Chapter 9

Digital Filter Design

Contents

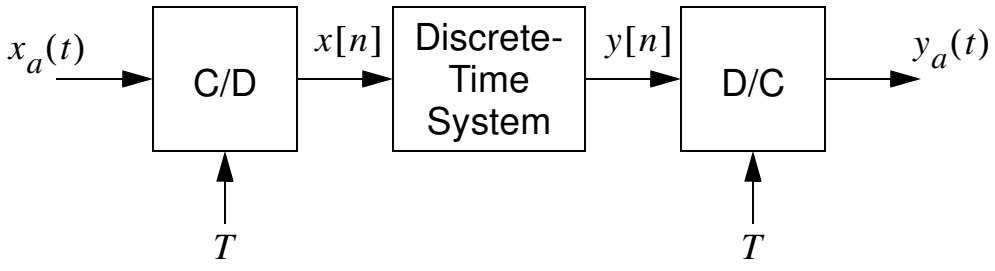
A filter is a frequency selective LTI system, that is a system that passes specified frequency components and rejects others. The discrete-time filter realizations of interest here are those LTI systems which have LCCDE representation and are causal. Note that for certain applications noncausal filters are appropriate.

An important foundation for digital filter design are the *classical* analog filter approximations. An overview of analog approximation techniques will be provided first.

The filter design problem can be grouped into three stages:

- Specification of the desired system properties (application driven)
- Approximation of the specifications using causal discrete-time systems
- System realization (technology driven - hardware/software)

This chapter will discuss primarily the approximation techniques. Realization techniques have been presented in part earlier. A common senario in which one finds a digital filter is in the filtering of a continuous-time signal using an A/D-$H(z)$-D/A system (earlier called a C/D-$H(e^{j\omega})$-D/C system).



A/D-$H(z)$-D/A system

Strictly speaking $H(z)$ is a discrete-time filter although it is commonly referred to as a *digital filter*

- Recall that for the continuous-time system described above (ideally)

$$H_{\text{eff}}(j\,\Omega) = \begin{cases} H(e^{j\,\Omega T}), & |\Omega| < \pi/T \\ 0, & |\Omega| \geq \pi/T \end{cases}$$

- Using the change of variables $\omega = \Omega T$ we can easily convert continuous-time specifications to discrete-time specifications i.e.

$$H(e^{j\omega}) = H_{\text{eff}}\left(j\,\frac{\omega}{T}\right), \quad |\omega| < \pi$$

9.1 Overview of Approximation Techniques

Digital filter design techniques fall into either IIR or FIR approaches

9.1.1 Approximation Approaches

- Placement of poles and zeros (ad-hoc)
- Numerical solution of differential equations
- Impulse invariant (step invariant etc.)
- Bilinear transformation
- Minimum mean-square error (frequency domain)

9.1.2 FIR Approximation Approaches

- Truncated impulse response with windows
- Frequency sampling

- Optimum equiripple approximations
- Minimum mean-square error (frequency domain)

Note: The above designs are also typically constrained to have linear phase.

9.2 Continuous-Time Filter Design Overview

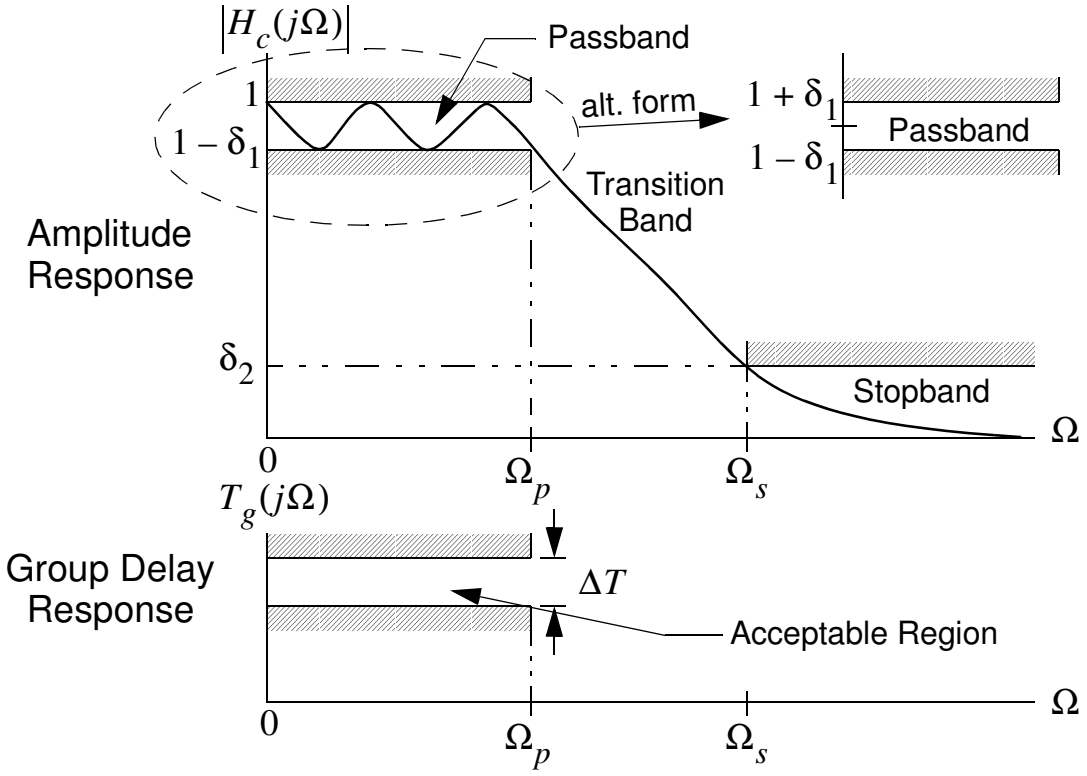
The general continuous-time system function of interest is of the form

$$H_c(s) = \frac{\sum_{m=0}^{M_c} c_m s^m}{\sum_{k=0}^{N} d_k s^k}$$

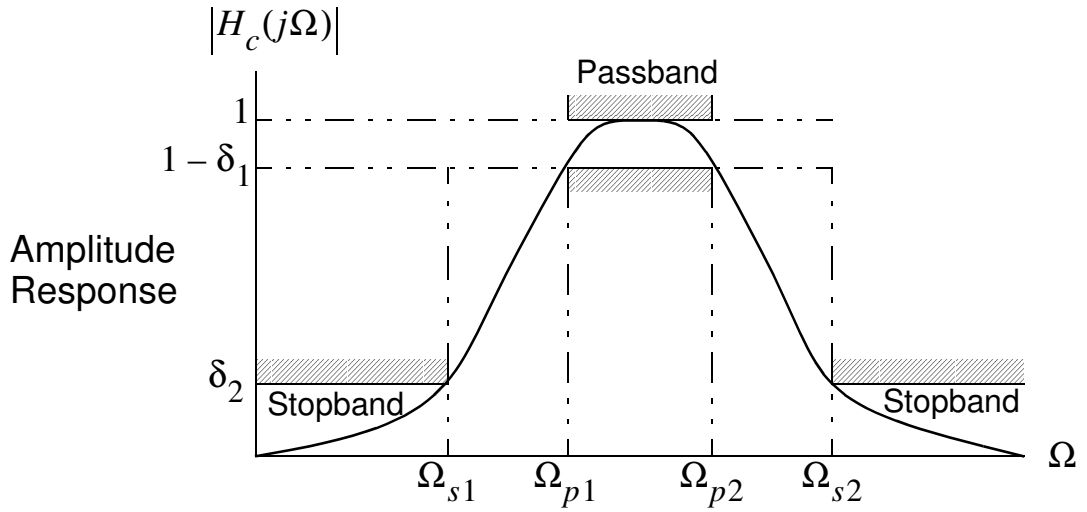
where $M_c \leq N$ insures finite gain as $\Omega \to \infty$. The classical analog filter designs which will be considered here are Butterworth (maximally flat), Chebyshev type I and II, and elliptical.

- Filter design usually begins with a specification of the desired frequency response
- The filter requirements may be stated in several ways:
  - Amplitude response $|H_c(j\,\Omega)|$
  - Phase response $\angle H_c(j\,\Omega) = \phi_c(j\,\Omega)$ or group delay $T_g(j\,\Omega) = -d\phi_c(j\,\Omega)/d\,\Omega$
  - A combination of amplitude and phase
- Group delay compensation may be provided by using allpass filters

Example 9.1: Lowpass Specifications



Example 9.2: Bandpass Amplitude Specifications



- In this chapter we will only consider filter requirements in terms of amplitude response

- Initially only lowpass characteristics will be considered with amplitude specifications:

  - Passband requirement —

$$1 \geq |H_c(j\Omega)| \geq 1 - \delta_1, \quad |\Omega| \leq \Omega_p$$

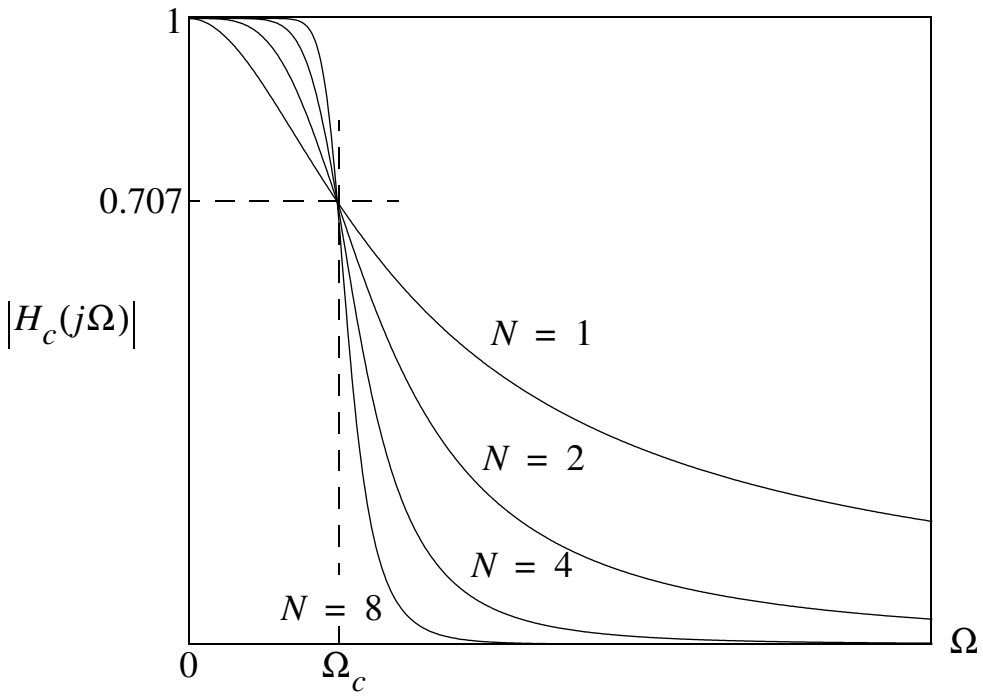
  - Stopband requirement —

$$|H_c(j\Omega)| \leq \delta_2, \quad |\Omega| \geq \Omega_s$$

- Later transformations to highpass, bandpass, and bandstop filters will be introduced

9.3 Butterworth Design

- Designed to maintain a constant amplitude response in the passband, and stopband



Butterworth Magnitude Response for order $N = 1, 2, 4$, and 8

- The Butterworth filter is optimum[1] in the sense that it provides the best Taylor series approximation to an *ideal* lowpass filter magnitude at both $\Omega = 0$ and ∞

- To achieve a Butterworth characteristic we require that the first $2N - 1$ derivatives of $|H_c(j\Omega)|^2 = 0$ at $\Omega = 0$ and ∞[2]

- Typically the cutoff frequency, Ω_c, is chosen to correspond to the 3dB point, that is

$$1 - \delta_1 = \frac{1}{\sqrt{2}}$$

[1] T.W. Parks and C.S. Burrus, *Digital Filter Design*, John Wiley & Sons, 1987.

[2] J.D. Rhodes *Theory of Electrical Filters*, John Wiley, 1976.

- Under the above assumption the magnitude-squared transfer function is

$$|H_c(j\Omega)|^2 = \frac{1}{1 + \left(\frac{\Omega}{\Omega_c}\right)^{2N}}$$

Butterworth Properties

- $|H_c(j\Omega)|_{\Omega=0} = 1$, for all N
- $|H_c(j\Omega)|_{\Omega=\Omega_c} = 1/\sqrt{2}$, for all N
- $|H_c(j\Omega)|$ is monotone decreasing for all Ω
- For $\Omega > \Omega_c$ $|H_c(j\Omega)|_{dB}$ has slope $-20N$ dB/decade or $-6N$ db/octave
- As $N \to \infty$ $|H_c(j\Omega)|$ approaches an ideal lowpass filter

System Function

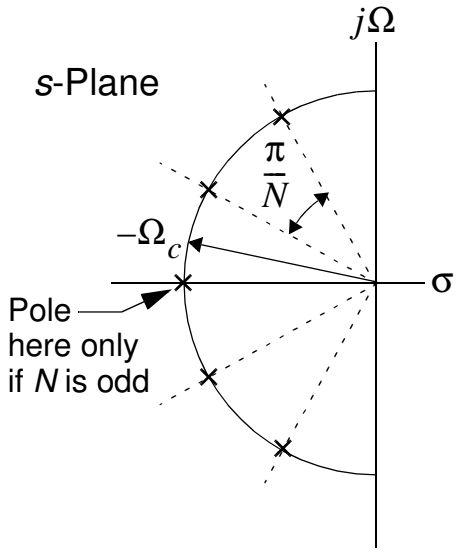
- From the form of $|H_c(j\Omega)|^2$ and the causality constraint, we can write

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

- $B_N(s)$ is an Nth order Butterworth polynomial with roots given by

$$s_k = \Omega_c e^{j\pi[0.5-(2k-1)/(2N)]}, \quad k = 1, 2, \ldots, N$$

- Note that $H_c(s)$ has N zeros at infinity and a pole on the negative real axis at $-\Omega_c$ if N is odd



Butterworth pole locations

Standard and Factored form Butterworth Polynomials for $\Omega_c = 1$ rad/sec:

Standard Form $B_N(s) = a_N s^N + a_{N-1}s^{N-1} + \cdots + a_1 s + a_0$							
a_6	a_5	a_4	a_3	a_2	a_1	a_0	N
					1	1	1
				1	$\sqrt{2}$	1	2
			1	2	2	1	3
		1	2.613	3.414	2.613	1	4
	1	3.236	5.236	5.236	3.236	1	5
1	3.864	7.464	9.141	7.464	3.864	1	6

Quadratic Factored Form $B_N(s)$	N
$s+1$	1
$s^2+\sqrt{2}s+1$	2
$(s^2+s+1)(s+1)$	3
$(s^2+0.76536s+1)(s^2+1.84776s+1)$	4
$(s+1)(s^2+0.6180s+1)(s^2+1.6180s+1)$	5
$(s^2+0.5176s+1)(s^2+\sqrt{2}s+1)(s^2+1.9318s+1)$	6

- To frequency scale the above polynomials to a new 3dB cutoff frequency simply let $s \rightarrow s/\Omega_c$

Example 9.3: Obtaining the Butterworth Polynomial

Design a Butterworth lowpass filter with 3dB frequency $\Omega_c = \Omega_p = 1$ rad/sec, $\Omega_s = 4$ rad/sec, and $20\log_{10}\delta_2 = -24$ dB.

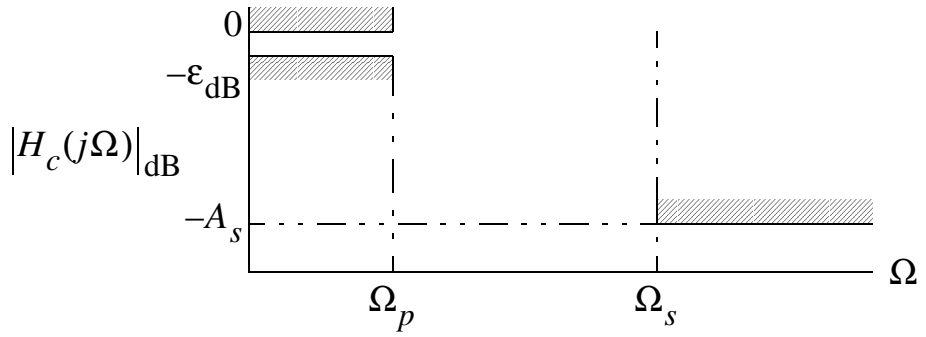
- Since Ω_s is two octaves above Ω_c we need a rolloff of 12 dB per octave $\rightarrow N \geq 2$ will work. Thus

$$H_c(s) = \frac{1}{s^2+\sqrt{2}s+1} \quad \text{from tables}$$

9.3.1 General Butterworth Design from Amplitude Specifications

Given:

1. $0 \geq 20\log_{10}|H_c(j\Omega)| \geq -\epsilon_{dB}, \quad \Omega \leq \Omega_p$
2. $20\log_{10}|H_c(j\Omega)| \leq -A_s, \quad \Omega \geq \Omega_s$



Amplitude response constraints

Find: N and Ω_c

- Solve the following equations:

$$|H_c(j\,\Omega_p)|_{dB} = 10\log\left[\frac{1}{1+(\Omega_p/\Omega_c)^{2N}}\right] = -\epsilon_{dB}$$

$$|H_c(j\,\Omega_s)|_{dB} = 10\log\left[\frac{1}{1+(\Omega_s/\Omega_c)^{2N}}\right] = -A_s$$

- Rewrite the above as

$$\left[10^{\epsilon_{dB}/10}-1\right] = \left(\frac{\Omega_p}{\Omega_c}\right)^{2N}$$

$$\left[10^{A_s/10}-1\right] = \left(\frac{\Omega_s}{\Omega_c}\right)^{2N}$$

- Dividing the first equation into the second gives

$$\left(\frac{\Omega_p}{\Omega_s}\right)^{2N} = \left[\frac{10^{\epsilon_{dB}/10}-1}{10^{A_s/10}-1}\right]$$

or

$$N = \left\lceil \frac{\log_{10}\left[\frac{10^{\epsilon_{dB}/10}-1}{10^{A_s/10}-1}\right]}{2\log_{10}(\Omega_p/\Omega_s)} \right\rceil$$

where $\lceil\ \rceil$ is the greatest integer operator

- Note: Since N is the greatest integer constraints (1) and (2) will not in general both be satisfied with equality at Ω_p and Ω_s respectively i.e.

$$|H_c(j\Omega_p)|_{dB} \geq -\epsilon_{dB} \text{ or } |H_c(j\Omega_s)|_{dB} \leq -A_s$$

- For equality at Ω_p (most popular) choose

$$\Omega_c = \frac{\Omega_p}{\left(10^{\epsilon_{dB}/10} - 1\right)^{1/(2N)}}$$

- For equality at Ω_s choose

$$\Omega_c = \frac{\Omega_s}{\left(10^{A_s/10} - 1\right)^{1/(2N)}}$$

- A third solution is to choose Ω_c somewhere in between the above solutions, then both requirements are exceeded

- Note: $\epsilon_{dB} = -20\log_{10}(1-\delta_1)$ and $A_s = -20\log_{10}\delta_2$

Example 9.4: A Butterworth Amplitude Response Design

Let $\Omega_p = 20$ rad/sec, $\epsilon_{dB} = 2$ dB
$\Omega_s = 30$ rad/sec, $A_s = 10$ dB

- Solving for N gives

$$N = \left\lceil \frac{\log_{10}\left[\frac{10^{2/10}-1}{10^{10/10}-1}\right]}{2\log_{10}(20/30)} \right\rceil$$
$$= \lceil 3.3709 \rceil = 4$$

- Matching at $\Omega_p \Longrightarrow$

$$\Omega_c = \frac{20}{\left(10^{2/10} - 1\right)^{1/8}} = 21.3868 \text{ rad/sec}$$

- The normalized $H_c(s)$ from the tables is

$$H_c(s) = \frac{1}{(s^2 + 0.76536s + 1)(s^2 + 1.84776s + 1)}$$

- Frequency scale $\Omega_c = 1 \rightarrow \Omega_c = 21.3868$ implies that we let $s \rightarrow s/21.3868$

- Finally, the frequency scaled system function is

$$H_c(s) = \frac{(21.3868)^4}{(s^2 + 16.37s + 457.4)(s^2 + 39.52s + 457.4)}$$

MATLAB Analysis

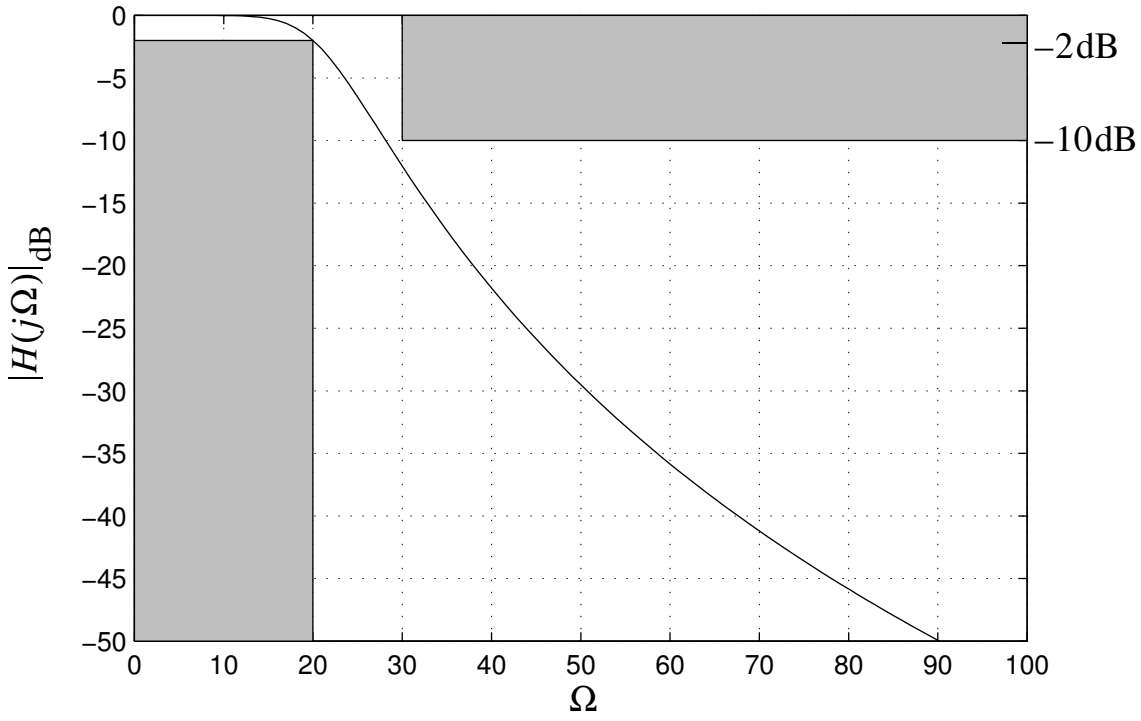
- We can use MATLAB to directly analyze the above Butterworth design

