LF.^2 -
4*omega_not^2/(B^2)));
poles_BF(N+1:2*N)       =   (B/2).*(poles_LF+ sqrt(poles_LF.^2 -
4*omega_not^2/(B^2)));
zeros_BF                =   zeros(1,N);
gain_BF                 =   gain_LF*B^N;
[num_BF,den_BF]         =   zp2tf(zeros_BF',poles_BF,gain_BF);
tf_BF                   =   tf(num_BF, den_BF);

%converting to z domain
```

```
poles_BF_z                =   (1 + poles_BF)./(1 - poles_BF);
temp                      =   prod(1-poles_BF);
gain_BF_z                 =   gain_BF/temp;
zeros_BF_z                =   [ones(1,N),-ones(1,N)];
[num_BF_z, den_BF_z]      =   zp2tf(zeros_BF_z',poles_BF_z,gain_BF_z);
DigitalFilter             =   tf(num_BF_z, den_BF_z,1/f_sample);
h                         =   fvtool(num_BF_z, den_BF_z);
```

FIR Bandpass Filter

```
% % Kalpesh Patil - 130040019
% % filter number 13
% % BPF FIR

m                       = 13;
q_m                     = floor(0.1*m);
r_m                     = m - 10*q_m;
Bl_m                    = 4 + 0.7*q_m + 2*r_m;
Bh_m                    = Bl_m + 10;

delta_1                 = 0.15;
delta_2                 = 0.15;
f_sample                = 100*1000;

omega_s1                = (Bl_m-2)*1000;
omega_p1                = (Bl_m)*1000;
omega_p2                = (Bh_m)*1000;
omega_s2                = (Bh_m+2)*1000;

f_analog_array          = [omega_s1 omega_p1 omega_p2 omega_s2];

%normalized specs
f_digital_array         = f_analog_array.*2*pi/f_sample;

% Kaiser window parameters
del_omega1              = f_digital_array(4)-f_digital_array(3);
del_omega2              = f_digital_array(2)-f_digital_array(1);
del_omega               = min(abs(del_omega1),abs(del_omega2));
A                       = -20*log10(del_omega);

% Order
N_1                     =  (A-8)/(2*2.285*del_omega);
N_limit                 =  ceil(N_1);
N                       =  N_limit + 5;

if(A<21)
     alpha=0;
else if(A<=50)
        alpha           =  0.5842*(A-21)^0.4+0.07886*(A-21);
     else
        alpha           =  0.1102*(A-8.7);
     end
end

omega_c1                =  (f_digital_array(2)+f_digital_array(1))*0.5;
omega_c2                =  (f_digital_array(4)+f_digital_array(3))*0.5;
```

```
h_ideal = [];
for k = -N:N
    if(k~=0)
        h_ideal(k+N+1) = (sin(omega_c2*k)-sin(omega_c1*k))/(pi*k);
    else
        h_ideal(k+N+1) =  0;
    end
end

h_ideal(N+1)            =   ((omega_c2-omega_c1)/pi);
beta                    =   alpha/N;

% Generating Kaiser window
h_kaiser                =   kaiser(2*N+1,beta);
h_org                   =   h_ideal.*h_kaiser';
fvtool(h_org);
```

IIR Bandstop Filter (Using Chebyshev)

```
% % Kalpesh Patil - 130040019
% % filter number 13
% % BSF IIR
% % Chebyshev

%finding cutoff frequencies
m                         =    13;
q_m                       =    floor(0.1*m);
r_m                       =    m - 10*q_m;
Bl_m                      =    4 + 0.9*q_m + 2*r_m;
Bh_m                      =    Bl_m + 10;

delta_1                   =    0.15;
delta_2                   =    0.15;
f_sample                  =    100*1000;

omega_p1                  =    (Bl_m-2)*1000;
omega_s1                  =    (Bl_m)*1000;
omega_s2                  =    (Bh_m)*1000;
omega_p2                  =    (Bh_m+2)*1000;

f_analog_array            =    [omega_p1 omega_s1 omega_s2 omega_p2];

%normalized specs
f_digital_array           =    f_analog_array.*2*pi/f_sample;
f_eqv_analog              =    tan(f_digital_array/2);

%analog LPF
omega_not                 =    sqrt(f_eqv_analog(1)*f_eqv_analog(4));
B                         =    f_eqv_analog(4) - f_eqv_analog(1);

%frequency transformation
f_eqv_analog_LPF          =    (B*f_eqv_analog)./(omega_not^2 -
f_eqv_analog.^2);

%designing this LPF
D1                        =    (1/(1-delta_1)^2)-1;
D2                        =    (1/delta_2^2)-1;
```

