Digital IIR Filter Design

Hemanth Kumar Desineedi and G V V Sharma*

Contents

1 **Analog Filter Specifications** 1

2 **Digital Filter Specification** 1
2.1 Bilinear Tranformation . . . 1

3 **IIR Low Pass Analog Filter** 2

4 **IIR Band Pass Analog Filter** 4

5 **IIR Digital Band Pass Filter** 5

***Abstract*—The process of designing analog and digital filters is explained by designing a band-pass filter. IIR LPF,BPF filters are designed. All computations are done in Python. The Chebychev function is employed for designing the IIR filter.**

1 Analog Filter Specifications

The filter specifications are available in Table I.

2 Digital Filter Specification

2.1 Bilinear Tranformation

Problem 2.1. Obtain the fiter specifications in the ω domain.

Solution:

$$\omega = 2\pi\frac{F}{F_s}. \tag{2.1.1}$$

Problem 2.2. The Bilinear transformation is given by

$$s = \frac{1-z^{-1}}{1+z^{-1}} \tag{2.2.1}$$

If $s = \jmath\Omega, z = e^{\jmath\omega}$, show that

$$\Omega = \tan\frac{\omega}{2} \tag{2.2.2}$$

*The author is with the Department of Electrical Engineering, Indian Institute of Technology, Hyderabad 502285 India e-mail: gadepall@iith.ac.in.

Item	**Symbol**	**Value**
Sampling Rate	F_s	48 kHz
Pass Band Tolerance	δ_1	0.15
Stop Band Tolerance	δ_2	0.15
Passband Lower Cutoff Frequency	F_{p2}	6 kHz
Passband Higher Cutoff Frequency	F_{p1}	7.2 kHz
Transition Band	ΔF	0.3 kHz
Stopband Lower Cutoff Frequency	F_{s2}	5.7 kHz
Stopband Higher Cutoff Frequency	F_{s1}	7.5 kHz

TABLE I

Problem 2.3. Write a Python function to obtain the filter specifications in the Ω domain using (2.2.2) and save it in the file **F_omega.py**.

Solution:

```
import numpy as np
def f_Omega(Fp1,Fp2,df,Fs):
        def F_w(F_s,F):
                w = 2*np.pi*float(
                   F)/F_s
                ap=np.tan(w/2.0)
                return ap

        #passband Omega cutoffs
        ap1=F_w(Fs,Fp1)
        ap2=F_w(Fs,Fp2)

        #stopband Omega cutoffs
        as1=F_w(Fs,Fp1+df)
        as2=F_w(Fs,Fp2-df)
```

```
        return ap1 ,ap2 ,as1 ,as2
```

3 IIR Low Pass Analog Filter

Problem 3.1. Transform the filter specifications from the bandpass Ω domain to the low pass Ω_L domain.

Solution:

$$\Omega_L = \frac{\Omega^2 - \Omega_0^2}{B\Omega}, \text{ where} \tag{3.1.1}$$

$$B = \Omega_{p1} - \Omega_{p2}, \Omega_0 = \sqrt{\Omega_{p1}\Omega_{p2}} \tag{3.1.2}$$

Problem 3.2. For $\Omega_{s1}\Omega_{s2}$, (3.3.1) yields Ω_{Ls1},Ω_{Ls2}. Obtain Ω_{Ls}, the low pass stopband cutoff from Ω_{Ls1},Ω_{Ls2}.

Solution:

$$\Omega_{Ls} = \min\left(|\Omega_{Ls1}|, |\Omega_{Ls2}|\right). \tag{3.2.1}$$

Problem 3.3. Obtain Ω_{Lp}.

Solution:

$$\Omega_{Lp} = \frac{\Omega_{p1}^2 - \Omega_0^2}{B\Omega} \tag{3.3.1}$$

Problem 3.4. Write a function in Python to obtain $B, \Omega_0, \Omega_{Lp}, \Omega_{Ls}$ from $\Omega_{p1}, \Omega_{p2}, \Omega_{s1}, \Omega_{s2}$. Save it in the file **Omegabp_Omegalp.py**.