- MATLAB has the `freqs()` function for continuous-time systems, analogous to the `freqz()` function we have been using in the discrete-time domain

- Frequency scaling is handled in the calculation by letting $\Omega \rightarrow \Omega/21.3868$

```
>> w = 0:1:100;
>> H = freqs(1,conv([1 .76536 1],[1 1.8477 1]),w/21.3868);
>> plot(w,20*log10(abs(H)))
```

```
>> grid
>> axis([0 100 -50 0])
>> patch([0 20 20 0],[-50 -50 -2 -2],[.75 .75 .75])
>> patch([30 100 100 30],[-10 -10 0 0],[.75 .75 .75])
```



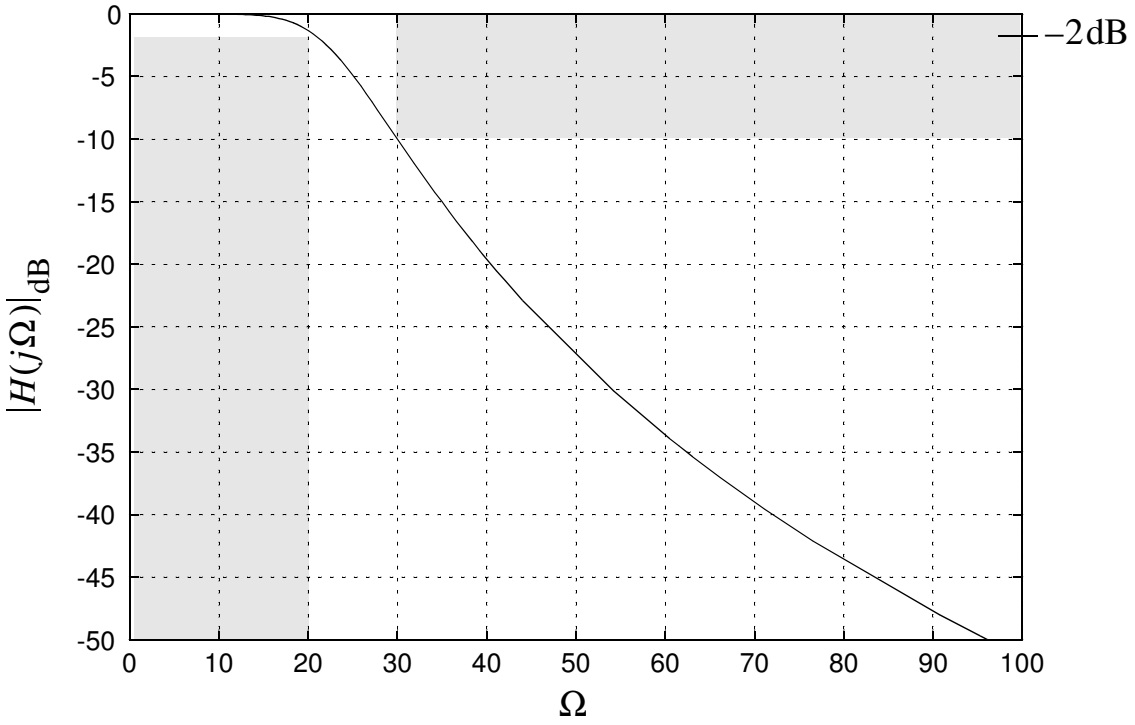
Butterworth magnitude response

MATLAB Filter Design

- An alternative approach is to design the filter completely using MATLAB filter design tools for the *s*-domain

```
>> [n,Wn] = buttord(20,30,2,10,'s')
n =
     4
Wn =
     2.2795e+001
```

```
>> [b,a] = butter(n,Wn,'s')
b =
    0             0             0
    0  2.7000e+005
a =
    1.0000e+000  5.9566e+001  1.7741e+003
    3.0952e+004  2.7000e+005
>> % Calculate the frequency response
>> [H,w] = freqs(b,a);
>> plot(w,20*log10(abs(H)));
```



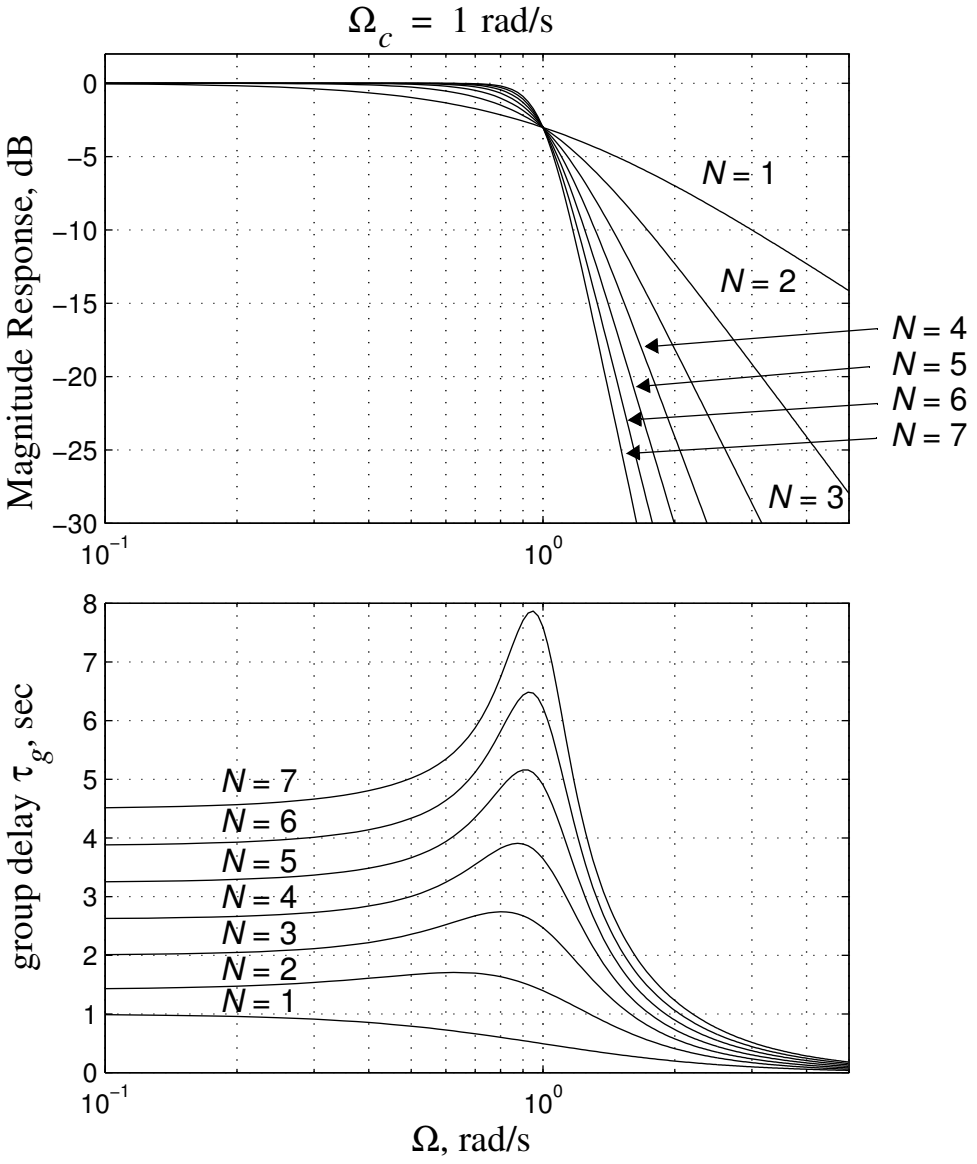
Butterworth magnitude response in the all MATLAB design

- Both the hand calculation and the MATLAB design use $N = 4$
- The normalized cutoff frequencies are slightly different, as MATLAB appears to choose Ω_c closer to the value we would obtain

by matching constraints at Ω_s, e.g.,

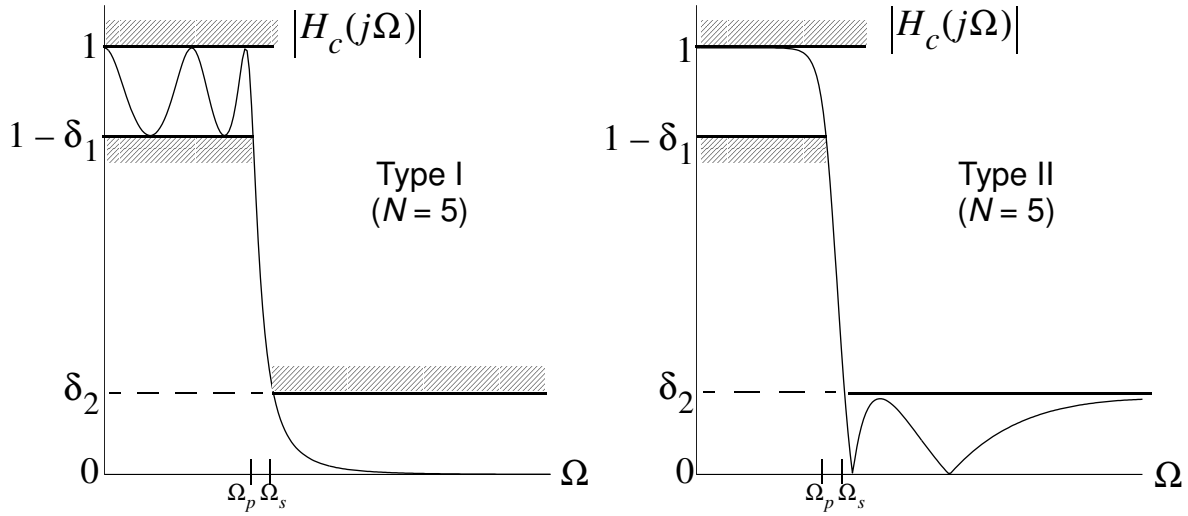
$$\Omega_c = \frac{30}{\left(10^{10/10} - 1\right)^{1/8}} = 22.795 \text{ rad/s}$$

9.3.2 Amplitude Response and Group Delay Summary



9.4 Chebyshev Design

A Chebyshev design achieves a more rapid rolloff rate near the cut-off frequency than the Butterworth by allowing ripple in the pass-band (type I) or stopband (type II). Monotonicity of the stopband or passband is still maintained respectively.



Differences between type I and II

9.4.1 Chebyshev Type I

- The magnitude response is given by

$$|H_c(j\,\Omega)|^2 = \frac{1}{1+\epsilon^2 T_N^2(\Omega/\,\Omega_c)}$$

where

$$T_N(x) = N\text{th order Chebyshev polynomial}$$

and ϵ specifies the passband ripple

- The Chebyshev polynomials are of the form

$$T_0(x) = 1, \;\; T_1(x) = x, \;\; T_2(x) = 2x^2 - 1 \ldots$$

  with recurrence formula

$$T_N(x) = 2xT_{N-1}(x) - T_{N-2}(x), \;\; N \geq 2$$

- An alternate form for $T_N(x)$, which will be useful in both analysis and design, is

$$T_N(x) = \begin{cases} \cos[N \cos^{-1} x], & |x| \leq 1 \\ \cosh[N \cosh^{-1} x], & |x| > 1 \end{cases}$$

Type I Properties

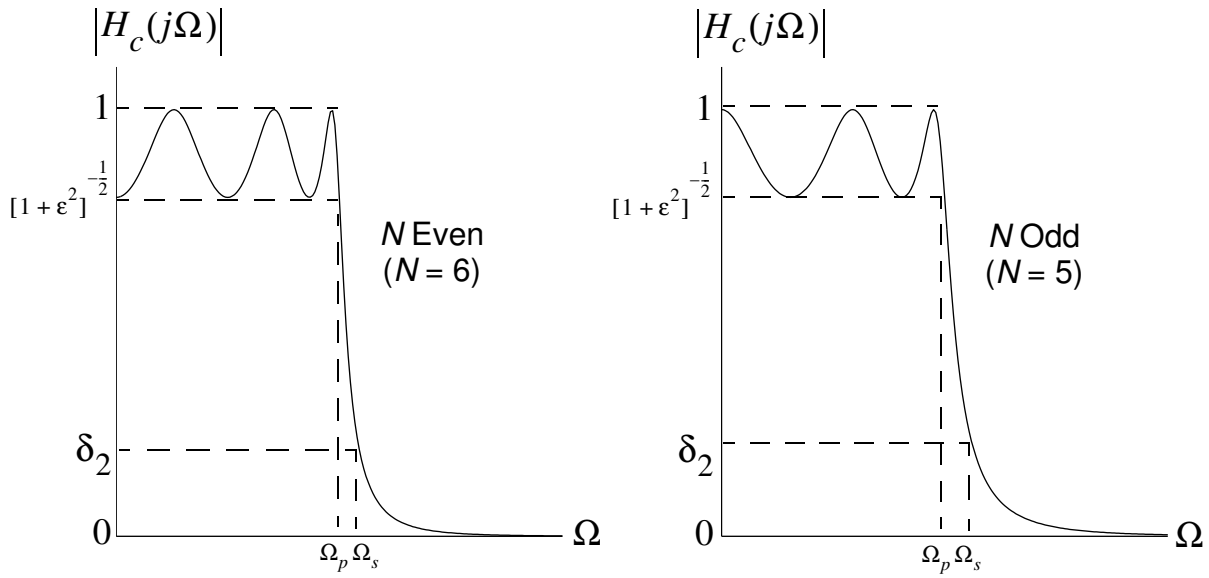
- $\frac{1}{\sqrt{1+\epsilon^2}} \leq |H_c(j\Omega)| \leq 1, \;\; 0 \leq \Omega \leq \Omega_c$

- The equiripple behavior gives a passband ripple in dB of

$$\epsilon_{dB} = 10 \log_{10}(1 + \epsilon^2) \text{ or } \epsilon = \sqrt{10^{\epsilon_{dB}/10} - 1}$$

- $|H_c(j\Omega)|$ is monotone decreasing for $\Omega \geq \Omega_p$

- The filter gain at $\Omega = 0$ is given by 1 if N is odd and $1/\sqrt{(1+\epsilon^2)}$ if N is even



The number of inflections indicates N

Type I System Function

- The system function is of the form

$$H_c(s) = \frac{K}{V_N(s)}$$

where

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

and

$$s_k = \Omega_c \left\{ -\sin\left[\frac{\pi}{2N}(2k-1)\right]\sinh\left[\frac{1}{N}\sinh^{-1}\frac{1}{\epsilon}\right] + j\cos\left[\frac{\pi}{2N}(2k-1)\right]\cosh\left[\frac{1}{N}\sinh^{-1}\frac{1}{\epsilon}\right]\right\}$$

for $k = 1, 2, \ldots, N$

- The poles are located on an ellipse with minor axis length $2a\Omega_p$ where

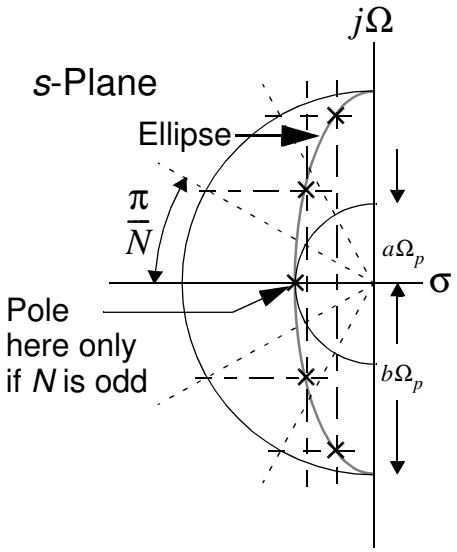
$$a = \frac{1}{2}(\alpha^{1/N} - \alpha^{-1/N})$$

and

$$\alpha = \epsilon^{-1} + \sqrt{1 + \epsilon^{-2}}$$

- The major axis length is $2b\Omega_p$ where

$$b = \frac{1}{2}(\alpha^{1/N} + \alpha^{-1/N})$$



Chebyshev type I pole locations

- The gain factor K is given by

$$K = \begin{cases} V_N(0), & N \text{ odd} \\ \frac{V_N(0)}{\sqrt{1+\epsilon^2}}, & N \text{ even} \end{cases}$$

which results from requiring that

$$H_c(0) = \begin{cases} 1, & N \text{ odd} \\ \frac{1}{\sqrt{1+\epsilon^2}}, & N \text{ even} \end{cases}$$

Normalized Chebyshev Polynomials $V_N(s)$ for $\epsilon_{dB} = 0.5$, 1, and 2 dB:

$V_N(s) = s^N + a_{N-1}s^{N-1} + \cdots + a_1 s + a_0$

N	a_0	a_1	a_2	a_3	a_4	a_5
			$\epsilon_{dB} = 0.5, \epsilon = 0.34931$			
1	2.86277					
2	1.51620	1.42562				
3	0.71570	1.53489	1.25291			
4	0.37905	1.02545	1.71687	1.19739		
5	0.17892	0.75252	1.30957	1.93737	1.17249	
6	0.09476	0.43237	1.17186	1.58976	2.17184	1.15918

$V_N(s) = s^N + a_{N-1}s^{N-1} + \cdots + a_1 s + a_0$

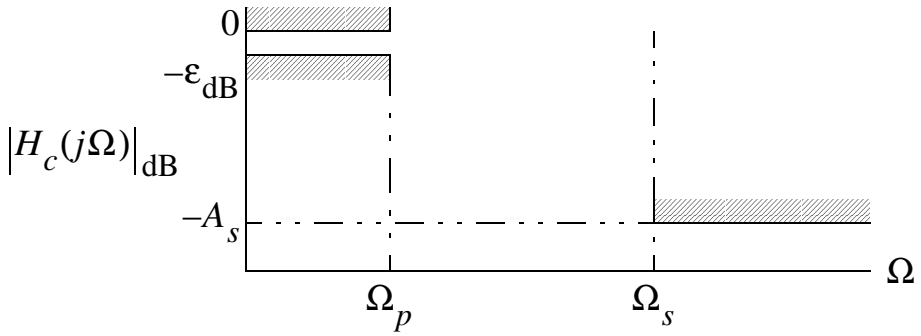
N	a_0	a_1	a_2	a_3	a_4	a_5
			$\epsilon_{dB} = 1, \epsilon = 0.50885$			
1	1.96523					
2	1.10251	1.09773				
3	0.49131	1.23841	.98834			
4	0.27563	0.74262	1.45392	0.95281		
5	0.12283	0.58053	0.97430	1.68881	0.93682	
6	0.06891	0.30708	0.93935	1.20214	1.93082	.92825

$V_N(s) = s^N + a_{N-1}s^{N-1} + \cdots + a_1 s + a_0$

N	a_0	a_1	a_2	a_3	a_4	a_5
			$\epsilon_{dB} = 2\,\epsilon = 0.76478$			
1	1.30756					
2	0.63677	0.80382				
3	0.32689	1.02219	0.73782			
4	0.20576	0.51680	1.25648	0.71621		
5	0.08172	0.45935	0.69348	1.49954	0.70646	
6	0.05144	0.21027	0.77146	0.86701	1.74586	0.70123

9.4.2 General Chebyshev Type I Design from Amplitude Specifications

Given: ϵ_{dB}, A_s, Ω_p, and Ω_s



Amplitude response constraints

Find: N

- To achieve the desired stopband attenuation, A_s, at Ω_s we set

$$|H_c(j\,\Omega_s)|^{-2} = 10^{A_s/10} = 1 + \epsilon^2 \cosh^2\left[N \cosh^{-1}(\Omega_s/\,\Omega_p)\right]$$

- Solve for N

$$N \cosh^{-1} \Omega_s/\,\Omega_p = \cosh^{-1}\left(\sqrt{\frac{10^{A_s/10}-1}{10^{\epsilon_{dB}/10}-1}}\right)$$

or

$$N = \left\lceil \frac{\cosh^{-1}\left(\sqrt{\frac{10^{A_s/10}-1}{10^{\epsilon_{dB}/10}-1}}\right)}{\cosh^{-1}\left(\Omega_s/\,\Omega_p\right)} \right\rceil$$

Example 9.5: A Chebyshev Type I Design

Design a Chebyshev type I lowpass filter to satisfy the following amplitude specifications: $\epsilon_{dB} = 2$dB, $A_s = 20$dB, $\Omega_p = 40$ rad/s, and $\Omega_s = 52$ rad/s.