```
epsilon                     =   sqrt(D1);
mod_f_eqv_analog_LPF        =   abs(f_eqv_analog_LPF);
stringent_omega_s           =
min(mod_f_eqv_analog_LPF(2),mod_f_eqv_analog_LPF(3));
temp                        =
acosh(sqrt(D2)/sqrt(D1))/acosh(stringent_omega_s/f_eqv_analog_LPF(1));
N                           =   ceil(temp);


%finding poles of equivalent low pass filter
omega_p                     =   mod_f_eqv_analog_LPF(4);
Ak                          =   ([1:2*N]*2 + 1)*pi/(2*N);
Bk                          =   asinh(1/epsilon)/N;
temp_real                   =   omega_p*sin(Ak)*sinh(Bk);
temp_imag                   =   omega_p*cos(Ak)*cosh(Bk);
temp                        =   temp_real + 1i*temp_imag;
poles_LF                    =   zeros(1,N);
t                           =   1;
for k = 1:2*N
    if(temp_real(k) < 0)
      poles_LF(t)           =   temp(k);
      t                     =   t+1;
    end
end

if (mod(N,2) == 0)
    d                       =   1/sqrt(1+epsilon^2);
else
    d                       =   1;
end

g                           =   ((-1)^N)*prod(poles_LF);
gain_k                      =   d*g;
zeros_LF                    =   [];
[num_LF,den_LF]             =   zp2tf(zeros_LF',poles_LF,gain_k);
tf_LF                       =   tf(num_LF,den_LF);

%converting back to band stop filter
poles_BF                    =   zeros(1,2*N);
poles_BF(1:N)               =   (B/2).*(1./poles_LF + sqrt(1./poles_LF.^2 -
4*omega_not^2/(B^2)));
poles_BF(N+1:2*N)           =   (B/2).*(1./poles_LF - sqrt(1./poles_LF.^2 -
4*omega_not^2/(B^2)));
zeros_BF                    =   [1i*omega_not*ones(1,N)' ; -
1i*omega_not*ones(1,N)'];
gain_BF                     =   gain_k/g;
[num_BF,den_BF]             =   zp2tf(zeros_BF,poles_BF,gain_BF);
tf_BF                       =   tf(num_BF, den_BF);


%converting to z domain
poles_BF_z                  =   (1 + poles_BF)./(1 - poles_BF);
temp                        =   prod(1-poles_BF);
gain_BF_z                   =   gain_BF*((1+omega_not^2)^N)/(prod(1-poles_BF));
zeros_BF_z                  =   (1 + zeros_BF)./(1 - zeros_BF);
[num_BF_z, den_BF_z]        =   zp2tf(zeros_BF_z,poles_BF_z,gain_BF_z);
DigitalFilter               =   tf(num_BF_z, den_BF_z,1/f_sample);
h                           =   fvtool(num_BF_z, den_BF_z);
```

FIR Bandstop Filter

```
% % Kalpesh Patil - 130040019
% % filter number 13
% % BSF FIR

m                       = 13;
q_m                     = floor(0.1*m);
r_m                     = m - 10*q_m;
Bl_m                    = 4 + 0.9*q_m + 2*r_m;
Bh_m                    = Bl_m + 10;

delta_1                 = 0.15;
delta_2                 = 0.15;
f_sample                = 100*1000;

omega_p1                = (Bl_m-2)*1000;
omega_s1                = (Bl_m)*1000;
omega_s2                = (Bh_m)*1000;
omega_p2                = (Bh_m+2)*1000;

f_analog_array          = [omega_p1 omega_s1 omega_s2 omega_p2];

%normalized specs
f_digital_array         = f_analog_array.*2*pi/f_sample;

% Kaiser window parameters
del_omega1              = f_digital_array(4)-f_digital_array(3);
del_omega2              = f_digital_array(2)-f_digital_array(1);
del_omega               = min(abs(del_omega1),abs(del_omega2));
A                       = -20*log10(del_omega);

% Order
N_1                     =  (A-8)/(2*2.285*del_omega);
N_limit                 =  ceil(N_1);
N                       =  N_limit + 5;

if(A<21)
     alpha=0;
else if(A<=50)
        alpha           =  0.5842*(A-21)^0.4+0.07886*(A-21);
     else
        alpha           =  0.1102*(A-8.7);
     end
end

omega_c1                =  (f_digital_array(2)+f_digital_array(1))*0.5;
omega_c2                =  (f_digital_array(4)+f_digital_array(3))*0.5;

h_ideal = [];
for k = -N:N
    if(k~=0)
        h_ideal(k+N+1) = - (sin(omega_c2*k)-sin(omega_c1*k))/(pi*k);
    else
        h_ideal(k+N+1) =   0;
    end
end
```

```
h_ideal(N+1)            =  1 - ((omega_c2-omega_c1)/pi);
beta                    =  alpha/N;

% Generating Kaiser window
h_kaiser                =  kaiser(2*N+1,beta);
h_org                   =  h_ideal.*h_kaiser';
fvtool(h_org);
```