Solution:

```
import numpy as np
def Omega_blp(ap1 ,ap2 ,as1 ,as2):

        # 1.Low pass Filter
            Specifications
        ao=np.sqrt(ap1*ap2)
        B=ap1-ap2
        alp=(ap1**2 - ao**2) /(B*
            ap1)
        als1=(as1**2 - ao**2) / (B
            *as1)
        als2=(as2**2 - ao**2) / (B
            *as2)
        als=np.minimum(abs(als1),
            abs(als2))
        return ao,B,alp,als
```

Problem 3.5. The magnitude squared of the Chebychev low pass filter is

$$|H_{LP}(j\Omega_L)|^2 = \frac{1}{1 + \varepsilon^2 c_N^2(\Omega_L/\Omega_{Lp})}, \tag{3.5.1}$$

where

$$c_N(x) = \cosh(N \cosh^{-1} x) \tag{3.5.2}$$

The filter parameters are defined to be N and ϵ. Express these in terms of δ_1 and δ_2 defined in Table I.

Solution:

$$\frac{\sqrt{D_2}}{c_N(\Omega_{Ls})} \leq \varepsilon \leq \sqrt{D_1} \tag{3.5.3}$$

$$N \geq \left\lceil \frac{\cosh^{-1}(\sqrt{D_2/D_1})}{\cosh^{-1}(\Omega_{Ls})} \right\rceil \tag{3.5.4}$$

where

$$D_1 = \frac{1}{(1-\delta_1)^2} - 1, D_2 = \frac{1}{\delta_2^2} - 1 \tag{3.5.5}$$

Problem 3.6. Write a Python function to obtain $N, \varepsilon, \Omega_{Lp}$ and save it in the file titled **N_epsilon.py**.

Solution:

```
import numpy as np
import math

from F_omega import f_Omega
from Omegabp_Omegalp import
    Omega_blp

def parameters(Fp2,Fp1,df,d,Fs):

        ap1,ap2,as1,as2=f_Omega(
            Fp1,Fp2,df,Fs)

        ao,B,alp,als=Omega_blp(ap1
            ,ap2,as1,as2)

        #  Low Pass Chebyshev
            filter parameters
        D1=(1/(1-d)**2)-1
        D2=(1/(d**2))-1
        cn=np.cosh(4*np.arccosh(
            als))
        R1=np.sqrt(D2) / cn   #Min
            Range
        R2=np.sqrt(D1)        #Max
            Range
```

```
            N=int(math.ceil((np.arccosh(np.sqrt(D2/D1))/ np.arccosh(als)))

            return N,R1,R2,alp
```

Problem 3.7. Given N and ε, find the analog IIR low pass Chebychev filter.

Solution:

$$H_{LP}(s)=\begin{cases}\frac{\Omega_{Lp}^{N}C_0\prod_{k=1}^{(N-1)/2}C_k}{(s+\Omega_{Lp}C_0)\prod_{k=1}^{(N-1)/2}(s^2+b_k\Omega_{Lp}s+C_k\Omega_{Lp}^2)} & N \text{ odd}\\ \frac{\Omega_{Lp}^{N}\prod_{k=1}^{(N)/2}C_k\frac{1}{\sqrt{1+\varepsilon^2}}}{\prod_{k=1}^{(N)/2}(s^2+b_k\Omega_{Lp}s+C_k\Omega_{Lp}^2)} & N \text{ even}\end{cases} \tag{3.7.1}$$

where

$$C_0 = y_N \tag{3.7.2}$$

$$C_k = y_N^2+\cos^2((2k-1)\pi/2N) \tag{3.7.3}$$

$$b_k = 2y_N\sin((2k-1)\pi/2N) \tag{3.7.4}$$

$$y_N=\frac{1}{2}\left[\left[\sqrt{1+\frac{1}{\varepsilon^2}}+\frac{1}{\varepsilon}\right]^{1/N}-\left[\sqrt{1+\frac{1}{\varepsilon^2}}+\frac{1}{\varepsilon}\right]^{-1/N}\right] \tag{3.7.5}$$

Problem 3.8. Write a function for computing coefficients of numerator and denominator of (3.7.1) and save it in the file titled **Gen_LPF_Coeff.py**.