- Using the design formula for N

$$N = \left\lceil \frac{\cosh^{-1}\left(\sqrt{\frac{10^{20/10}-1}{10^{2/10}-1}}\right)}{\cosh^{-1}(52/40)} \right\rceil = \lceil 4.3 \rceil = 5$$

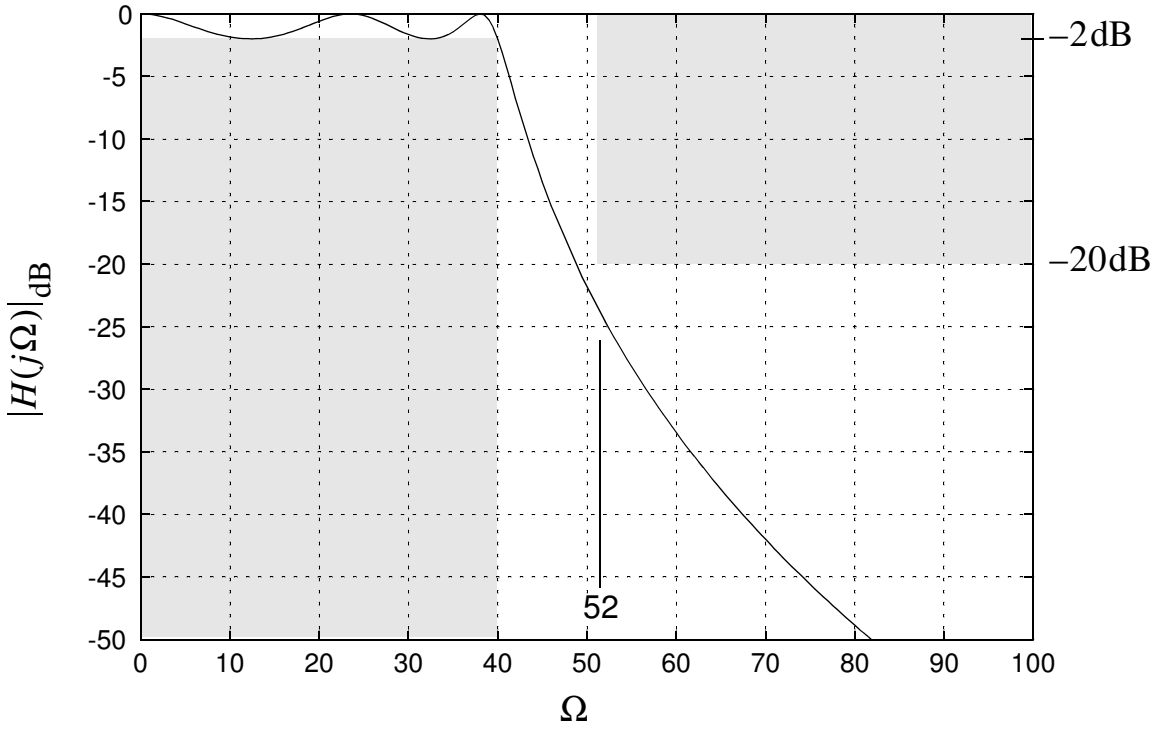
- From the 2dB ripple table (9-20),

$$H_c(s) = \left. \frac{0.0817}{s^5 + .706s^4 + 1.50s^3 + .694s^2 + .459s + .0817} \right|_{s \to s/40}$$

MATLAB Analysis

- We can use MATLAB to directly analyze the above Chebyshev type I design
- Frequency scaling is handled in the calculation by letting $\Omega \to \Omega/40$

```
>> w = 0:1:100;
>> w = 0:100/200:100;
>> H = freqs(0.08172,...
     [1 .70646 1.4995 .6935 .4593 .08172],w/40);
>> plot(w,20*log10(abs(H)))
>> axis([0 100 -50 0])
>> grid
```



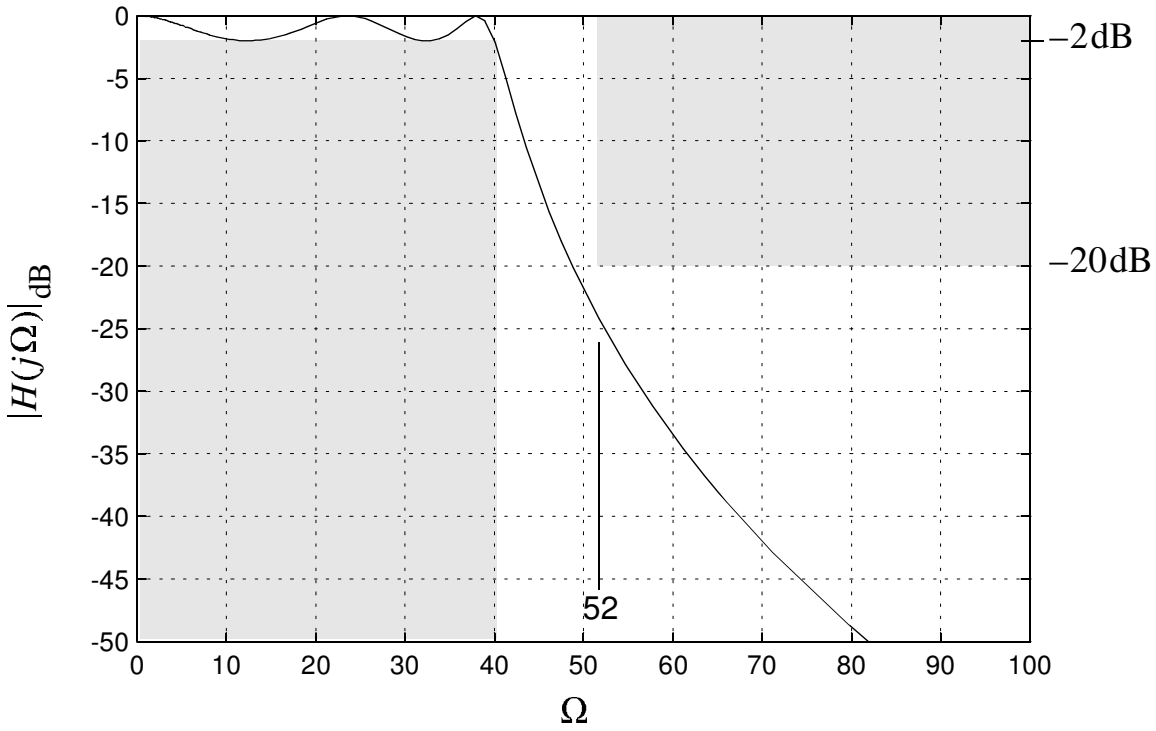
Chebyshev type I magnitude response

MATLAB Filter Design

- An alternative approach is to design the filter completely using MATLAB filter design tools for the *s*-domain

```
>> [n,Wn] = cheb1ord(40,52,2,20,'s')
n =
     5
Wn =
    40
>> [b,a] = cheby1(n,2,Wn,'s')
b =
         0         0         0         0
         0  8.3684e+006
a =
    1.0000e+000  2.8258e+001  2.3993e+003
    4.4383e+004  1.1759e+006  8.3684e+006
>> % Calculate the frequency response
```

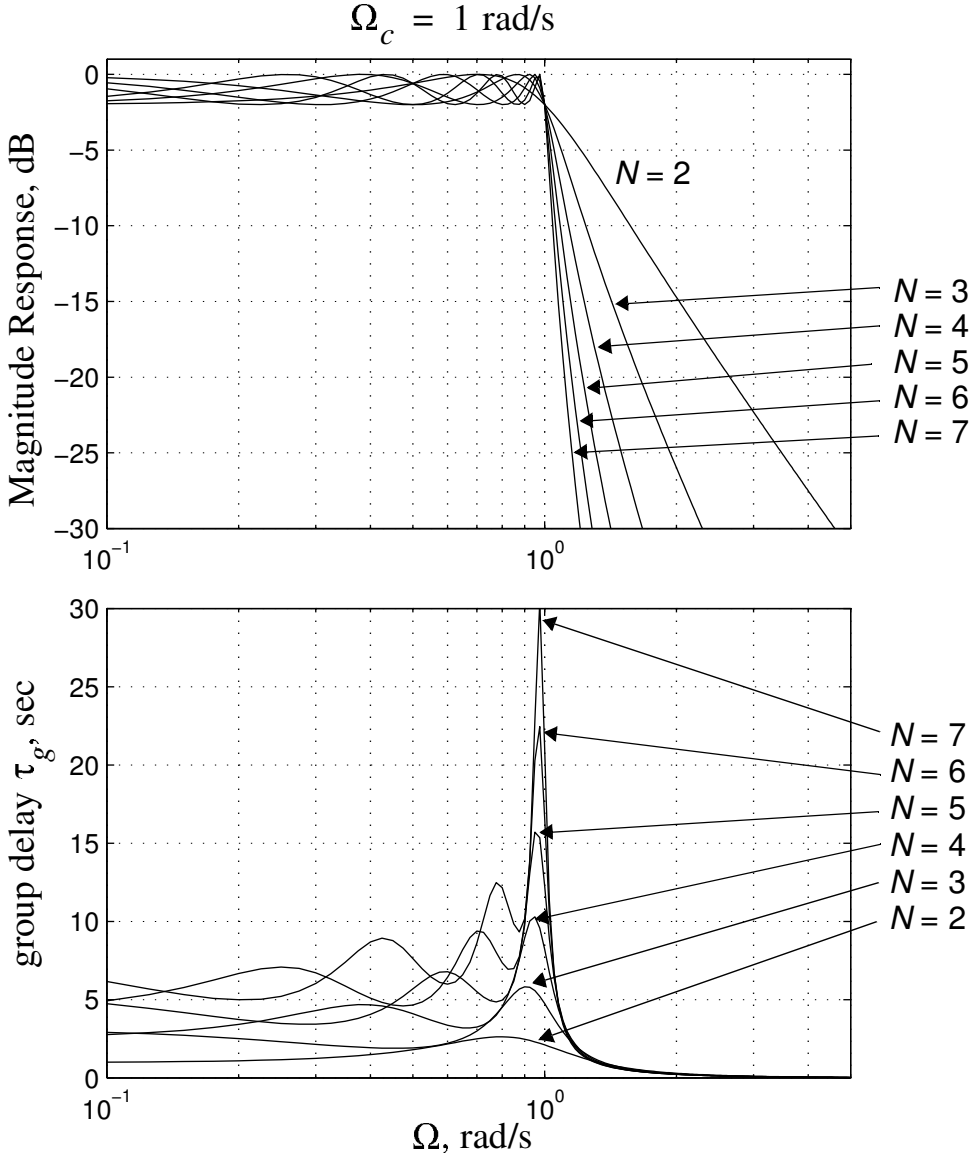
```
>> [H,w] = freqs(b,a);
>> plot(w,20*log10(abs(H)));
```



Chebyshev type I magnitude response in the all MATLAB design

- Both the hand calculation and the MATLAB design result in $N = 4$
- The coefficients are only slightly differnt, e.g., the numerator coefficient in the hand calculation after scaling is $0.0817 \times 40^5 = 8.36608 \times 10^6$ compared with 8.3684×10^6 from the MATLAB design
- The plotted results look virtually identical

9.4.3 Amplitude Response and Group Delay Summary



9.4.4 Chebyshev Type II

- The magnitude response is given by

$$|H_c(j\Omega)|^2 = \frac{1}{1+\epsilon^2\left[\frac{T_N^2(\Omega_s/\Omega_p)}{T_N^2(\Omega_s/\Omega)}\right]}$$

- Note that the passband ripple is still ϵ_{dB} since $|H_c(j\Omega_p)|^2 = 1/(1+\epsilon^2)$
- $|H_c(j\Omega)|$ is monotone decreasing for $\Omega \leq \Omega_p$

Type II System Function

- The system function is of the form

$$H_c(s) = \frac{\prod_{m=1}^{N}(s - z_m)}{\prod_{k=1}^{N}(s - p_k)}$$

where

$$z_m = \frac{j\Omega_s}{\cos\left[\frac{\pi}{2N}(2m-1)\right]}, \quad m = 1, 2, \ldots, N$$

$$p_k = \alpha_k + j\beta_k, \quad k = 1, 2, \ldots, N$$

and

$$\alpha_k = \frac{\Omega_p \Omega_s \sigma_k}{\sigma_k^2 + \Omega_k^2}, \quad \beta_k = \frac{-\Omega_p \Omega_s \Omega_k}{\sigma_k^2 + \Omega_k^2}$$

$$\sigma_k = -\sin\left[\frac{\pi}{2N}(2k-1)\right]\sinh\left[\frac{1}{N}\sinh^{-1}\frac{1}{\epsilon}\right]$$

$$\Omega_k = \cos\left[\frac{\pi}{2N}(2k-1)\right]\cosh\left[\frac{1}{N}\sinh^{-1}\frac{1}{\epsilon}\right]$$

Example 9.6: A Chebyshev Type II Design

Design a Chebyshev type II lowpass filter to satisfy the following amplitude specifications: $\epsilon_{dB} = 2$dB, $A_s = 40$dB, $\Omega_p = 100$ rad/s, and $\Omega_s = 200$ rad/s.

- To find the filter order N set

$$|H_c(j\Omega_s)|^{-2} = 10^{A_s/10} = 1 + \epsilon^2 \frac{T_N^2(\Omega_s/\Omega_p)}{T_N^2(\Omega_s/\Omega_s)}$$

- Since $T_N(1) = 1$ and $\Omega_s/\Omega_p > 1$,

$$10^{A_s/10} = 1 + \epsilon^2 T_N^2(\Omega_s/\Omega_p)$$

or

$$N = \left\lceil \frac{\cosh^{-1}\left(\sqrt{\frac{10^{A_s/10}-1}{10^{\epsilon_{dB}/10}-1}}\right)}{\cosh^{-1}\left(\Omega_s/\Omega_p\right)} \right\rceil$$

which is the same as the type I formula

- For the problem of interest

$$N = \left\lceil \frac{\cosh^{-1}\left(\sqrt{\frac{10^{40/10}-1}{10^{2/10}-1}}\right)}{\cosh^{-1}\left(200/100\right)} \right\rceil$$
$$= \lceil 4.227 \rceil = 5$$

MATLAB Analysis

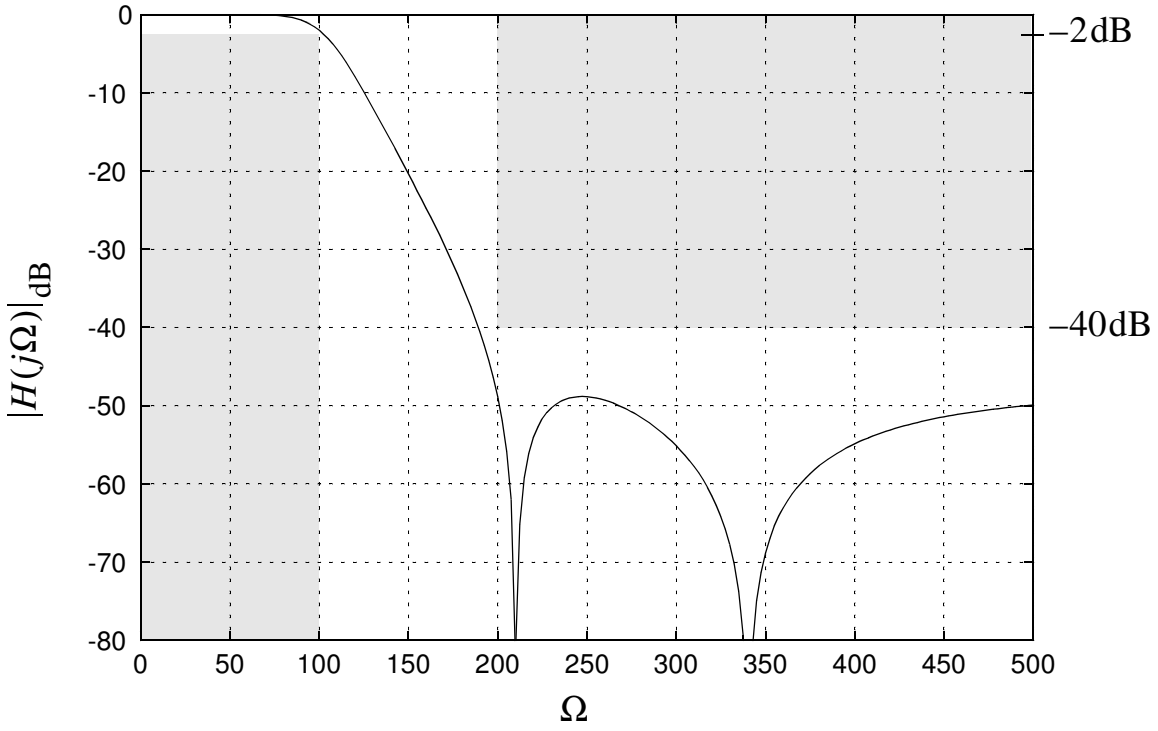
- We can use MATLAB to directly analyze the above Chebyshev type II design

- Rather than actually computing the polynomial coefficients, numerator and denominator, we will calculate just the frequency response magnitude

- To do that we will write a simple `m-file` to evaluate $T_N(\Omega)$

```
function Tn = chebpoly(N,x)
%        Tn = chebpoly(N,x)
% Chebyshev polynomial T_N(x) for use in filter analysis
% N = order
% x = vector of input values

Tn = zeros(size(x));
s0 = find(abs(x) < 1);
s1 = find(abs(x) >= 1);
Tn(s0) = cos(N*acos(x(s0)));
Tn(s1) = cosh(N*acosh(x(s1)));
```

```
>> w = 0:500/200:500;
>> H = -10*log10(1 + (10^(2/10)-1)...
     *(chebpoly(5,2)./chebpoly(5,200./w)).^2);
Warning: Divide by zero.
>> plot(w,H)
>> grid
>> axis([0 500 -80 0])
```

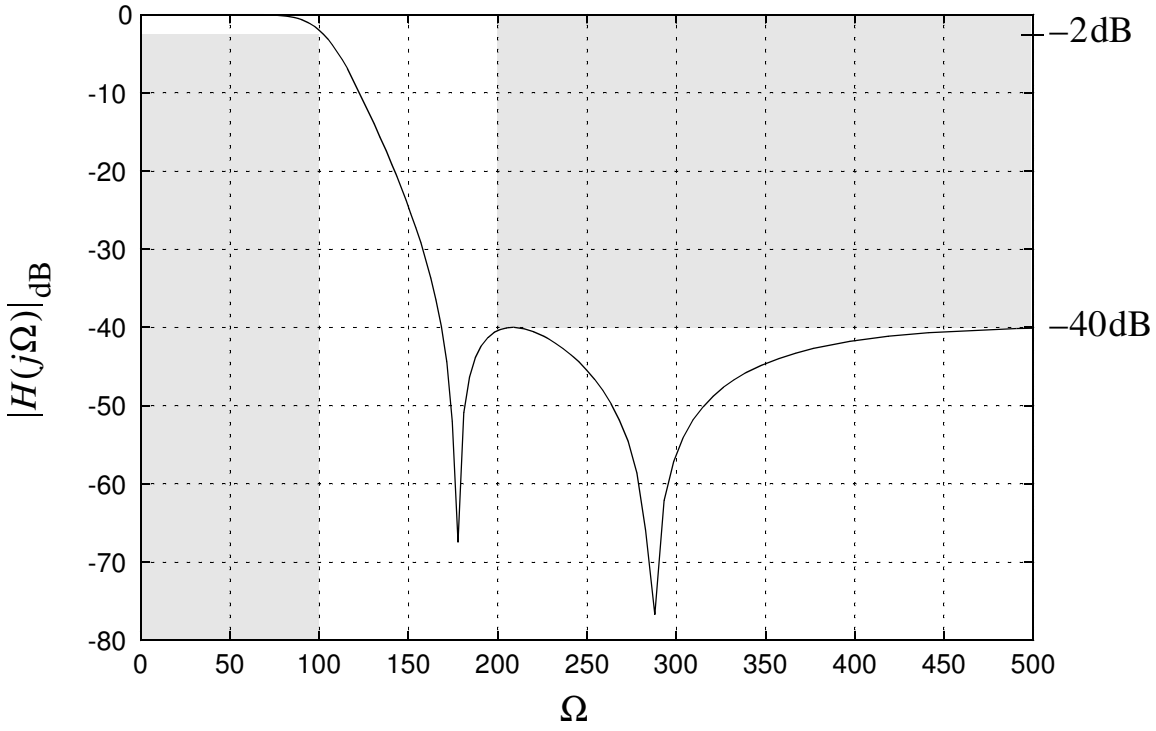


Chebyshev type II magnitude response

MATLAB Filter Design

- An alternative approach is to design the filter completely using MATLAB filter design tools for the *s*-domain

```
>> [n,Wn] = cheb2ord(100,200,2,40,'s')
n =
     5
Wn =
    1.6864e+002
>> [b,a] = cheby2(n,40,Wn,'s')
b =
         0   8.4325e+000  -8.9874e-013
    9.5927e+005  -2.5210e-008   2.1825e+010
a =
    1.0000e+000   3.6244e+002   6.5647e+004
    7.4342e+006   5.3163e+008   2.1825e+010
>> % Calculate the frequency response
>> [H,w] = freqs(b,a);
>> plot(w,20*log10(abs(H)));
```



Chebyshev type II magnitude response in the all MATLAB design

- Both the hand calculation and the MATLAB design result in $N = 5$
- The hand calculation and the use of `cheb2ord()` yield different results in other respects however
- In the MATLAB design the minimum stopband attenuation is exactly 40dB rather than the 48dB value of the hand calculation; why the difference?
- In the hand calculation the minimum attenuation increases from 40dB to 48dB when the filter order is rounded up from 4.227 to 5
- In the MATLAB calculation ϵ is reduced to allow the minimum attenuation to drop to 40dB
- The filter cutoff frequency is then be reduced (to 168.6 rad/s) so that the ϵ_{dB} point is still at 200 rad/s

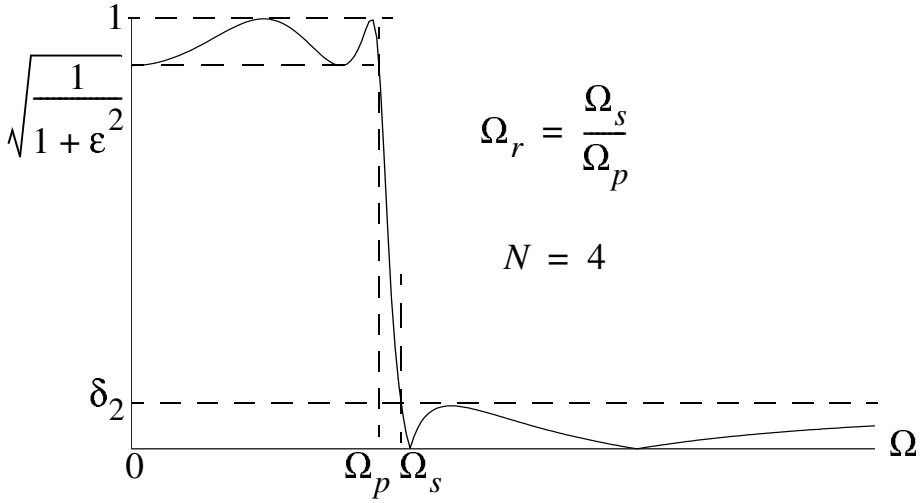
9.5 Elliptic Design

Allows both passband and stopband ripple to obtain a narrow transition band. The elliptic (Cauer) filter is optimum in the sense that no other filter of the same order can provide a narrower transition band.

- The squared magnitude response is given by

$$|H_c(j\Omega)|^2 = \frac{1}{1+\epsilon^2 U_N^2(\Omega/\Omega_p)}$$

where $U_N(\Omega)$ is a Jacobian elliptic function



- Define the transition region ratio as

$$\Omega_r = \frac{\Omega_s}{\Omega_p}$$

- The normalized lowpass system function can be written in factored form as

$$H_c(s) = \begin{cases} \frac{H_0}{s+s_0}\prod_{i=1}^{(N-1)/2}\frac{s^2+A_{0i}}{s^2+B_{1i}s+B_{0i}}, & N \text{ odd} \\ \prod_{i=1}^{(N-1)/2}\frac{s^2+A_{0i}}{s^2+B_{1i}s+B_{0i}}, & N \text{ even} \end{cases}$$

Note: $H_c(s)$ contains conjugate pairs of zeros on the $j\Omega$ axis which give the stopband nulls

- The filter coefficients H_0, and s_0 if N is odd, and A_{0i}, B_{1i}, and B_{0i} are determined from the filter specifications ϵ_{dB}, A_s, and Ω_r

- A set of formulas useful for direct calculation of the elliptic lowpass pole and zero locations can be found in Antoniou[3] and MATLAB can be used directly as we will see in the examples

- The following pages contain several of the design data tables which give the achieved value for Ω_r as a function of N for fixed ϵ_{dB} and A_s

- These tables assume quadratic filter sections described above have been written as a ratio of N-order polynomials, where the numerator contains only even-order coefficients due to the zeros being on the $j\omega$-axis, e.g.,

$$H(s) = \begin{cases} \frac{b_N s^N + b_{N-2} s^{N-2} + \cdots + b_2 s^2 + b_0}{s^N + a_{N-1} s^{N-1} + \cdots + a_1 s + a_0}, & N \text{ even} \\ \frac{b_{N-1} s^{N-1} + b_{N-3} s^{N-3} + \cdots + b_2 s^2 + b_0}{s^N + a_{N-1} s^{N-1} + \cdots + a_1 s + a_0}, & N \text{ odd} \end{cases}$$

  also the filters have a normalized cuttoff frequency of $\Omega_p = 1$ rad/s

[3] A. Antoniou, *Digital Filters: Analysis and Design*, McGraw Hill, New York, 1979.

Passband Ripple $\epsilon_{\text{dB}} = 1$, Stopband Attenuation $A_s = 20$ dB

N	$H(s) = \dfrac{b_N s^N + b_{N-1} s^{N-1} + \ldots + b_1 s + b_0}{a_N s^N + a_{N-1} s^{N-1} + \ldots + a_1 s + a_0}$						$\dfrac{\Omega_s}{\Omega_p}$
2	$b_{N,\ldots,0}$						2.3240
				0.10001	0	1.02771	
	$a_{N,\ldots,0}$						
				1.00000	1.02957	1.15311	
3	$b_{N,\ldots,0}$						1.3078
			0	0.32057	0	0.66468	
	$a_{N,\ldots,0}$						
			1.00000	0.96658	1.24066	0.66468	
4	$b_{N,\ldots,0}$						1.0902
		0.09998	0	0.54212	0	0.52554	
	$a_{N,\ldots,0}$						
		1.00000	0.90376	1.67646	0.86878	0.58967	
5	$b_{N,\ldots,0}$					0	1.0282
		0.28302	0	0.74704	0	0.47648	
	$a_{N,\ldots,0}$					1.00000	
		0.92728	2.04748	1.40558	1.03362	0.47648	
6	$b_{N,\ldots,0}$				0.09996	0	1.0090
		0.60190	0	0.95505	0	0.45419	
	$a_{N,\ldots,0}$				1.00000	0.89292	
		2.57830	1.68592	2.08694	0.79228	0.50961	
7	$b_{N,\ldots,0}$			0	0.27936	0	1.0029
		0.99934	0	1.16875	0	0.44882	
	$a_{N,\ldots,0}$			1.00000	0.92339	3.00646	
		2.28719	2.99590	1.81256	0.98946	0.44882	
8	$b_{N,\ldots,0}$		0.09996	0	0.69714	0	1.0009
		1.53744	0	1.38404	0	0.44378	
	$a_{N,\ldots,0}$		1.00000	0.89183	3.56074	2.56222	
		4.61773	2.44887	2.55492	0.77848	0.49793	
9	$b_{N,\ldots,0}$	0.00000	0.27912	0	1.27505	0	1.0003
		2.15772	0	1.60682	0	0.44503	
	$a_{N,\ldots,0}$	1.00000	0.92301	3.99980	3.20335	5.98206	
		4.08266	3.96480	2.24735	0.98254	0.44503	

Passband Ripple $\epsilon_{\text{dB}} = 1$, Stopband Attenuation $A_s = 30$ dB

N	$H(s) = \frac{b_N s^N + b_{N-1} s^{N-1} + \ldots + b_1 s + b_0}{a_N s^N + a_{N-1} s^{N-1} + \ldots + a_1 s + a_0}$						$\frac{\Omega_s}{\Omega_p}$
2	$b_{N,\ldots,0}$						4.0036
				0.03164	0	0.99795	
	$a_{N,\ldots,0}$						
				1.00000	1.07759	1.11971	
3	$b_{N,\ldots,0}$						1.7324
			0	0.14902	0	0.56866	
	$a_{N,\ldots,0}$					0	
			1.00000	0.97015	1.24597	0.56866	
4	$b_{N,\ldots,0}$						1.2504
		0.03163	0	0.28089	0	0.38968	
	$a_{N,\ldots,0}$						
		1.00000	0.92931	1.56646	0.83918	0.43723	
5	$b_{N,\ldots,0}$					0	1.0955
		0.11635	0	0.39623	0	0.31327	
	$a_{N,\ldots,0}$					1.00000	
		0.92004	1.93449	1.22581	0.89778	0.31327	
6	$b_{N,\ldots,0}$				0.03163	0	1.0380
		0.26420	0	0.50595	0	0.27874	
	$a_{N,\ldots,0}$				1.00000	0.91055	
		2.37658	1.58926	1.68344	0.67748	0.31275	
7	$b_{N,\ldots,0}$			0	0.11191	0	1.0154
		0.46635	0	0.61684	0	0.26297	
	$a_{N,\ldots,0}$			1.00000	0.91236	2.82822	
		2.03866	2.61733	1.38890	0.78964	0.26297	
8	$b_{N,\ldots,0}$		0.03158	0	0.28755	0	1.0063
		0.73224	0	0.73158	0	0.25535	
	$a_{N,\ldots,0}$		1.00000	0.90756	3.31899	2.44379	
		3.91523	2.16991	1.88278	0.63372	0.28651	
9	$b_{N,\ldots,0}$	0	0.11124	0	0.56917	0	1.1486
		1.05647	0	0.85077	0	0.25223	
	$a_{N,\ldots,0}$	1.00000	0.91113	3.79911	2.92176	5.36035	
		3.36246	3.32451	1.60406	0.76327	0.25223	

Passband Ripple $\epsilon_{\text{dB}} = 1$, Stopband Attenuation $A_s = 40$ dB

N	$H(s) = \dfrac{b_N s^N + b_{N-1} s^{N-1} + \ldots + b_1 s + b_0}{a_N s^N + a_{N-1} s^{N-1} + \ldots + a_1 s + a_0}$						$\dfrac{\Omega_s}{\Omega_p}$
2	$b_{N,\ldots,0}$						7.0434
				0.01001	0	0.98757	
	$a_{N,\ldots,0}$						
				1.00000	1.09150	1.10807	
3	$b_{N,\ldots,0}$						2.4162
			0	0.06920	0	0.52652	
	$a_{N,\ldots,0}$						
			1.00000	0.97825	1.24339	0.52652	
4	$b_{N,\ldots,0}$						1.5154
		0.01000	0	0.15020	0	0.32196	
	$a_{N,\ldots,0}$						
		1.00000	0.93913	1.51372	0.80369	0.36125	
5	$b_{N,\ldots,0}$					0	1.2187
		0.04698	0	0.22007	0	0.22985	
	$a_{N,\ldots,0}$					1.00000	
		0.92339	1.84712	1.12923	0.78813	0.22985	
6	$b_{N,\ldots,0}$				0.01000	0	1.0989
		0.11725	0	0.27999	0	0.18600	
	$a_{N,\ldots,0}$				1.00000	0.91544	
		2.23780	1.47986	1.43164	0.56516	0.20869	
7	$b_{N,\ldots,0}$			0	0.04340	0	1.0460
		0.21884	0	0.33726	0	0.16406	
	$a_{N,\ldots,0}$			1.00000	0.91246	2.66449	
		1.87478	2.27753	1.12567	0.61545	0.16406	
8	$b_{N,\ldots,0}$		0.01000	0	0.11911	0	1.0217
		0.35275	0	0.39612	0	0.15275	
	$a_{N,\ldots,0}$		1.00000	0.91034	3.12221	2.28339	
		3.39450	1.85875	1.44392	0.48587	0.17139	
9	$b_{N,\ldots,0}$	0	0.04267	0	0.25083	0	1.0103
		0.51899	0	0.45762	0	0.14682	
	$a_{N,\ldots,0}$	1.00000	0.91002	3.59456	2.71901	4.74646	
		2.85738	2.71414	1.19524	0.56223	0.14682	

Passband Ripple $\epsilon_{\text{dB}} = 1$, Stopband Attenuation $A_s = 50$ dB

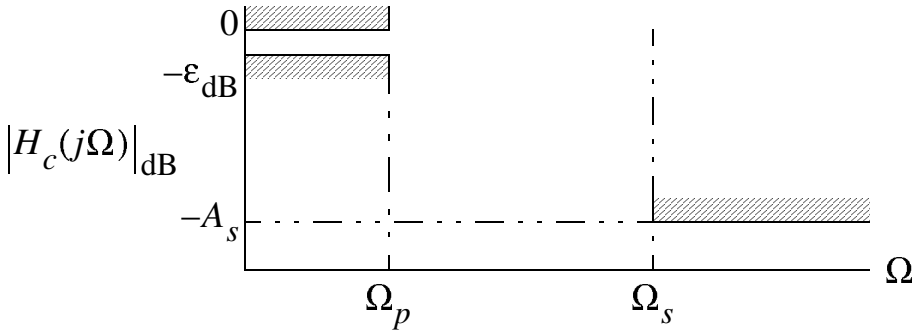
N	$H(s) = \frac{b_N s^N + b_{N-1} s^{N-1} + \ldots + b_1 s + b_0}{a_N s^N + a_{N-1} s^{N-1} + \ldots + a_1 s + a_0}$						$\frac{\Omega_s}{\Omega_p}$
2	$b_{N,\ldots,0}$						12.4939
				0.00315	0	0.98415	
	$a_{N,\ldots,0}$						
				1.00000	1.09575	1.10424	
3	$b_{N,\ldots,0}$						3.4607
			0	0.03212	0	0.50751	
	$a_{N,\ldots,0}$						
			1.00000	0.98333	1.24105	0.50751	
4	$b_{N,\ldots,0}$						1.9083
		0.00316	0	0.08198	0	0.28704	
	$a_{N,\ldots,0}$						
		1.00000	0.94472	1.48661	0.77888	0.32206	
5	$b_{N,\ldots,0}$					0	1.4072
		0.01881	0	0.12710	0	0.18519	
	$a_{N,\ldots,0}$					1.00000	
		0.92736	1.78896	1.07156	0.71304	0.18519	
6	$b_{N,\ldots,0}$				0.00316	0	1.1989
		0.05259	0	0.16247	0	0.13585	
	$a_{N,\ldots,0}$				1.00000	0.91857	
		2.14114	1.39498	1.26852	0.48165	0.15243	
7	$b_{N,\ldots,0}$			0	0.01652	0	1.1013
		0.10409	0	0.19279	0	0.11047	
	$a_{N,\ldots,0}$			1.00000	0.91412	2.53421	
		1.75435	2.01674	0.95022	0.48840	0.11047	
8	$b_{N,\ldots,0}$		0.00316	0	0.04927	0	1.0527
		0.17251	0	0.22225		0.09677	
	$a_{N,\ldots,0}$		1.00000	0.91171	2.96074	2.14052	
		2.99780	1.59958	1.14659	0.37114	0.10858	
9	$b_{N,\ldots,0}$	0	0.01595	0	0.11006	0	1.0277
		0.25802	0	0.25292	0	0.08913	
	$a_{N,\ldots,0}$	1.00000	0.91050	3.41139	2.55050	4.22165	
		2.46916	2.22050	0.91837	0.41013	0.08913	

Passband Ripple $\epsilon_{\text{dB}} = 1$, Stopband Attenuation $A_s = 60$ dB

N	$H(s) = \dfrac{b_N s^N + b_{N-1} s^{N-1} + \ldots + b_1 s + b_0}{a_N s^N + a_{N-1} s^{N-1} + \ldots + a_1 s + a_0}$						$\dfrac{\Omega_s}{\Omega_p}$
2	$b_{N,\ldots,0}$						22.1625
				0.00100	0	0.98313	
	$a_{N,\ldots,0}$						
				1.00000	1.09713	1.10309	
3	$b_{N,\ldots,0}$						5.0218
			0	0.01491	0	0.49879	
	$a_{N,\ldots,0}$						
			1.00000	0.98593	1.23970	0.49879	
4	$b_{N,\ldots,0}$						2.4608
		0.00100	0	0.04533	0	0.26844	
	$a_{N,\ldots,0}$						
		1.00000	0.94808	1.47203	0.76357	0.30119	
5	$b_{N,\ldots,0}$					0	1.6715
		0.00751	0	0.07561	0	0.16017	
	$a_{N,\ldots,0}$					1.00000	
		0.93047	1.75190	1.03574	0.66440	0.16017	
6	$b_{N,\ldots,0}$				0.00100	0	1.3435
		0.02380	0	0.09829	0	0.10751	
	$a_{N,\ldots,0}$				1.00000	0.92113	
		2.07439	1.33460	1.16076	0.42428	0.12063	
7	$b_{N,\ldots,0}$				0.00623	0	1.1855
		0.05032	0	0.11537	0	0.08040	
	$a_{N,\ldots,0}$			1.00000	0.91594	2.43609	
		1.66522	1.82732	0.83023	0.40146	0.08040	
8	$b_{N,\ldots,0}$		0.00100	0	0.02043	0	1.1031
		0.08620	0	0.13033	0	0.06565	
	$a_{N,\ldots,0}$		1.00000	0.91304	2.83243	2.02530	
		2.69950	1.40225	0.94302	0.29045	0.07366	
9	$b_{N,\ldots,0}$	0	0.00588	0	0.04843	0	1.0582
		0.13087	0	0.14513	0	0.05719	
	$a_{N,\ldots,0}$	1.00000	0.91134	3.25687	2.41057	3.79969	
		2.16887	1.84761	0.72702	0.30441	0.05719	

9.5.1 Elliptic Design from Amplitude Specifications

Given: ϵ_{dB}, A_s, Ω_p, and Ω_s



Amplitude response constraints

- $\Omega_r = \Omega_s / \Omega_p$
- Find N from the tables for the specified ϵ_{dB} and A_s by choosing N such that the table value of Ω_r is less than or equal to the desired Ω_s / Ω_p
- Frequency scale the resulting $H(s)$ using $s \rightarrow s / \Omega_p$

Example 9.7: An Elliptic Design

Design an elliptic lowpass filter to satisfy the following amplitude specifications: $\epsilon_{dB} = 1$dB, $A_s = 40$dB, $f_p = 10$ kHz, and $f_s = 14.4$ kHz.

- $\Omega_r = 14.4/10 = 1.44$

- From the 1 dB ripple, 40 dB stopband attenuation table we find that $N \geq 5$ meets or exceeds the Ω_r requirement (i.e. in this case the filter will actually have $\Omega_r = 1.2187$, so we can scale Ω_p to be $2\pi \times 10$ krad and then the desired stopband attenuation will actually be achieved at $\Omega = 2\pi \times 1.2187^4 = 76.572$ krad/s or 12.187 kHz

- The normalized $H(s)$ ($\Omega_p = 1$) is

$$H(s) = \big(0.04698s^4 + 0.22007s^2 + 0.22985\big)/ \big(s^5 + 0.92339s^4 + 1.84712s^3 + 1.12923s^2 + 0.78813s + 0.22985\big)$$

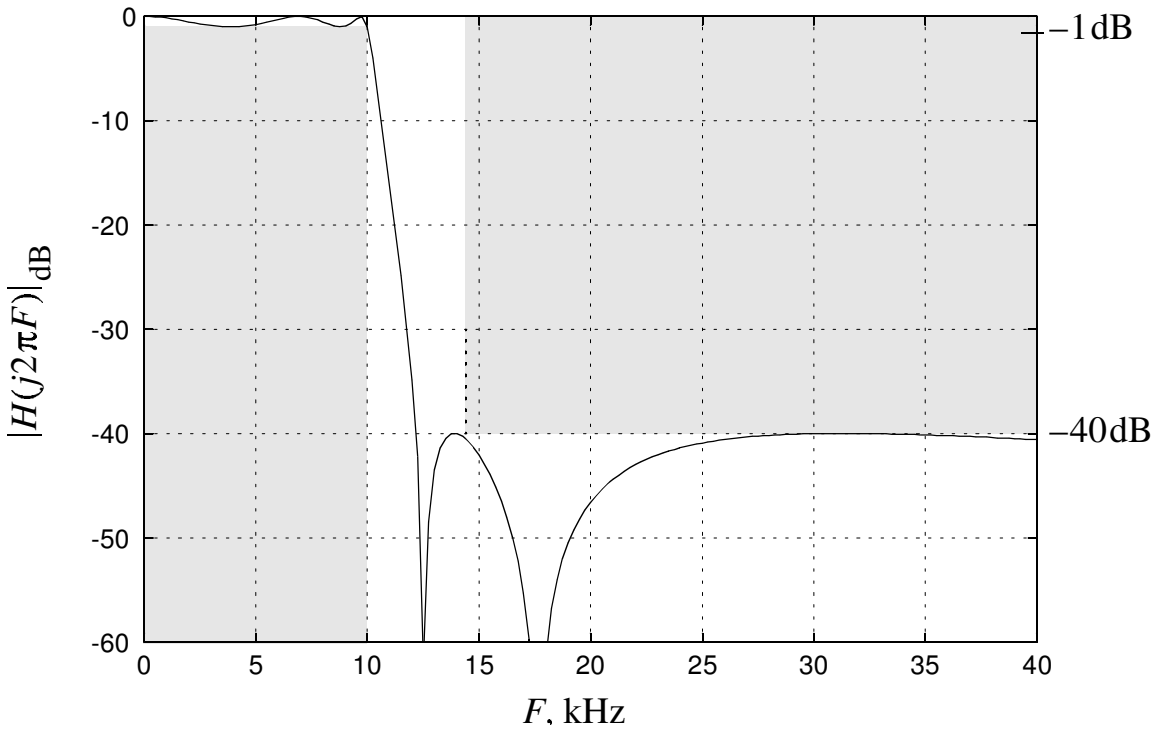
- To complete the design scale $H(s)$ by letting $s \rightarrow s/(2\pi \times 10,000)$

MATLAB Analysis

- We can use MATLAB to directly analyze the above elliptic design

- Frequency scaling is handled in the calculation by letting $\Omega \rightarrow \Omega/(2\pi \times 10,000)$

```
>> f = 0:50000/200:50000; % freq. axis in Hz
>> w = 2*pi*f; % freq. axis in rad/s
>> H = freqs([.04698 0 .22007 0 .22985],...
[1 .92339 1.84712 1.12923 .78813 .22985],...
w/(2*pi*10000));
>> plot(w/(1000*2*pi),20*log10(abs(H))) %in kHz
```



Elliptic magnitude response

MATLAB Filter Design

- An alternative approach is to design the filter completely using MATLAB filter design tools for the *s*-domain

```
>> [n,Wn] = ellipord(2*pi*10000,2*pi*14400,1,40,'s')
n =
     5
Wn =
6.2832e+004
>> Wn/(2*pi)
ans =
     10000 % Fn in Hz
>> [b,a] = ellip(n,1,40,Wn,'s')
b =
```

```
    0   2.9517e+003 -1.1275e-007
    5.4588e+013  -7.9768e+002   2.2509e+023
a =
    1.0000e+000   5.8018e+004   7.2921e+009
    2.8011e+014   1.2283e+019   2.2509e+023
```

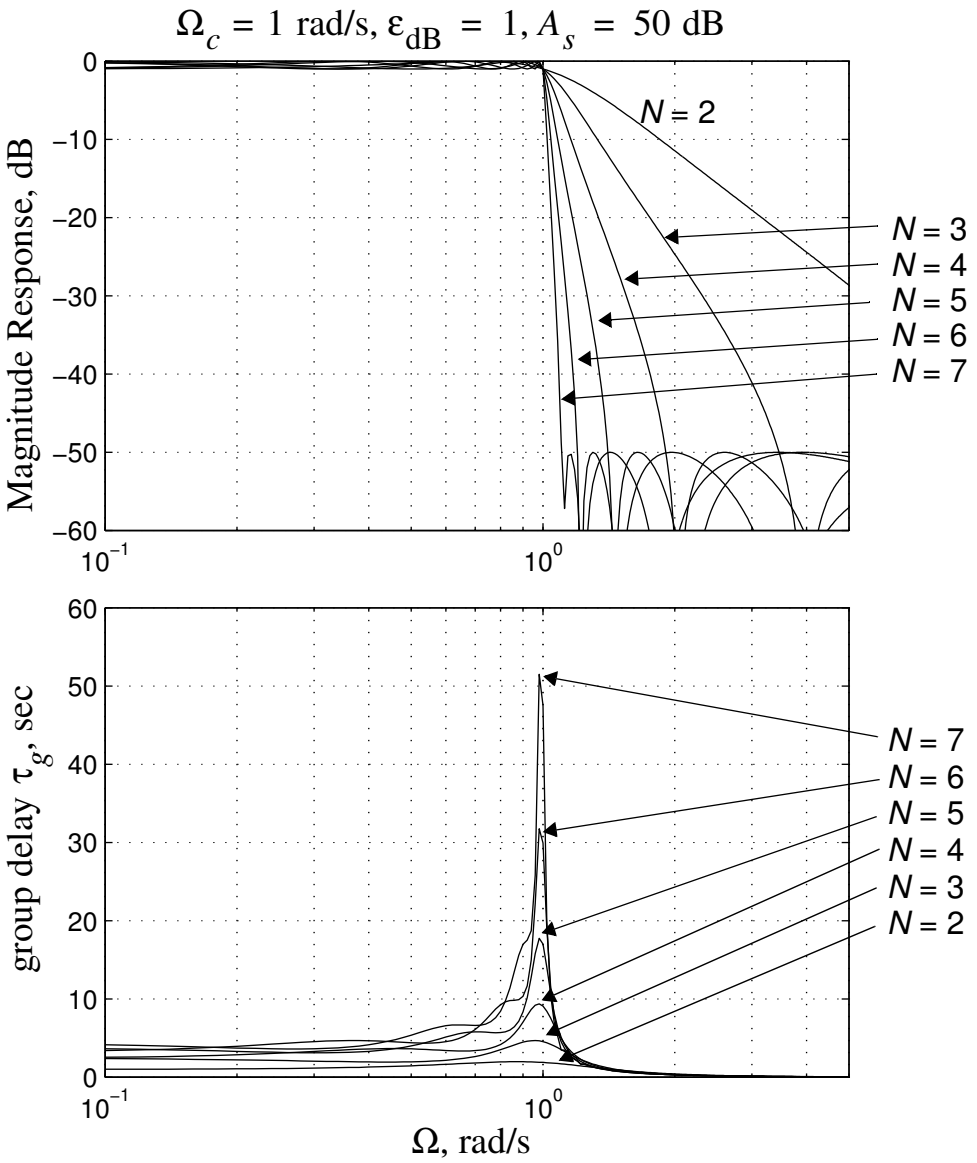
- The above results are identical to those obtained from the table except the scaling factor of `Wn` has been included, e.g.,

$$b_n \rightarrow b_n \times (\Omega_p)^n, \qquad a_n \rightarrow a_n \times (\Omega_p)^n$$

- The table values were generated using MATLAB's `ellip()` with `Wn = 1` in the first place

9.5.2 Amplitude Response and Group Delay Summary

Consider an elliptic design with $\Omega_s = 1$ rad/s, $\epsilon_{\text{dB}} = 1$, and $A_s = 50$ dB, and vary the filter order from 2 to 7.



9.6 The Design of Discrete-Time IIR Filters from Analog Prototypes

In this section we will discuss the design of discrete-time IIR filters using *classical* continuous-time design procedures. In particular we will consider design by impulse invariance and the bilinear transformation.