Solution:

```
import numpy as np

def filt_design(N,e,alp):
    yN= (1/2.0)*((np.sqrt(1+(1/e**2))+(1/e))**(1/N)-(np.sqrt(1+(1/e**2))+(1/e))**(-1/N))
    C0=yN
    C=[]
    b=[]
    for k in range(1,(N/2)+1):
        C.append(yN**2+np.cos((2*k-1)*np.pi/(2.0*N))**2)
        b.append(2*yN*np.sin((2*k-1)*np.pi/(2.0*N)))


    tot_ck=1
    pole_sum=1
    if(N%2==0):
                for k in range(N/2):
                        tot_ck=tot_ck*C[k]
                        pole_sum=pole_sum*np.poly1d([1,b[k]*alp,C[k]*(alp**2)])
                tot_ck=np.poly1d([0,1])*(alp**N)*tot_ck*np.sqrt(1/(1+e**2))
                #pole_sum=np.array(pole_sum)
    else:
                for k in range(N/2):
                        tot_ck=tot_ck*C[k]
                        pole_sum=pole_sum*np.poly1d([1,b[k]*alp,C[k]*(alp**2)])
                tot_ck=np.poly1d([0,1])*(alp**N)*tot_ck*C0
                pole_sum=pole_sum*np.poly1d([1,alp*C0])

    return tot_ck,pole_sum
```

Problem 3.9. Plot (3.7.1) for $N = 4$ and different values of ε.

Solution:

```
import numpy as np
import matplotlib.pyplot as plt
```

```
from N_epsilon import parameters
from Gen_LPF_Coeff import
    filt_design

#Passband Cutoffs
Fp2=6.0
Fp1=7.2

#Transition Band
df=0.3

#Passband && stopband tolerance
d=0.15

#sampling rate
Fs=48

N,R1,R2,alp = parameters(Fp2,Fp1,
    df,d,Fs)

e_array=np.linspace(R1,R2,6)

Omga_L=np.linspace(0,2,200)

s=1j*Omga_L

for e in e_array:
        b,a=filt_design(N,e,alp)

        H_lp=np.polyval(b,s)/np.
            polyval(a,s)

        plt.plot(Omga_L,abs(H_lp),
            label=round(e,2))

plt.xlabel('$Frequency(\Omega)$')
plt.ylabel('$|H_{lp}(j\Omega)|$')
plt.title('Magnitude response of
    LPF for different values of $\
    epsilon$')
plt.legend()
plt.grid()
plt.show()
```

Problem 3.10. What value of ε should be selected?

Solution: There is a trade-off between passband and stopband frequencies. So select the ε value which satisfies only passband frequency. For the above example we assume ε=0.6197

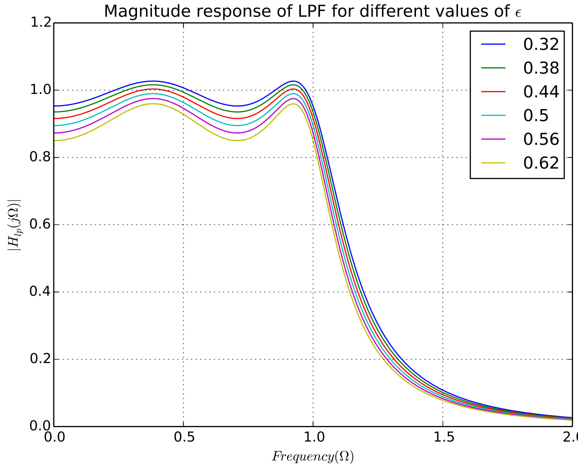

Fig. 3.9

4 IIR Band Pass Analog Filter

Problem 4.1. Obtain the band pass filter from low pass filter in (3.7.1)

Solution:

The analog band pass filter can be obtained from analog low pass filter as

$$H_{BP}(s) = H_{LP}(s_L)\Big|_{s_L=\frac{s^2+\Omega_0^2}{Bs}} \tag{4.1.1}$$

Problem 4.2. Write a Python script for computing coefficients of analog BPF for $N = 4, \varepsilon = 0.6197$ and plot the magnitude response.