9.6.1 Impulse Invariant Design

In section 3.4.2 of the text (notes page 4-23, Example 4-4), the impulse invariant design was first introduced. Recall that we chose the discrete-time impulse response to be a sampled version of the corresponding continuous-time impulse response.

- Let $h_c(t) \overset{\mathcal{L}}{\longleftrightarrow} H_c(s)$. Then set

$$h[n] = T_d h_c(nT_d)$$

  where T_d is the *design* sampling period, which may differ from the A/D and D/A sampling period

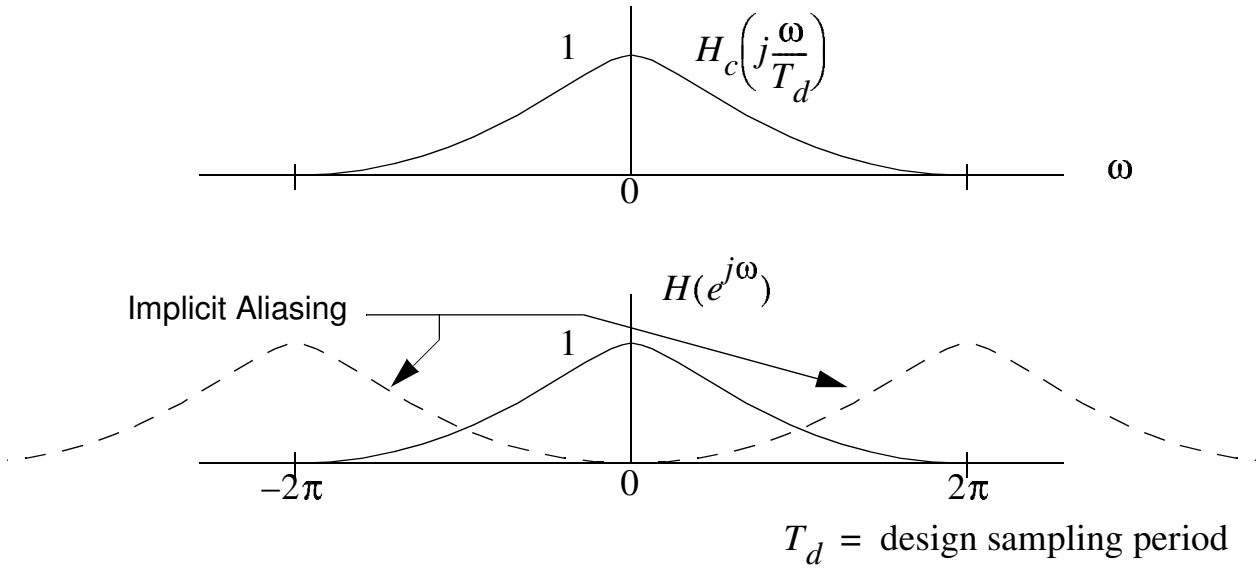
- We thus can write

$$H(e^{j\omega}) = \sum_{k=-\infty}^{\infty} H_c\left(j\frac{\omega}{T_d} + j\frac{2\pi}{T_d}k\right)$$

- Recalling that $H(e^{j\omega}) = H(z)|_{z=e^{sT_d}}$ we can also write that the system function is given by

$$H(z) = \mathcal{Z}\left[\left\{\mathcal{L}^{-1}[H_c(s)]\right\}\Big|_{t=nT_d}\right]$$

- Note that $H(e^{j\omega})$ is just an aliased version of $H_c(j\omega/T_d)$



Aliasing inherent in the impulse invariant technique

- The stopband characteristic of $H_c(j\,\Omega)$ is maintained only if the aliased tails are small
- To reduce aliasing:
  1. Restrict $H_c(j\,\Omega)$ to be lowpass with a monotonic stopband
  2. Decrease T_d (increase f_s) or decrease the filter cutoff frequency

Basic Design Technique

- Expand $H_c(s)$ using partial fractions (for this development we assume there are no repeated poles)

$$H_c(s) = \sum_{k=1}^{N} \frac{A_k}{s - s_k}$$

- Note: In the following analysis a *gain factor* G is introduced which also includes T_d. The gain factor allows the dc filter gains to be equal.

- Inverse Laplace transforming $H_c(s)$ we get

$$h_c(t) = \sum_{k=1}^{N} A_k e^{s_k t} u(t)$$

so

$$h[n] = \underbrace{G}_{\text{gain factor}} \sum_{k=1}^{N} A_k (e^{s_k T_d})^n u[n]$$

and

$$H(z) = G \sum_{k=1}^{N} \frac{A_k}{1 - e^{s_k T_d} z^{-1}}$$

- The s-plane poles are mapped to the z-plane using

$$s = s_k \Rightarrow z = p_k = e^{s_k T_d}$$

- To match the dc gains set $H(e^{j0}) = H(1) = H_c(0)$ which implies that we set

$$G \sum_{k=1}^{N} \frac{A_k}{1 - e^{s_k T_d}} = -\sum_{k=1}^{N} \frac{A_k}{s_k}$$

Example 9.8: Design from a Rational $H(s)$

Let

$$H_c(s) = \frac{0.5(s+4)}{(s+1)(s+2)} = \frac{1.5}{s+1} - \frac{1}{s+2}$$

- Inverse Laplace transforming yields

$$h_c(t) = [1.5e^{-t} - e^{-2t}]\,u(t)$$

- Sampling $h_c(t)$ every T_d seconds and gain scaling we obtain

$$h[n] = G[1.5e^{-nT_d} - e^{-2nT_d}]\,u[n]$$

  which implies that

$$H(z) = G\left[\frac{1.5}{1 - e^{-T_d}z^{-1}} - \frac{1}{1 - e^{-2T_d}z^{-1}}\right]$$

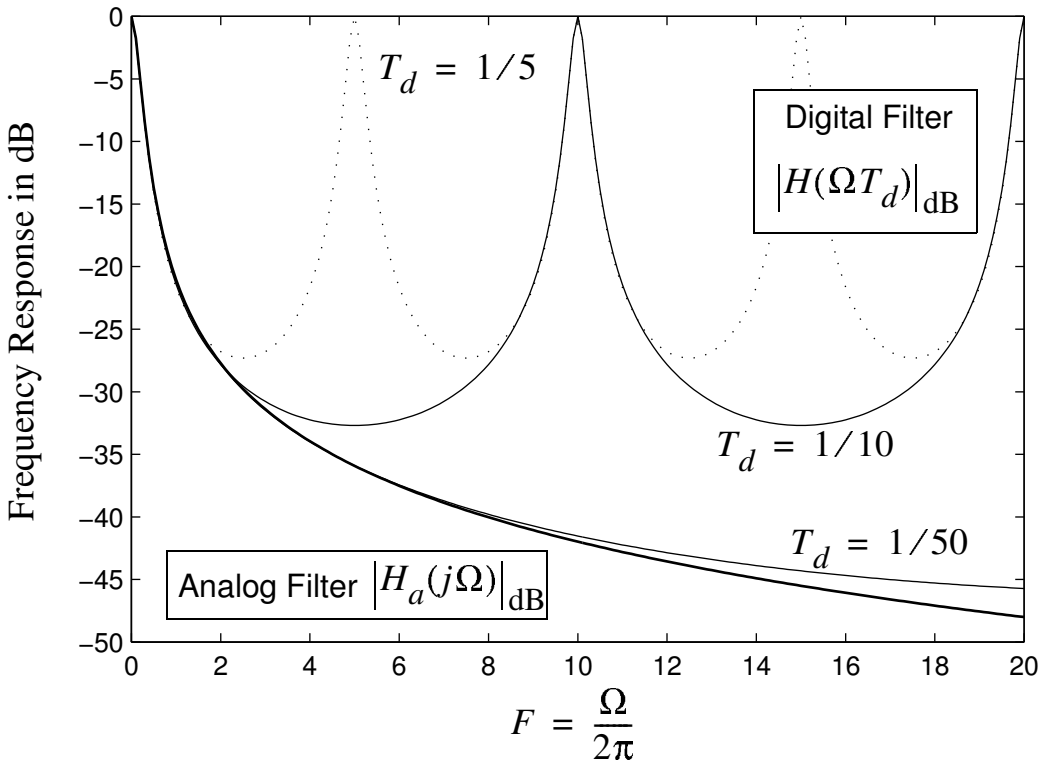
- Setting $H_c(0) = H(1)$ requires that

$$G = \left[\frac{1.5}{1 - e^{-T_d}} - \frac{1}{1 - e^{-2T_d}}\right]^{-1}$$

MATLAB Analysis

- From the above analysis we can have MATLAB directly evaluate $H(e^{j\omega})$ for various values of T_d or equivalently sampling rate $f_s = 1/T_d$; here we have used $T + 1/5$, 1/10, and 1/50 sec.

```
>> f = 0:20/200:20;
>> w = 2*pi*f;
>> Td = 1/10; G = 1/(1.5/(1-exp(-Td)) - 1/(1-exp(-2*Td)));
>> H = G*(1.5./(1 - exp(-Td)*exp(-j*w*Td)) ...
- 1./(1-exp(-2*Td)*exp(-j*w*Td)));
>> Ha = freqs([.5 2],[1 3 2],w);
>> plot(f,20*log10(abs(Ha)))
>> hold
Current plot held
>> plot(f,20*log10(abs(H)),'-.')
```



Frequency response of impulse invariant design for various T_d values

MATLAB Impulse Invariant Synthesis

- An alternative approach is to use MATLAB to synthesize the z-domain impulse invariant deign directly from s-domain rational transfer function

- The key function is `impinvar()` which converts a rational function in s to a rational function in z for a given sampling frequency (see the chapter appendix)

```
>> as = conv([1 1],[1 2]) % multiply out den. poly
    as = 1     3     2
>> bs = 0.5*[1 4]
    bs = 0.5000    2.0000
>> [bz10,az10] = impinvar(bs,as,10); % fs = 10 Hz
```

```
    bz10 =  0.5000   -0.3233
    az10 =  1.0000   -1.7236    0.7408
>> [bz50,az50] = impinvar(bs,as,50); % fs = 50 Hz
    bz50 =  0.5000   -0.4610
    az50 =  1.0000   -1.9410    0.9418
>> [Hc,wc] = freqs(bs,as);
>> [H10,F10] = freqz(bz10,az10,256,'whole',10);
>> [H50,F50] = freqz(bz50,az50,256,'whole',50);
>> plot(wc/(2*pi),20*log10(abs(Hc)))
ż hold
Current plot held
>> plot(F10,20*log10(abs(H10/H10(1))),'--')
>> plot(F50,20*log10(abs(H50/H50(1))),'-.')
```

- In the above gain normalization to unity at dc is accomplished by dividing through by $H(1)$; the plots are not shown as they are identical to the previous results
- The MATLAB synthesis produces $H(z)$ in a nice rational form via the polynomial coefficient vectors `bz10` and `az10`, etc.

Designing for Discrete-Time Specifications

- If the design is required to meet specific discrete time specifications, then we begin by transforming the discrete time specifications to continuous time using $\Omega_i = \omega_i / T_d$ for $i = p$ and s
- Next $H_c(s)$ is found using the approximation methods described earlier in this chapter
- Given $H_c(s)$ we then proceed to find $H(z)$ using impulse invariance

Example 9.9: A Chebyshev Design

Design a Chebyshev type I lowpass filter to satisfy the following amplitude specifications: $\epsilon_{dB} = 1$ dB, $A_s = 20$ dB, $\omega_p = 0.1\pi$, and $\omega_s = 0.3\pi$.

- Using the design formula for N

$$N = \lceil 2.08 \rceil = 3$$

  Note: $\Omega_s/\Omega_p = \omega_s/\omega_p$

- The normalized system function is

$$H_c(s) = \frac{0.0152}{s^3 + 0.3105s^2 + 0.1222s + 0.0152}$$

  with poles at

$$0.1549e^{j180^\circ}, \quad 0.3132e^{\pm j104.4^\circ}$$

- To frequency scale $H_c(s)$ let

$$s \rightarrow \frac{s}{\omega_p/T_d} \Rightarrow s_k \rightarrow s_k\frac{\omega_p}{T_d}$$

  so in the z-domain

$$p_k = e^{T_d s_k \omega_p/T_d} = e^{s_k\omega_p}$$

  Note that when the design originates from discrete time specifications the poles of $H(z)$ are independent of T_d

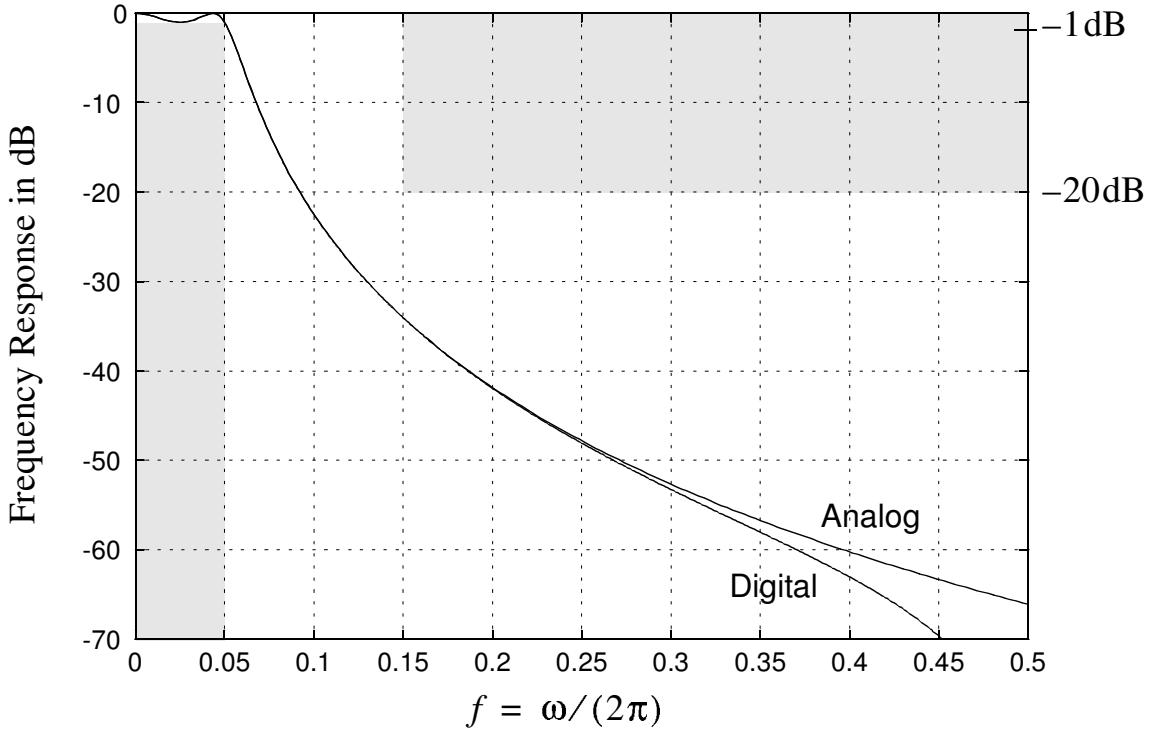
- To complete the design expand $H_c(s)$ in a partial fraction expansion and then use the A_k's and p_k's to define $H(z)$ (left as an exercise)

MATLAB Impulse Invariant Synthesis

- As in the previous example we can again use the MATLAB function `impinvar()` to synthesize an impulse invariant filter directly from amplitude response specifications

- The design will assume that $T_d = 1$

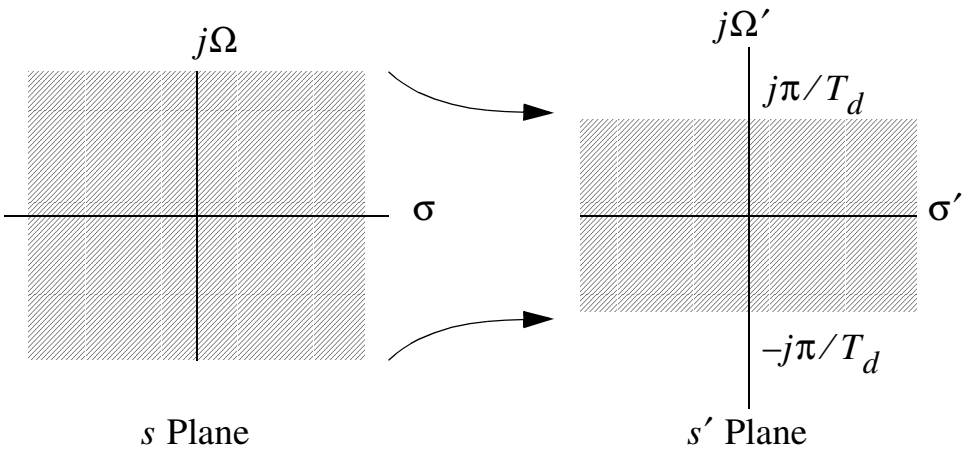
```
>> [n,Wn] = cheb1ord(0.1*pi,0.3*pi,1,20,'s')
n =
    3
Wn =
    0.3142
>> [bs,as] = cheby1(n,1,Wn,'s')
bs =
         0    0.0000    0.0000    0.0152
as =
    1.0000    0.3105    0.1222    0.0152
>> W = 0:4/200:4;
>> Hc = freqs(bs,as,W); %Compute s-domain freq. resp.
>> [bz,az] = impinvar(bs,as,1)
>> %impulse invariant design using Fs = 1
bz =
    0.0000    0.0068    0.0061
az =
    1.0000   -2.6223    2.3683   -0.7331
>> [H,w] = freqz(bz,az); %Compute z-domain aliased resp.
>> plot(W/(2*pi),20*log10(abs(Hc)))
>> hold
Current plot held
>> plot(w/(2*pi),20*log10(abs(H/H(1)))) %gain normalize at dc
```



Frequency response of Chebyshev impulse invariant design for $f_s = 1$ Hz

9.6.2 Bilinear Transformation Design

The impulse invariant technique suffers from aliasing due to the many-to-one mapping $z = e^{sT_d}$. To correct this problem employ a one-to-one mapping from say s to s' which compresses the entire s-plane into a strip.



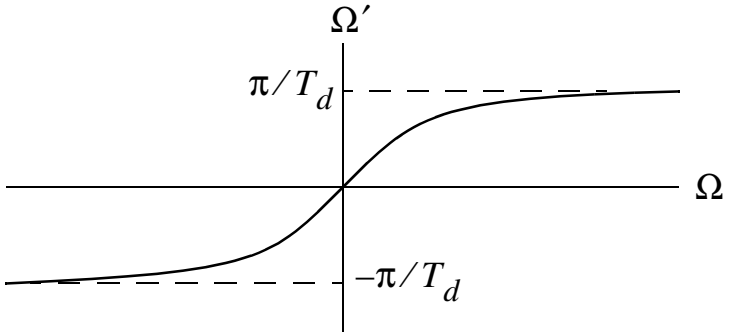
Desired mapping from s to s' to avoid aliasing

- The one-to-one mapping of interest is

$$s' = \frac{2}{T_d} \tanh^{-1}\left(\frac{sT_d}{2}\right)$$

- Consider just the $j\,\Omega$ axis for a moment

$$\Omega' = \frac{2}{T_d} \tan^{-1}\left(\frac{\Omega T_d}{2}\right)$$



Mapping from Ω to Ω'

- Clearly the entire Ω-axis has been compressed (*warped*) to fit the interval $[-\pi/T_d, \pi/T_d]$

- Note that for ΩT_d small the mapping is approximately linear, i.e. $\Omega \approx \Omega'$, $|\Omega| \ll 1/T_d$

- The complete mapping from s to z is obtained by setting $z = e^{s'T_d}$ or $s' = \frac{1}{T_d} \ln z$

$$s' = \frac{1}{T_d} \ln z = \frac{2}{T_d} \tanh^{-1}\left(\frac{sT_d}{2}\right)$$

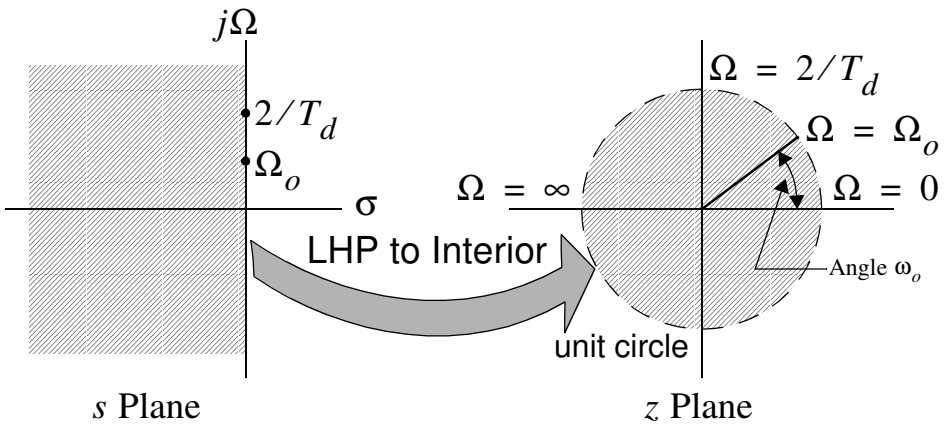
or

$$s = \frac{2}{T_d} \tanh\left(\frac{\ln z}{2}\right) = \frac{2}{T_d} \left.\frac{1 - e^{-2x}}{1 + e^{-2x}}\right|_{x=\frac{\ln z}{2}}$$

- Finally we have the bilinear transform equations:

$$s = \frac{2}{T_d}\left(\frac{1 - z^{-1}}{1 + z^{-1}}\right) \quad \text{or} \quad z = \frac{1 + \frac{T_d}{2}s}{1 - \frac{T_d}{2}s}$$

- Mapping properties:



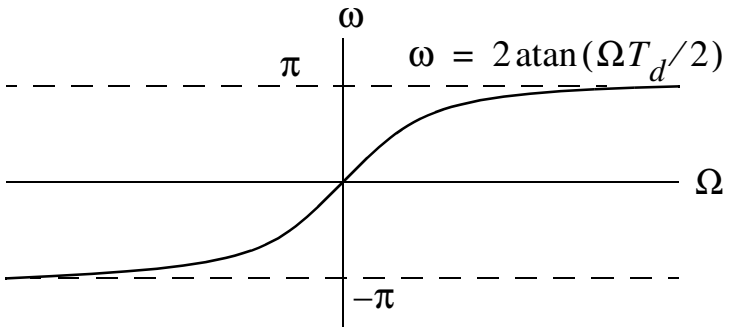
Mapping from Ω to Ω'

- The basic filter design equation is

$$H(z) = \left. H_c(s)\right|_{s=\frac{2}{T_d}\left(\frac{1-z^{-1}}{1+z^{-1}}\right)}$$

- The discrete-time frequency variable must be $\omega = \Omega' T_d$ with the frequency axis mappings being

$$\Omega = \frac{2}{T_d}\tan(\omega/2) \text{ or } \omega = 2\tan^{-1}(\Omega T_d/2)$$



Mapping from Ω to ω

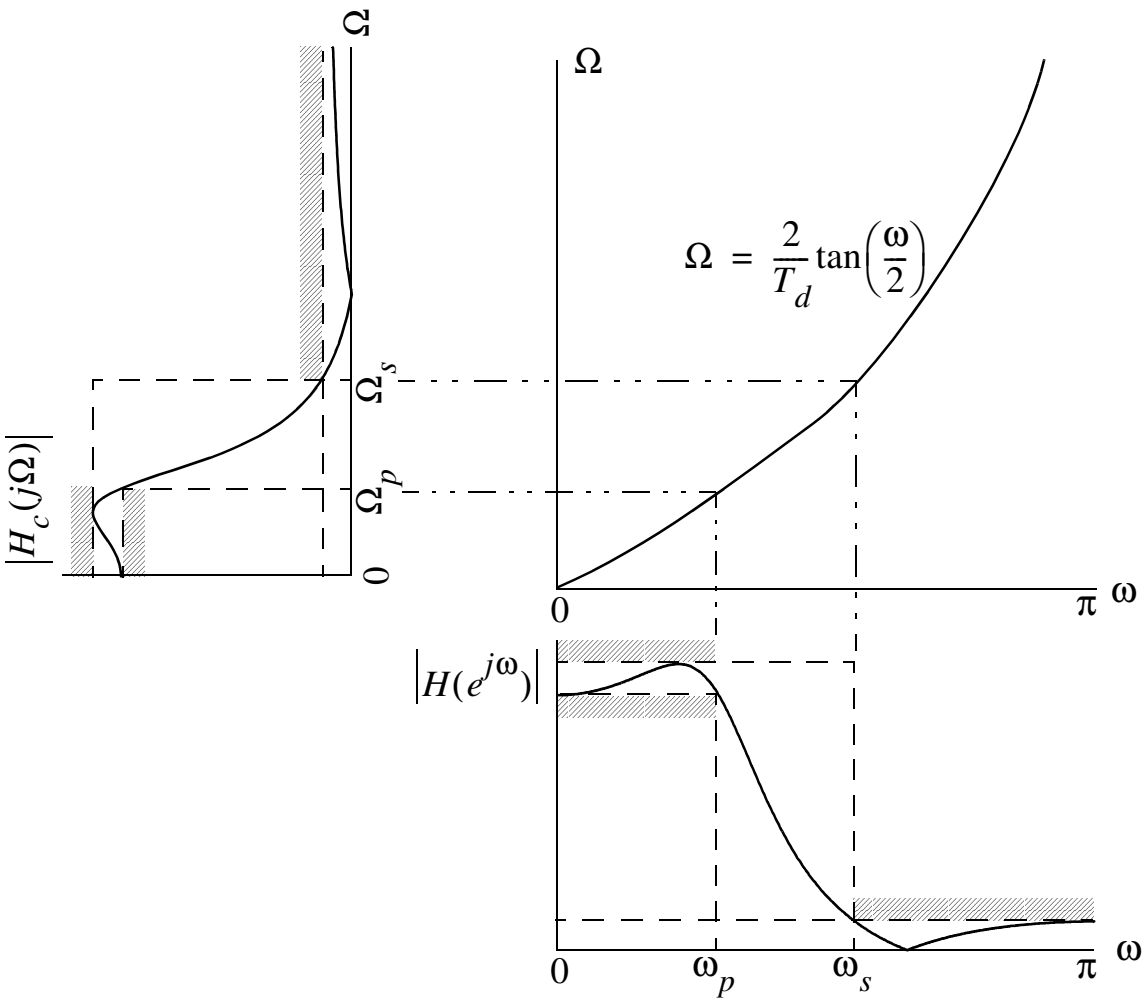
- Note: Lowpass designs such as the Butterworth and Chebyshev type I, which have N zeros at infinity in the s-plane, will have N zeros at $z = -1$ in the z-plane as a result of the bilinear mapping

Frequency Warping

- The nonlinear transformation from Ω to ω is known as *frequency warping*

- The discrete-time frequency response is

$$H(e^{j\omega}) = H_c(j\Omega)|_{\Omega=\frac{2}{T_d}\tan(\omega/2)}$$



Frequency axis warping (compression) of a continuous-time lowpass filter into a discrete-time filter

- In a practical design in order to preserve the desired discrete-time critical frequencies such as ω_p and ω_s, the corresponding continuous-time frequencies Ω_p and Ω_s must be *pre-warped* using

$$\Omega_i = \frac{2}{T_d}\tan\left(\frac{\omega_i}{2}\right)$$

- Note that the frequency axis compression can make the transition ratio ω_s/ω_p less than Ω_s/Ω_p. The net result being that

the actual filter order N may be less than that needed for a pure continuous-time design.

Bilinear Transformation Design Steps

1. Given discrete-time amplitude specifications, corresponding critical frequencies, and the sampling spacing T_d, we first design a corresponding continuous-time filter, $H_c(s)$, using pre-warped critical frequencies

$$\Omega_p = \frac{2}{T_d}\tan\left(\frac{\omega_p}{2}\right), \quad \Omega_s = \frac{2}{T_d}\tan\left(\frac{\omega_s}{2}\right)$$

   Note: For convenience T_d may be set to unity since upon transforming $H_c(s)$ back to $H(z)$ it will cancel out.

   The first step in finding $H_c(s)$ is to determine N and any other needed parameters.

2. Map the $H_c(s)$ design to $H(z)$ using the bilinear transform, that is set

$$H(z) = H_c\left(\frac{2}{T_d}\frac{1-z^{-1}}{1+z^{-1}}\right)$$

   Important Simplifications for step 2:

   - The algebraic substitution of $\frac{2}{T_d}\frac{1-z^{-1}}{1+z^{-1}}$ for every s in $H_c(s)$ is tedious
   - Consider developing a general formula for calculating the z-plane poles and zeros directly from the s-plane poles and zeros

- In general

$$H_c(s) = \underbrace{K}_{\text{Gain Factor}} \frac{\prod_{m=1}^{M_c}(s-\sigma_m)}{\prod_{k=1}^{N}(s-s_k)}$$

- Finite poles and zeros in the s-plane map to finite poles and zeros in the z-plane, while zeros at infinity in the s-plane map to zeros at $z=-1$ in the z-plane, thus

$$H(z) = b_o(1+z^{-1})^{N-M_c}\frac{\prod_{m=1}^{M_c}(1-z_m z^{-1})}{\prod_{k=1}^{N}(1-p_k z^{-1})}$$

  where b_o is a new gain factor

- From the bilinear transformation

$$z_m = \frac{1+\frac{T_d}{2}\sigma_m}{1-\frac{T_d}{2}\sigma_m}, \quad p_k = \frac{1+\frac{T_d}{2}s_k}{1-\frac{T_d}{2}s_k}$$

  Note: σ_m, s_k, z_m, and p_k are in general complex.

- The gain factor b_o is used to equate the filter gains at some prescribed frequency. For lowpass filters match the dc gains (i.e. set $H(1) = H_c(0)$).

Example 9.10: A Bilinear Lowpass Design

Design a discrete-time lowpass filter to satisfy the following amplitude specifications: $\epsilon_{dB} = 3.01$ dB, $A_s = 15$ dB, $\omega_p = 0.5\pi$, and $\omega_s = .75\pi$. Assume a monotonic passband and stopband is desired. Assume $1/T_d = f_s = 2$ kHz.

- The prewarped critical frequencies are:

$$\Omega_p = 2 \cdot 2000 \tan(0.5\pi/2) = 4000\text{rad/s}$$

$$\Omega_s = 2 \cdot 2000 \tan(0.75\pi/2) = 9657 \text{rad/s}$$

Note: $\omega_s/\omega_p = 1.5 < \Omega_s/\Omega_p = 2.414$ as expected

- Since both the passband and stopband are required to be monotonic, a Butterworth approximation will be used

$$N = \left\lceil \frac{\log_{10}\left[\frac{10^{3.01/10}-1}{10^{15/10}-1}\right]}{2\log_{10}(4000/9657)} \right\rceil$$
$$= \lceil 1.9412 \rceil = 2$$

- From the Butterworth design tables we can immediately write

$$H_c(s) = \left. \frac{1}{s^2 + \sqrt{2}s + 1} \right|_{s=\frac{s}{4000}}$$

- Now find $H(z)$ by first noting that $M_c = 0$, $N = 2$, and

$$s_k = 4000 e^{\pm j135^\circ}, \quad k = 1, 2$$

- Using the pole/zero mapping formulas

$$p_1 = \frac{1 + \frac{1}{4000}s_1}{1 - \frac{1}{4000}s_1} = \frac{1 + e^{j135^\circ}}{1 - e^{j135^\circ}} = 0.414 e^{j\,90^\circ}$$

$$p_2 = \frac{1 + e^{-j135^\circ}}{1 - e^{-j135^\circ}} = 0.414 e^{-j\,90^\circ} = p_1^*$$

- We can now write

$$H(z) = b_o(1 + z^{-1})^2 \frac{1}{(1 - p_1 z^{-1})(1 - p_1^* z^{-1})}$$

- Find b_o by setting $H(1) = H_c(0)$

$$1 = \frac{b_o(1+1)^2}{(1-p_1)(1-p_1^*)} = \frac{4b_o}{1+|p_1|^2}$$

or

$$b_o = \frac{1+0.414^2}{4} = 0.293$$

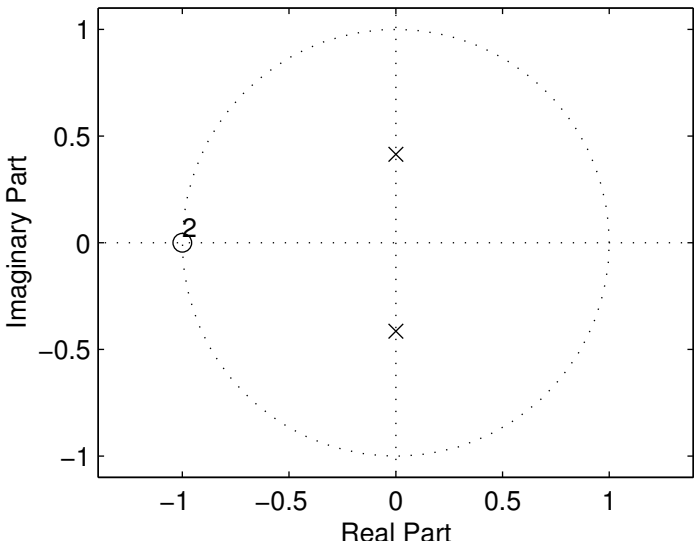
- Finally after multiplying out the numerator and denominator we obtain

$$H(z) = \frac{1+2z^{-1}+z^{-2}}{3.4142+0.5857z^{-2}}$$

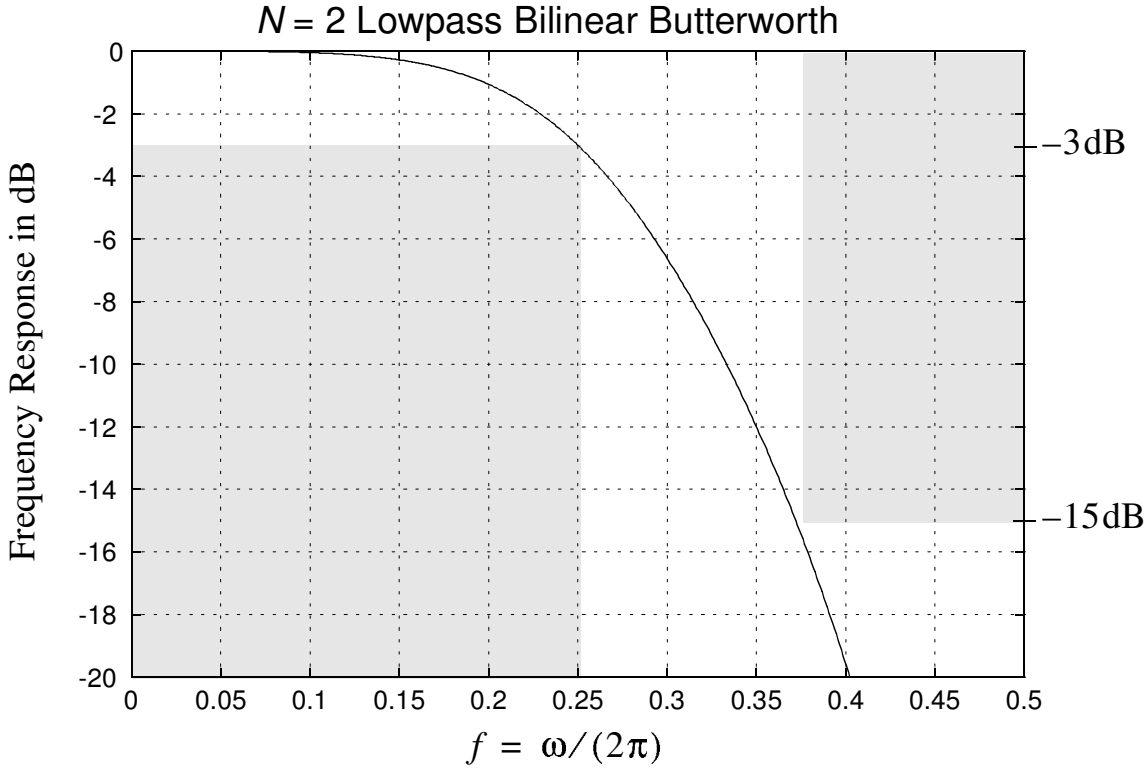
MATLAB Bilinear Filter Analysis

- We can use MATLAB to complete the analysis of the above filter design by computing the pole-zero plot, the frequency response magnitude in dB, and the group delay

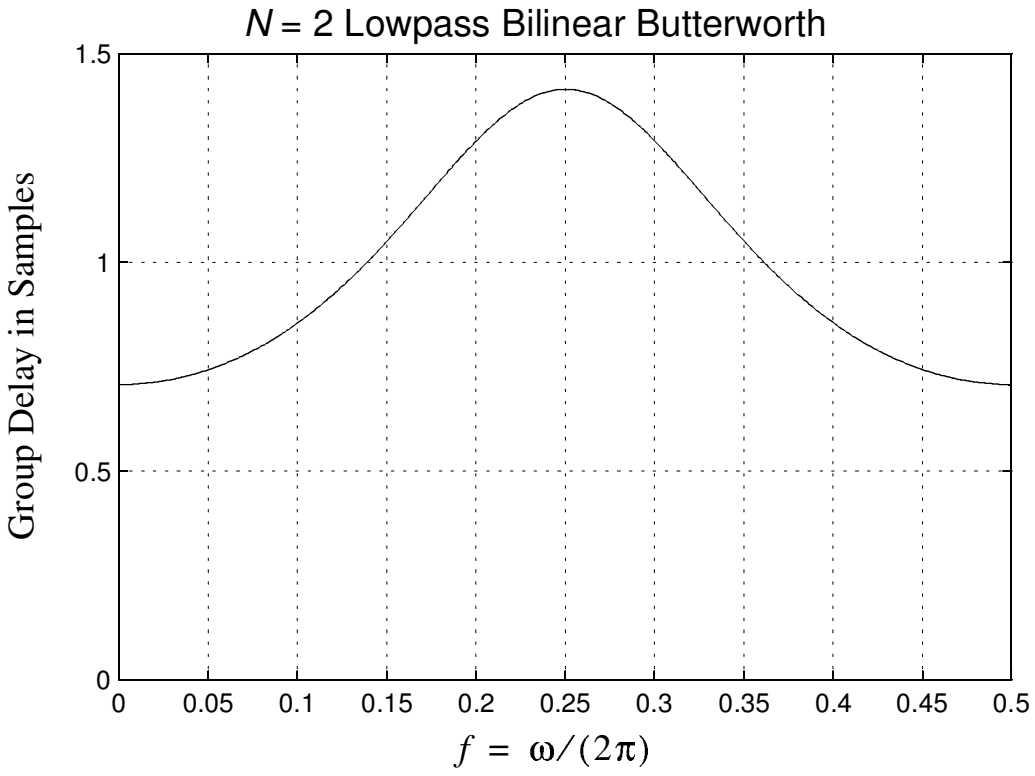
```
>> zplane([1 2 1],[3.412 0 0.5857])

>> [H,F] = freqz([1 2 1],[3.412 0 0.5857],512,1);
>> plot(F,20*log10(abs(H)))
>> grid
>> axis([0 .5 -20 0])
>> [Hg,F] = grpdelay([1 2 1],[3.412 0 0.5857],512,1);
>> plot(F,Hg)
>> grid
>> axis([0 .5 0 1.5])
```



Pole-zero map from MATLAB analysis



Magnitude response in dB from MATLAB analysis



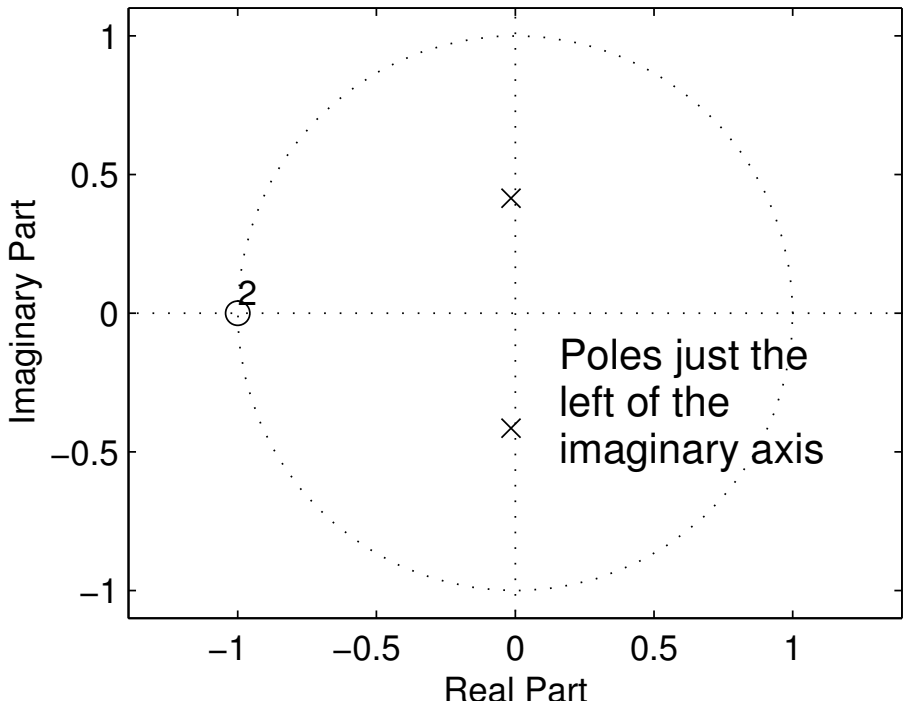
Group delay in samples from MATLAB analysis

MATLAB Bilinear Filter Design

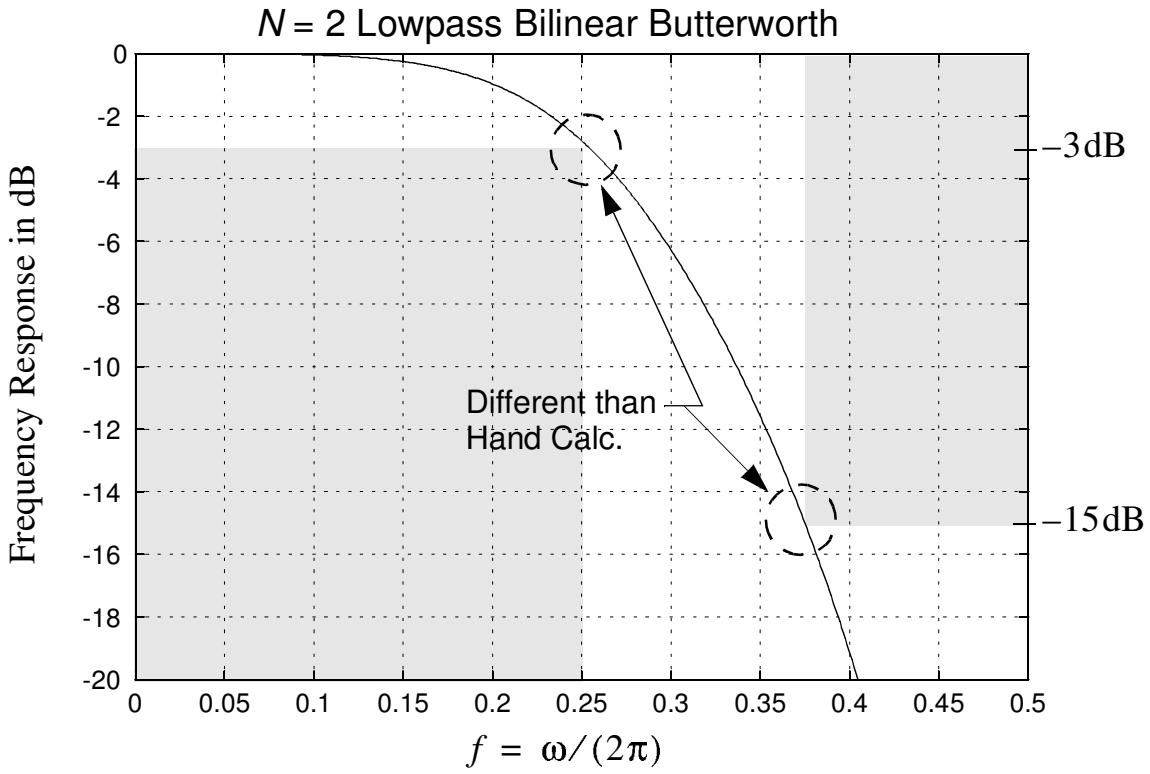
- In MATLAB a complete bilinear filter synthesis, for a specific analog prototype, can be done in one step
- We simply use the filter design functions, in this case since we desire a Butterworth design we use `butterord` and `butter`

```
>> % In a digital design the critical frequencies are
>> % entered as w/pi, i.e., they lie on the interval
>> % (0,1).
>> [n,Wn] = buttord(0.5,0.75,3.01,15)
>> n =
    2
>> Wn =
    0.5083
>> [b,a] = butter(n,Wn)
```

```
b =
    0.3005    0.6011    0.3005
a =
    1.0000    0.0304    0.1717
>> zplane(b,a)
>> [H,F] = freqz(b,a,512,1);
>> plot(F,20*log10(abs(H)))
>> grid
>> axis([0 .5 -20 0])
>> [Hg,F] = grpdelay(b,a,512,1);
>> plot(F,Hg)
>> grid
>> axis([0 .5 0 1.5])
```



Pole-zero map from the MATLAB design



Magnitude response in dB from the MATLAB design

- Note that the hand calculation and the MATLAB design get slightly different results
- In the MATLAB design the filter cutoff frequency ω_c is not simply set to ω_s, but rather to a value slightly higher so as to meet or exceed both the passband and stopband requirements
- The poles in this case do not lie exactly on the imaginary axis, but slightly to the left