Solution:

```
import numpy as np
import matplotlib.pyplot as plt

from F_omega import f_Omega
from Omegabp_Omegalp import
    Omega_blp
from Gen_LPF_Coeff import
    filt_design

#Passband Cutoffs
Fp2=6.0
Fp1=7.2

#Transition Band
df=0.3

#Passband && stopband tolerance
d=0.15
```

```
#sampling rate
Fs=48

ap1,ap2,as1,as2=f_Omega(Fp1,Fp2,df
    ,Fs)
ao,B,alp,als = Omega_blp(ap1,ap2,
    as1,as2)


N=4
e=0.6197

#coefficients of numerator and
    denominator polynomials of LPF
    filter
b,a=filt_design(N,e,alp)
b=np.array(b)
a=np.array(a)


a1=0

for i in range(N+1):
        coeff=a[i]*(B**i)*(np.
            poly1d([1,0]))**(i)
        poly_denom=(np.poly1d
            ([1,0,ao**2]))**(N-i)
        a1=a1+poly_denom*coeff
numer=(np.poly1d([1,0]))**(N)
b1=numer*(B**N)*b

np.savetxt('Numerator_ABPF',np.
    array(b1))
np.savetxt('Denominator_ABPF',np.
    array(a1))

Omga_bp=np.linspace(-1,1,200)
s=1j*Omga_bp

H_bp=np.polyval(b1,s)/np.polyval(
    a1,s)

s1=np.array([1j*ap1,1j*ap2,1j*as1
    ,1j*as2])
H_point=np.polyval(b1,s1)/np.
    polyval(a1,s1)

pt=[5,-80,5,-80]
for i in range(len(s1)):
        A=[np.around(abs(s1[i]),
```

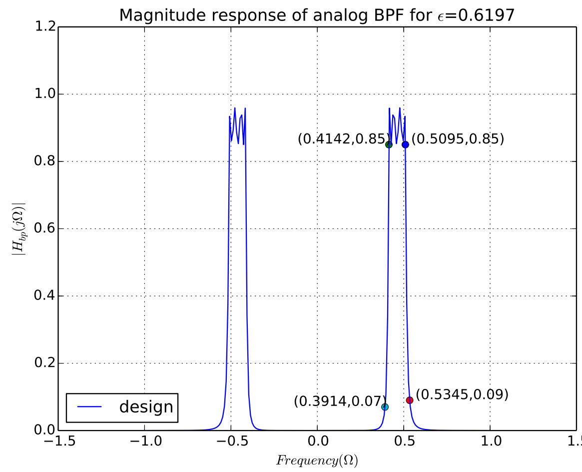

Fig. 4.2

```
            decimals=4)]
        B=[np.around(abs(H_point[i
            ]),decimals=2)]
        plt.plot(A,B,'o')

        for xy in zip(A,B):
                plt.annotate('(%s
                    ,%s)'%xy,xy=xy,
                    xytext=(pt[i],1)
                    ,textcoords='
                    offset points')
plt.plot(Omga_bp,abs(H_bp),'b',
    label='design')
plt.xlabel('$Frequency(\Omega)$')
plt.ylabel('$|H_{bp}(j\Omega)|$')
plt.title('Magnitude response of 
    analog BPF for $\epsilon$=0.6197
    ')
plt.axis([-1.5,1.5,0,1.2])
plt.legend(loc='lower left')
plt.grid()
plt.show()
```

5 IIR Digital Band Pass Filter

Problem 5.1. Obtain the IIR digital band pass filter from corresponding analog filter using Bilinear transformation.