Example 9.11: Rework of Ex 9.8 with Bilinear Transform

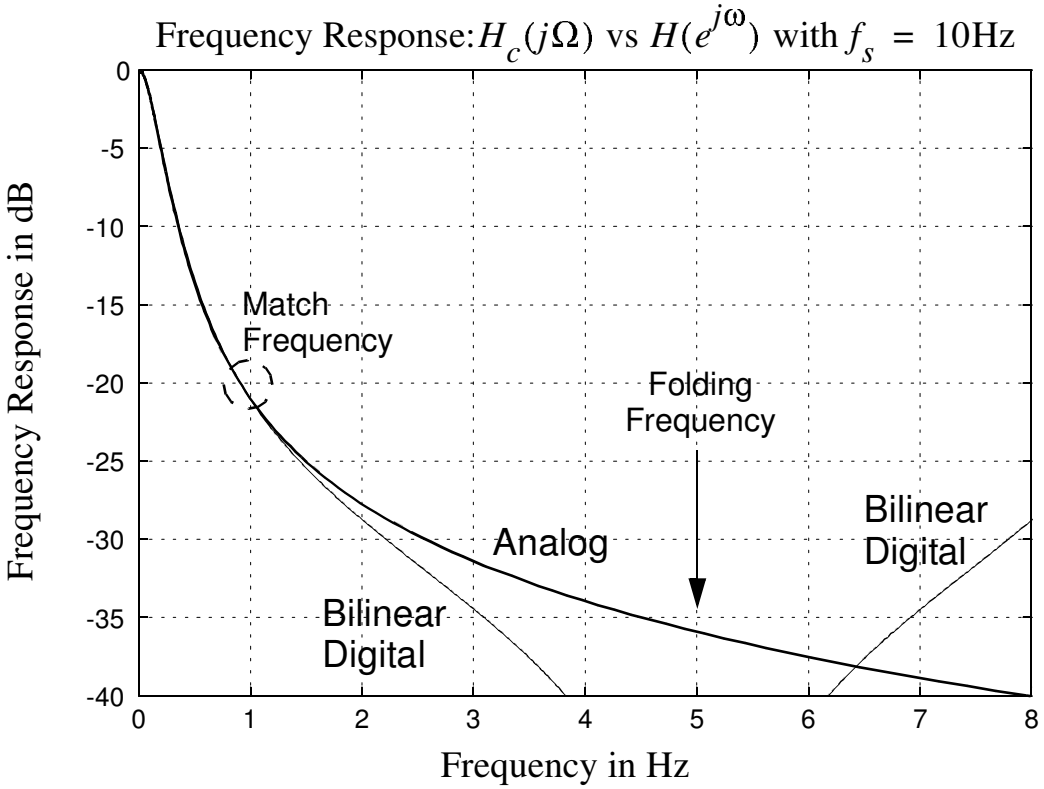
In the original example we started from am s-domain specification

$$H_c(s) = \frac{0.5(s+4)}{(s+1)(s+2)}$$

- A z-domain equivalent of this system function can be designed using the bilinear transform
- Since the bilinear transform warps the frequency axis we must specify an analog vs digital mathching frequency as well as the sampling frequency
- For this example we will choose $f_s = 10$ Hz and the transformation invariant matching frequency to be 1 Hz
- The key function from MATLAB that design the bilinear transform filter for a given s-domain description is `bilinear`
- To use `bilinear` we load in the numerator and denominator s-domain polynomials followed by the sampling frequency in Hz and the matching frequency in Hz

```
>> as = conv([1 1],[1 2])
as =
     1     3     2
>> bs = 0.5*[1 4];
>> [bz,az] = bilinear(bs,as,10,1)
bz =
    0.0269    0.0092   -0.0177
az =
```

```
    1.0000   -1.7142    0.7326
>> [Hc,wc] = freqs(bs,as);
>> [H10,F10] = freqz(bz,az,512,'whole',10);
>> plot(wc/(2*pi),20*log10(abs(Hc)))
>> grid
>> hold
Current plot held
>> plot(F10,20*log10(abs(H10)),'r')
>> axis([0 8 -40 0])
```

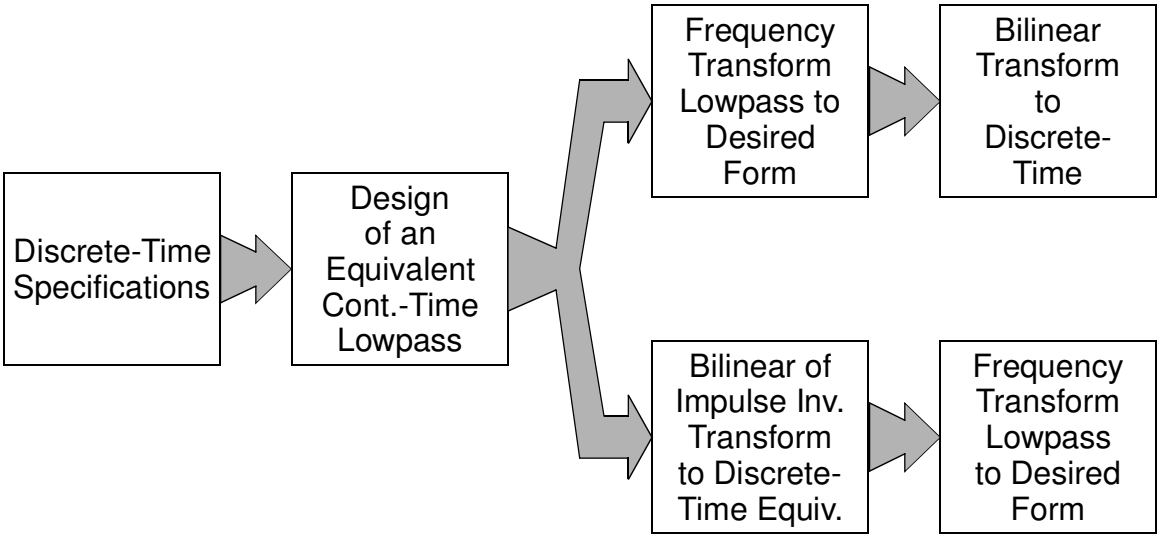


Magnitude response in dB of analog and bilinear digital design with $f_s = 10$ Hz

9.7 Frequency Transformations

Of the classical filter design techniques discussed thus far all have produced lowpass designs. Ultimately a highpass, bandpass, or bandstop design may be desired. The appropriate frequency or *spectral* transformation on either a continuous-time or discrete-time lowpass filter is used to yield the desired filter.

Assuming that the desired end result is a discrete-time design, as the following figure indicates, one of two paths may be followed.



The two procedures that may be followed to frequency transform a continuous-time lowpass filter to a discrete-time filter.

- If a bilinear transformation is used then either a continuous-time or a discrete-time transformation may be used
- If the underlying design technique uses impulse invariance, then due to aliasing problems only a discrete-time transformation should be used

9.7.1 Continuous-Time Transformations

In the following frequency transformations the lowpass filter is assumed to have a one rad/s cutoff frequency (i.e. $\Omega_p = 1$).

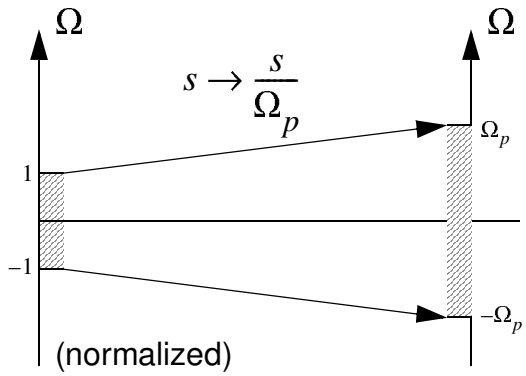
- Lowpass Frequency Scaling:

  This transformation has been used already, but is repeated here for completeness.

$$s \longrightarrow \frac{s}{\Omega_p}$$

  or

$$H_{\text{desired}}(s) = H_c\left(\frac{s}{\Omega_p}\right)$$



Lowpass to Lowpass Mapping

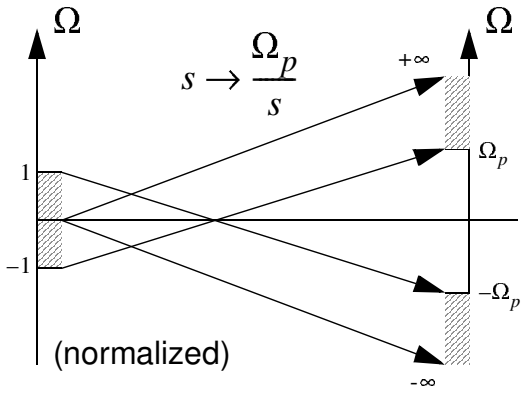
- Lowpass to Highpass:

  Assuming the desired cutoff frequency is Ω_p, the transformation is

$$s \longrightarrow \frac{\Omega_p}{s}$$

or

$$H_{\text{desired}}(s) = H_c\left(\frac{\Omega_p}{s}\right)$$



Lowpass to Highpass Mapping

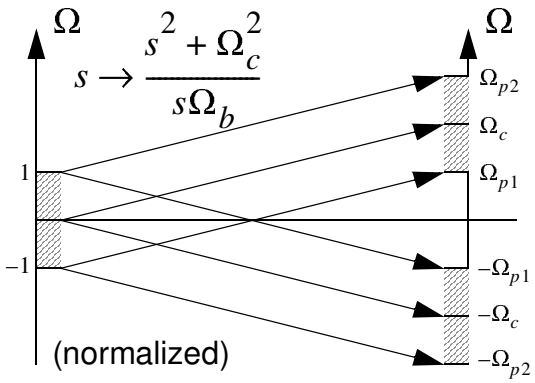
- Lowpass to Bandpass:

Assuming the desired lower and upper cutoff frequency are respectively Ω_{p1} and Ω_{p2} the transformation is

$$s \longrightarrow \frac{s^2 + \Omega_c^2}{s\,\Omega_b}$$

where

$$\Omega_c = \sqrt{\Omega_{p1}\Omega_{p2}}$$
$$= \text{center freq.}$$
$$\Omega_b = \Omega_{p2} - \Omega_{p1}$$
$$= \text{bandwidth}$$



Lowpass to Bandpass Mapping

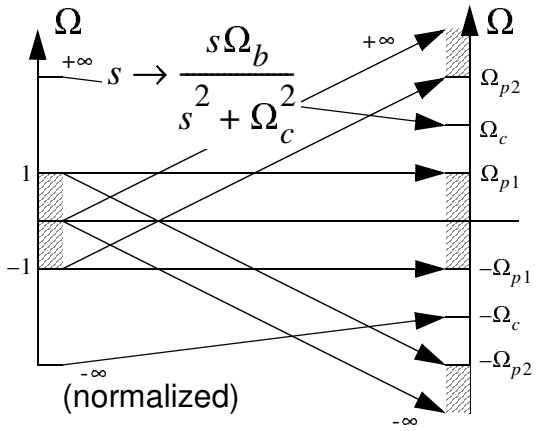
- Lowpass to Bandstop:

Assuming the desired lower and upper cutoff frequency are respectively Ω_{p1} and Ω_{p2} the transformation is

$$s \longrightarrow \frac{s\Omega_b}{s^2 + \Omega_c^2}$$

where

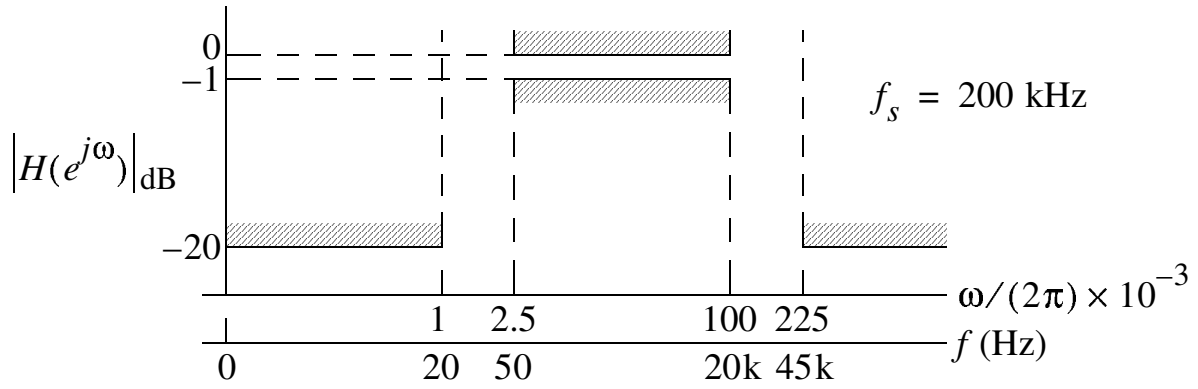
$$\Omega_c = \sqrt{\Omega_{p1}\Omega_{p2}}$$
$$= \text{center freq.}$$
$$\Omega_b = \Omega_{p2} - \Omega_{p1}$$
$$= \text{bandwidth}$$



Lowpass to Bandstop Mapping

Example 9.12: A Bandpass Bilinear Design

Bandpass design using a continuous-time (analog) transformation followed by bilinear transformation to obtain $H(z)$. The desired amplitude specifications of an equivalent C/D-$H(e^{j\omega})$-D/C system with $f_s = 200$ kHz are: $\epsilon_{dB} = 1$ dB, $A_s = 20$ dB, $f_{s1} = 20$ Hz, $f_{p1} = 50$ Hz, $f_{p2} = 20$ kHz, and $f_{s2} = 45$ kHz. Design the filter using a Chebyshev type I approximation.



Bandpass amplitude specifications

- The design steps may be summarized as follows:

  1. Map the digital bandpass specifications back to analog bandpass specifications using prewarping
  2. Map the analog bandpass filter back to an analog lowpass prototype $H_c(s)$ of order N
  3. Transform $H_c(s)$ using the analog lowpass to bandpass mapping
  4. Transform $H_{bp}(s)$ to $H(z)$ using the bilinear transform

- The discrete-time critical frequencies are: $\omega_{s1} = 10^{-4} \times 2\pi$, $\omega_{p1} = 2.5 \times 10^{-4} \times 2\pi$, $\omega_{p2} = 0.1 \times 2\pi$, and $\omega_{s2} = 0.225 \times 2\pi$

- Find the equivalent prewarped continuous-time critical frequencies using $2\pi f_k = \Omega_k = (2/T)\tan(\omega_k/2)$

$$\begin{aligned} \Omega_{s1} &= 125.66 \text{ rad/s}, \quad \Omega_{s2} = 341.63 \text{ krad/s} \\ \Omega_{p1} &= 314.16 \text{ rad/s}, \quad \Omega_{p2} = 129.67 \text{ krad/s} \end{aligned}$$

- To determine the filter order N, we map the bandpass design back to the lowpass domain using

$$s \longleftarrow \frac{s^2 + \Omega_c^2}{s\Omega_b}$$

- In the lowpass domain we denote the two possible solutions for Ω_s as A, and B

$$A = \frac{\Omega_{p1}\Omega_{p2} - \Omega_{s1}^2}{\Omega_{s1}(\Omega_{p2} - \Omega_{p1})}, \quad B = \frac{\Omega_{s2}^2 - \Omega_{p1}\Omega_{p2}}{\Omega_{s1}(\Omega_{p2} - \Omega_{p1})}$$

- To insure that both upper and lower stopband attenuation specifications are met let

$$\Omega_s = \min\{A,\ B\}$$

From the specifications of the problem at hand

$$A = 2.5052, \quad \text{and} \quad B = 2.6401,$$

thus

$$\Omega_s = 2.5052$$

- Using the Chebyshev design formula for N

$$N = \lceil 2.335 \rceil = 3$$

- From the polynomial tables for 1 dB ripple,

$$H_c(s) = \frac{0.4913}{s^3 + 0.988s^2 + 1.238s + 0.491}$$

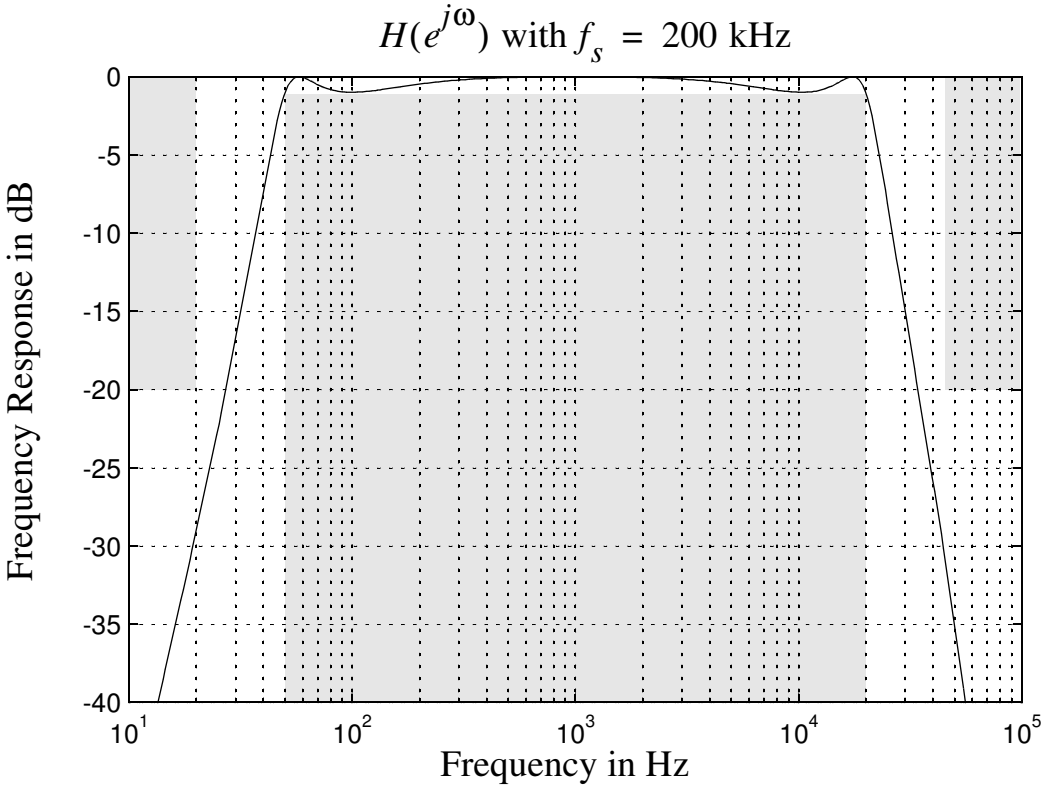
- To obtain the discrete-time design perform in sequence the following transformations:

$$s \rightarrow \frac{s^2 + \Omega_c^2}{s\,\Omega_b} \rightarrow \frac{2}{T}\frac{1 - z^{-1}}{1 + z^{-1}}$$

MATLAB Bilinear Filter Analysis

- We can use MATLAB to complete the analysis of the above filter design by computing the frequency response magnitude in dB

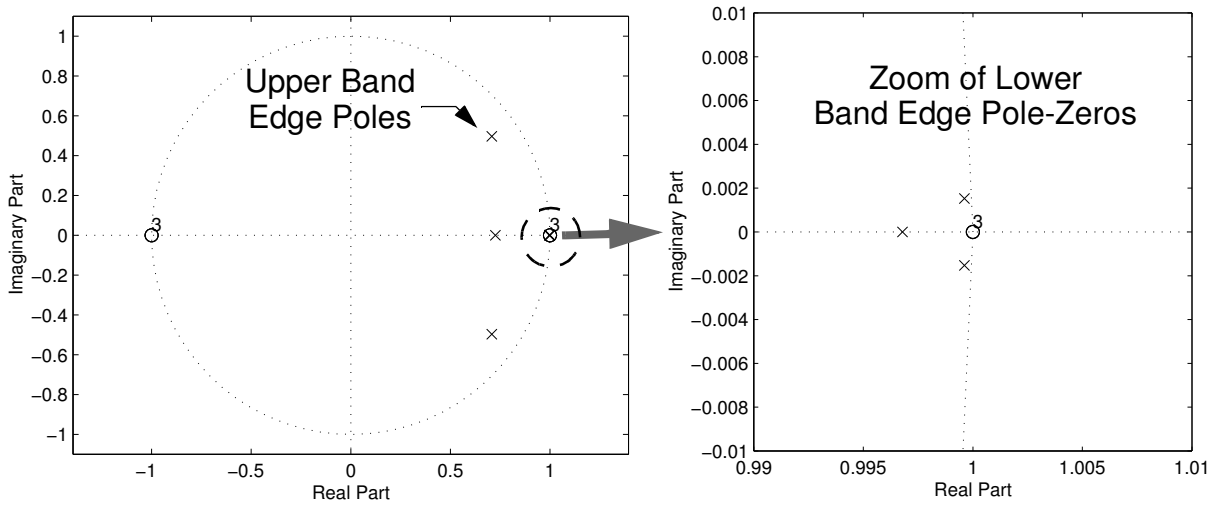
```
>> F = logspace(1,5,500);
>> f = F/200000;
>> % Inverse map z to s
>> s1 = (1-exp(-j*2*pi*f))./(1+exp(-j*2*pi*f));
>> % Inverse map bandpass s to lowpass s
>> s = ((2*200000*s1).^2 + wc^2)./(2*200000*s1*wb);
>> H = 0.4913./(s.^3 + 0.988*s.^2 + 1.238*s + 0.491);
>> semilogx(F,20*log10(abs(H)))
>> axis([10 100000 -40 2])
>> grid
```