Solution: The IIR digital BPF can be obtained from the analog BPF as

$$H_{BP}(z) = H_{BP}(s)\Big|_{s=\frac{1-z^{-1}}{1+z^{-1}}} \tag{5.1.1}$$

Problem 5.2. Write a Python script for finding the digital BPF coefficinets from the analog BPF coefficients and plot the magnitude response of the digital BPF.

Solution:

```
import numpy as np
import matplotlib.pyplot as plt

#Passband Cutoffs
Fp2=6.0
Fp1=7.2

#Transition Band
df=0.3

#Passband && stopband tolerance
d=0.15

#sampling rate
Fs=48

def W(Fs,F):
    w=2*float(F)/Fs
    return w

wp1=W(Fs,Fp1)
wp2=W(Fs,Fp2)
ws1=W(Fs,Fp1+df)
ws2=W(Fs,Fp2-df)

N=4
b1=np.loadtxt('Numerator_ABPF')
a1=np.loadtxt('Denominator_ABPF')



a2=0
for i in range(2*N+1):
        coeff1=a1[i]*np.poly1d
            ([1,1])**i
        poly_denom1=np.poly1d
            ([-1,1])**(2*N-i)
        a2=a2+poly_denom1*coeff1
b2=(np.poly1d([-1,1])**N)*(np.
    poly1d([1,1])**N)*b1[0]

np.savetxt('Numerator_DPBF',np.
    array(b2))
np.savetxt('Denominator_DBPF',np.
    array(a2))

w_bp=np.linspace(-0.5,0.5,200)

z_inv=np.exp(-1j*w_bp*np.pi)

H_bp=np.polyval(b2,z_inv)/np.
    polyval(a2,z_inv)

w_point=np.array([wp1,wp2,ws1,ws2
    ])

z_inv1=np.exp(-1j*w_point*np.pi)

H_point=np.polyval(b2,z_inv1)/np.
    polyval(a2,z_inv1)


pt=[1,-80,1,-80]
for i in range(len(w_point)):
        A=[np.around(w_point[i],
            decimals=4)]
        B=[np.around(abs(H_point[i
            ]),decimals=2)]
        plt.plot(A,B,'o')

        for xy in zip(A,B):
                plt.annotate('(%s
                    ,%s)'%xy,xy=xy,
                    xytext=(pt[i],1)
                    ,textcoords='
                    offset points')

plt.plot(w_bp,abs(H_bp),'b',label=
    'design')
plt.xlabel('$Frequency(\omega/\pi)
    $')
plt.ylabel('$|H_{bp}(\omega)|$')
plt.title('Magnitude response of 
    digital BPF for $\epsilon$
    =0.6197')
plt.axis([-1,1,0,1])
plt.legend()
plt.grid()
plt.show()
```

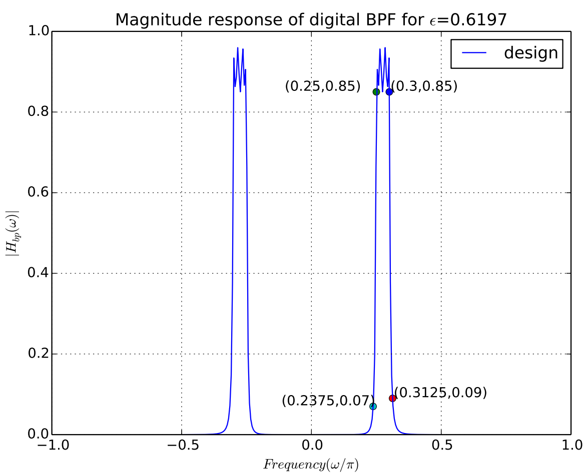
Magnitude response of digital BPF for ϵ=0.6197
design
(0.25,0.85)
(0.3,0.85)
(0.2375,0.07)
(0.3125,0.09)
$|H_{bp}(\omega)|$
$Frequency(\omega/\pi)$
1.0
0.8
0.6
0.4
0.2
0.0
−1.0
−0.5
0.0
0.5
1.0

Fig. 5.2