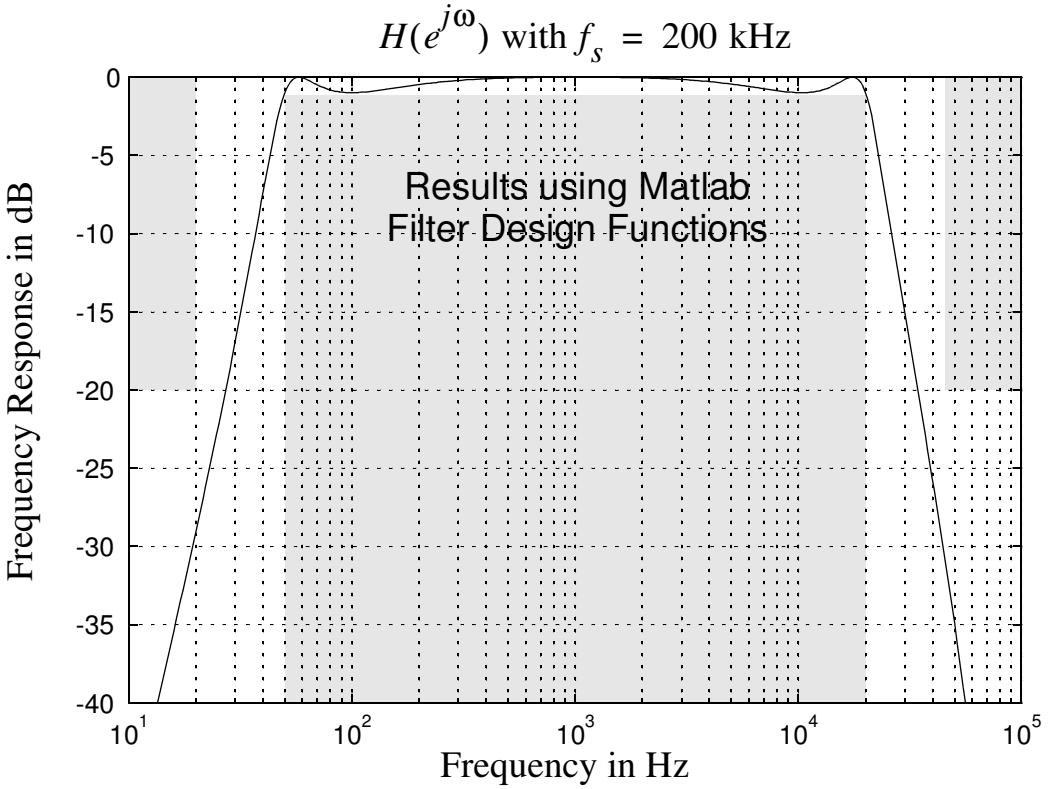
MATLAB Bilinear Filter Design

- In MATLAB a complete bilinear filter synthesis, for a specific analog prototype, can be done in one step
- We simply use the filter design functions, in this case since we desire a Chebyshev type I design we use `cheb1ord` and `cheby1`

```
>> % In a digital design the critical frequencies are
>> % entered as w/pi, i.e., they lie on the interval
>> % (0,1).
>> [N,wn] = cheb1ord([50/100000 20000/100000],...
    [20/100000 45000/100000],1,20)
N =     3
wn =    0.0005    0.2000
>> [b,a] = cheby1(N,1,wn)
b =   0.0114         0   -0.0342         0
      0.0342         0   -0.0114
a =   1.0000   -5.1378   11.1842  -13.2660
      9.0712   -3.3922    0.5406
>> zplane(b,a)
>> [H,F] = freqz(b,a,512,200000);
>> semilogx(F,20*log10(abs(H)))
>> grid
>> axis([10 100000 -40 0])
```



z-domain pole-zero plot from MATLAB design



Magnitude response in dB from MATLAB design

- The two set of results presented above appear to produce similar results
- The MATLAB design approach is simple to use and provides the complete filter solution, that is the difference equation coefficient vectors `a` and `b`

9.8 Discrete-Time Transformations

We desire a rational function $Z^{-1} = G(z^{-1})$ that can be used to produce

$$H(z) = H_{lp}(Z)\Big|_{Z^{-1}=G(z^{-1})}$$

where Z is associated with the lowpass prototype and z is associated with the transformed filter. Since $H_{lp}(Z)$ and $G(z^{-1})$ are both rational, we must have:

1. $G(z^{-1})$ map the interior of the Z-plane into the interior of the z-plane

2. $G(z^{-1})$ map the unit circle of the Z-plane onto the unit circle of the z-plane

A fundamental property of $G(z^{-1})$ can be observed by letting $Z = e^{j\theta}$ and $z = e^{j\omega}$, then

$$e^{-j\theta} \overset{\text{must}}{=} |G(e^{-j\omega})|e^{j\angle G(e^{-j\omega})}$$

hence

$$|G(e^{-j\omega})| = 1 \;\Rightarrow\; G(z^{-1}) \text{ is allpass}$$

- The commonly referenced transformations published by Constantinides[4] are of the general form

$$Z^{-1} = G(z^{-1}) = \pm\prod_{k=1}^{N}\frac{z^{-1}-\alpha_k}{1-\alpha_k z^{-1}}$$

[4] A.G. Constantinides, "Spectral Transformations for Digital Filters," *Proc. IEE*, Vol. 117, No. 8, pp. 1585-1590, Aug. 1970

Transformations From a Digital Lowpass with Cutoff Frequency θ_p

Filter Type	Transformation	Associated Design Formulas
Filter Type	$Z^{-1} = \frac{z^{-1}-\alpha}{1-\alpha z^{-1}}$	$\alpha = \dfrac{\sin\left(\frac{\theta_p-\omega_p}{2}\right)}{\sin\left(\frac{\theta_p+\omega_p}{2}\right)}$ $\omega_p =$ desired cutoff frequency
Highpass	$Z^{-1} = -\frac{z^{-1}-\alpha}{1-\alpha z^{-1}}$	$\alpha = -\dfrac{\sin\left(\frac{\theta_p+\omega_p}{2}\right)}{\sin\left(\frac{\theta_p-\omega_p}{2}\right)}$ $\omega_p =$ desired cutoff frequency
Bandpass	$Z^{-1} = -\frac{z^{-2}-\frac{2\alpha k}{k+1}z^{-1}+\frac{k-1}{k+1}}{\frac{k-1}{k+1}z^{-2}-\frac{2\alpha k}{k+1}z^{-1}+1}$	$\alpha = \dfrac{\sin\left(\frac{\omega_{p2}+\omega_{p1}}{2}\right)}{\sin\left(\frac{\omega_{p2}-\omega_{p1}}{2}\right)}$ $k = \cot\left(\dfrac{\omega_{p2}-\omega_{p1}}{2}\right)\tan\left(\dfrac{\theta_p}{2}\right)$ $\omega_{p1} =$ desired lower cutoff frequency $\omega_{p2} =$ desired upper cutoff frequency
Bandstop	$Z^{-1} = -\frac{z^{-2}-\frac{2\alpha k}{k+1}z^{-1}+\frac{1-k}{1+k}}{\frac{1-k}{1+k}z^{-2}-\frac{2\alpha k}{k+1}z^{-1}+1}$	$\alpha = \dfrac{\sin\left(\frac{\omega_{p2}+\omega_{p1}}{2}\right)}{\sin\left(\frac{\omega_{p2}-\omega_{p1}}{2}\right)}$ $k = \cot\left(\dfrac{\omega_{p2}-\omega_{p1}}{2}\right)\tan\left(\dfrac{\theta_p}{2}\right)$ $\omega_{p1} =$ desired lower cutoff frequency $\omega_{p2} =$ desired upper cutoff frequency

- Note that the discrete-time (digital) transformations can be performed directly by substituting $G(z^{-1})$ for each Z^{-1}, or by calculating the new pole and zero locations by solving

$$Z_k^{-1} = G(z_k^{-1}), \quad P_k^{-1} = G(p_k^{-1})$$

The new poles and zeros are p_k and z_k respectively.

9.9 Design of Discrete-Time FIR Filters

By definition the impulse response is of finite duration. In contrast to the IIR design techniques, most FIR design techniques originate from the desire to approximate the amplitude response of a discrete-time system. Alternate names for FIR systems include:

- Nonrecursive filter
- Moving average filter
- Transversal filter
- Tapped delay line filter
- FIR advantages over IIR include:
  - Can be designed to have exactly linear phase
  - Since FIR filters are typically implemented with norecursive structures, they are inherently stable
  - Quantization effects can be minimized more easily
- FIR disadvantages over IIR:
  - A higher filter order is required to obtain the same amplitude response compared to a similar IIR design
  - The higher filter order also implies higher computational complexity
  - The higher order filter also implies greater memory requirements for storing coefficients

9.9.1 Design Using Windowing

Let the frequency response of the desired LTI system we wish to approximate be given by

$$H_d(e^{j\omega}) = \sum_{n=-\infty}^{\infty} h_d[n]e^{-j\omega n}$$

where $h_d[n]$ is the corresponding impulse response.

- Consider obtaining a causal FIR filter that approximates $h_d[n]$ by letting

$$h[n] = \begin{cases} h_d[n], & 0 \le n \le M, \\ 0, & \text{otherwise} \end{cases}$$

  The FIR filter then has frequency response

$$H(e^{j\omega}) = \sum_{n=0}^{M} h[n]e^{-j\omega n}$$

- Note that since we can write

$$h_d[n] = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\omega})e^{j\omega n}\,d\omega$$

  we are actually forming a finite Fourier series approximation to $H_d(e^{j\omega})$

- Since the ideal $H_d(e^{j\omega})$ may contain discontinuities at the band edges, truncation of the Fourier series will result in the Gibbs phenomenon

- To allow for a less abrupt Fourier series truncation and hence reduced Gibbs phenomenon oscillations, we may generalize $h[n]$ by writing

$$h[n] = h_d[n]w[n]$$

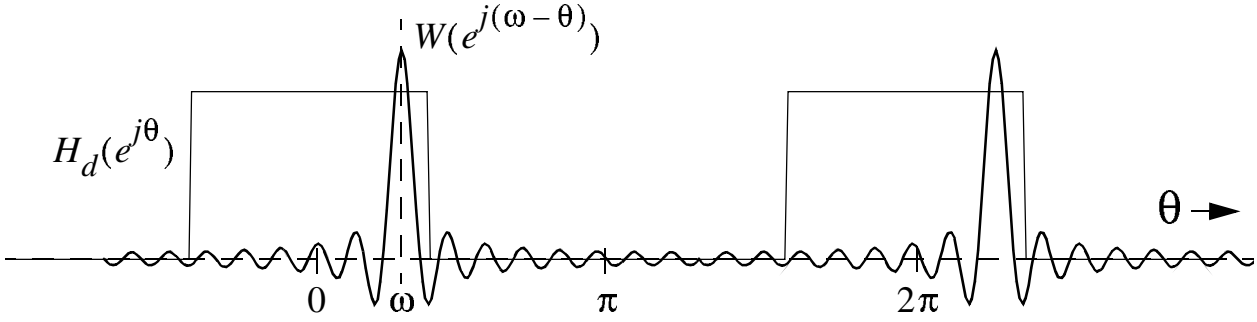
where $w[n]$ is a finite duration window function of length $M + 1$

Special Case: Rectangular Window

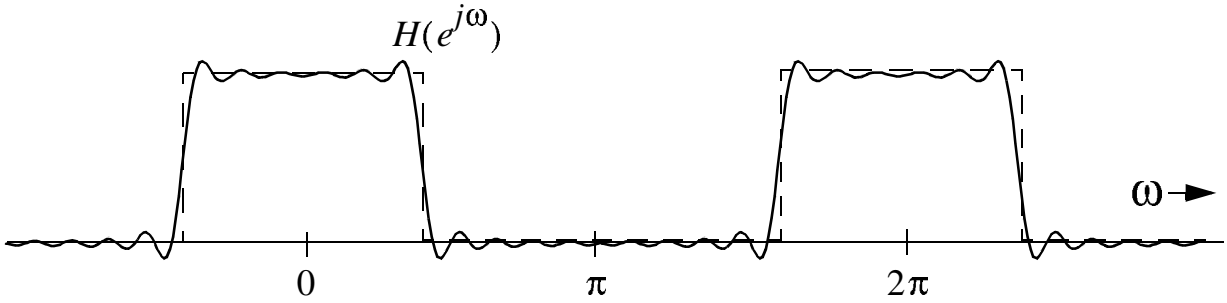
For this case $w[n]$ is simply unity over the window duration and due to the modulation theorem we can write

$$H(e^{j\omega}) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(e^{j\theta})W(e^{j(\omega-\theta)})\,d\theta$$

- Suppose $H_d(e^{j\omega})$ is an ideal lowpass filter, then $H(e^{j\omega})$ is a smeared version of the ideal response as shown on the following page



(a) Circular convolution (windowing)



(b) Resulting frequency response of windowed sinc(x)

(a) Convolution of $H_d(e^{j\omega})$ with $W(e^{j\omega})$, (b) Resulting $H(e^{j\omega})$

- Use of the rectangular window results in peak overshoot of 9% in both the passband and stopband. Equivalently the peak passband ripple is 0.75 dB and the peak stopband ripple is down only 21 dB.

- To reduce the overshoot a tapered window such as those used in spectral analysis may be used. The tradeoff for reduced overshoot will be a wider transition band at the discontinuity due to $W(e^{j\omega})$ having a wider main lobe.

Imposing a Linear Phase Constraint

In practice we would like $h[n]$ to be causal and have generalized linear phase.

- As a first step in achieving this goal note that all the window functions discussed earlier are symmetrical about $M/2$, i.e.

$$w[n] = \begin{cases} w[M-n], & 0 \le n \le M \\ 0, & \text{otherwise} \end{cases}$$

- If the desired impulse response is either symmetric or antisymmetric about $M/2$ then the resulting frequency response will be

$$H(e^{j\omega}) = A_e(e^{j\omega})e^{-j\omega M/2}, \quad A_e(e^{j\omega}) \text{ real and even fctn. of } \omega$$

or

$$H(e^{j\omega}) = jA_o(e^{j\omega})e^{-j\omega M/2}, \quad A_o(e^{j\omega}) \text{ real and odd fctn. of } \omega$$

respectively

- Note that the symmetric case leads to type I or II generalized linear phase, while the antisymmetric case leads to type III or IV generalized linear phase

- For $h[n]$ symmetric about $M/2$ we can write magnitude response, $A_e(e^{j\omega})$, as

$$A_e(e^{j\omega}) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_e(e^{j\theta})W_e(e^{j(\omega-\theta)})\,d\theta$$

where $H_e(e^{j\omega})$ and $W_e(e^{j\omega})$ are real functions corresponding to the magnitude responses of the desired filter frequency response and window function frequency response respectively

9.9.2 Lowpass Filter Design

Design an FIR lowpass filter using

$$H_d(e^{j\omega}) = \begin{cases} e^{-j\omega M/2}, & |\omega| < \omega_c \\ 0, & \text{otherwise} \end{cases}$$

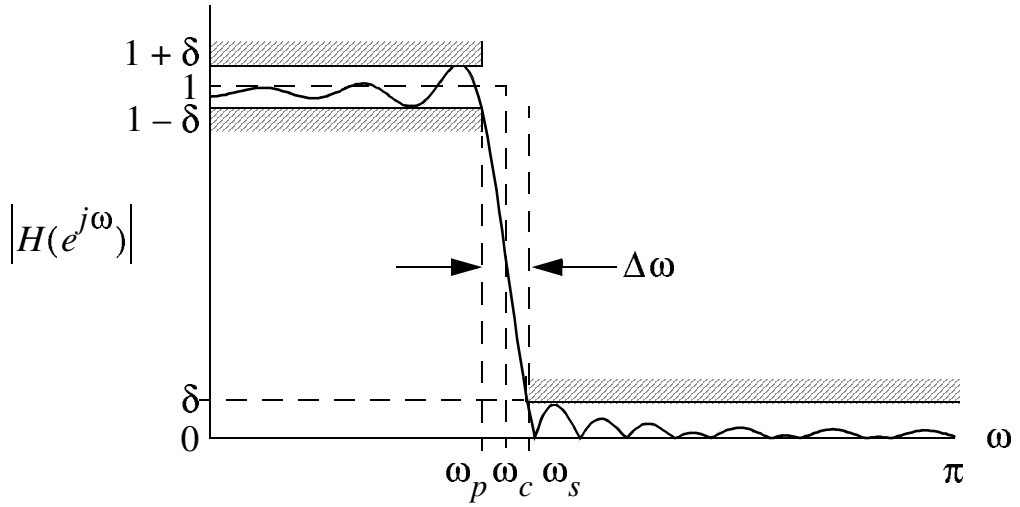
- Inverse Fourier transforming we find that

$$h_d[n] = \frac{\sin[\omega_c(n - M/2)]}{\pi(n - M/2)}$$

- Assuming $w[n]$ is symmetric about $M/2$, then the linear phase $h[n]$ is

$$h[n] = \frac{\sin[\omega_c(n - M/2)]}{\pi(n - M/2)} w[n]$$

- The relevant lowpass amplitude specifications of interest are shown below



- Note that the stopband attenuation in dB is $A_s = -20\log_{10}\delta$, and the paek ripple in dB is $\epsilon_{dB} = 20\log_{10}(1 + \delta)$

- For the Rectangular, Bartlett, Hanning, Hamming, and Blackman window functions the relevant design data is given in the following table:

Window Characteristics for FIR Filter Design			
Window Type	Transition Bandwidth $\Delta\omega$	Minimum Stopband Atten., A_s	Equivalent Kaiser Window β
Rectangular	$1.81\pi/M$	21 dB	0
Bartlett	$1.80\pi/M$	25 dB	1.33
Hanning	$5.01\pi/M$	44 dB	3.86
Hamming	$6.27\pi/M$	53 dB	4.86
Blackman	$9.19\pi/M$	74 dB	7.04

General Design Steps:

1. Choose the window function, $w[n]$, that just meets the stopband requirements as given in the table above.

2. Choose the filter length, M, (actual length is $M+1$) such that

$$\Delta\omega \leq \omega_s - \omega_p$$

3. Choose ω_c in the truncated impulse response such that

$$\omega_c = \frac{\omega_p + \omega_s}{2}$$

4. Plot $|H(e^{j\omega})|$ to see if the specifications are satisfied.

5. Adjust ω_c and M if necessary to meet the requirements. If possible reduce M.

   Note: This is a "Trial and Error" technique unless one chooses to use a Kaiser window (see below).

Kaiser Window Method:

1. Let $w[n]$ be a Kaiser window i.e.

$$w[n] = \begin{cases} \frac{I_0[\beta(1-[(n-\alpha)/\alpha]^2)^{1/2}]}{I_0(\beta)}, & 0 \le n \le M, \\ 0, & \text{otherwise} \end{cases}$$

   where $\alpha = M/2$.

2. Choose β for the specified A_s as

$$\beta = \begin{cases} .1102(A_s - 8.7), & A_s > 50 \\ .5842(A_s - 21)^{0.4} + .07886(A_s - 21), & 21 \le A_s \le 50 \\ 0.0, & A_s < 21 \end{cases}$$

3. The window length M is then chosen to satisfy

$$M = \frac{A_s - 8}{2.285\Delta\omega}$$

4. The value for ω_c is chosen as before.

   Note: Using the Kaiser empirical formulas M can be determined over a wide range of $\Delta\omega$ and A_s values to within ± 2. Very little if any iteration is needed.

Example 9.13: FIR Lowpass

Design an FIR lowpass using the windowing method such that $\omega_p = 0.4\pi$, $\omega_s = 0.6\pi$, and $\delta = 0.0032$ ($A_s = 50$ dB).

- From the window characteristics table we immediately see that for $A_s = 50$ a Hamming window will work

- To find M set

$$\Delta\omega = 0.1\pi \geq \frac{6.27\pi}{M}$$

or

$$M \geq \lceil 31.35 \rceil = 32$$

- The cutoff frequency is

$$\omega_c = \frac{.2 + .3}{2} \times 2\pi = 0.25 \times 2\pi$$

- If a Kaiser window is desired, then for β choose

$$\beta = 0.5842(50 - 21)^{0.4} + 0.07886(50 - 21) = 4.5335$$

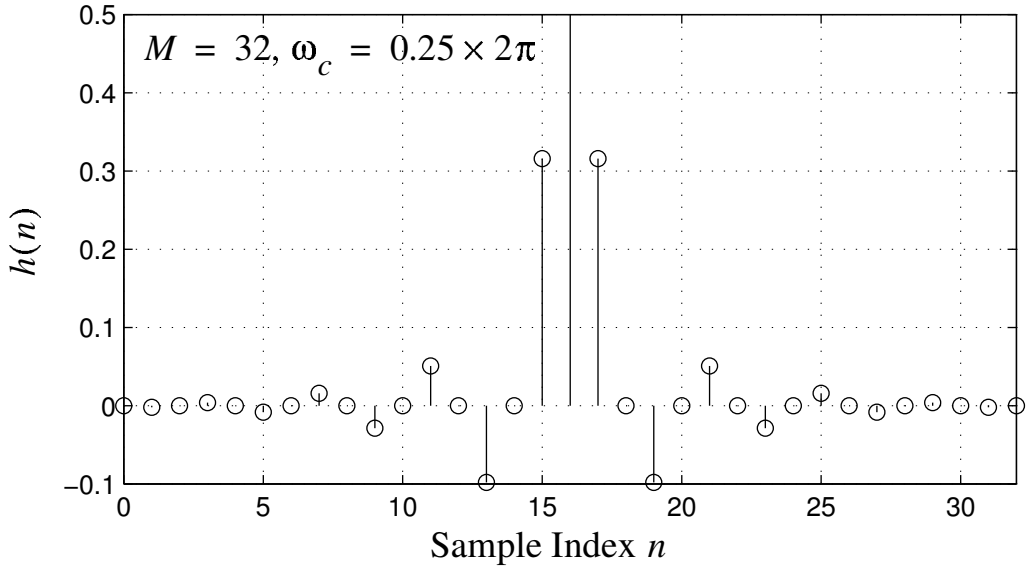
- The prescribed value for M should be

$$M = \left\lceil \frac{50 - 8}{2.285(0.2\pi)} \right\rceil = \lceil 29.25 \rceil = 30$$

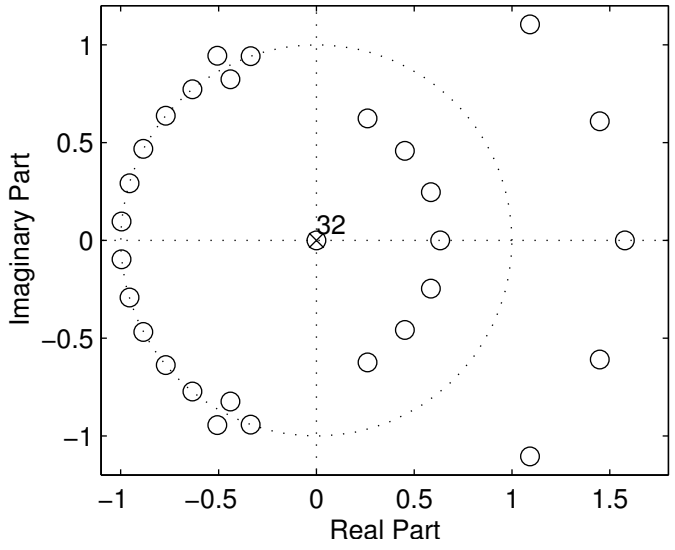
- Verification of these designs can be accomplished using the MATLAB filter design function `fir1()`

```
>> % Hamming window design
>> % (note fir1 uses hamming  by default)
>> b = fir1(32,2*.25,hamming(32+1));
>> stem(0:length(b)-1,b)
>> grid
>> zplane(b,1)
>> [H,F] = freqz(b,1,512,1);
>> plot(F,20*log10(abs(H)))
>> axis([0 .5 -70 2])
>> grid
>> % Kasier window design
>> bk = fir1(30,2*.25,kaiser(30+1,4.5335));
>> [Hk,F] = freqz(bk,1,512,1);
```

```
>> plot(F,20*log10(abs(Hk)))
>> axis([0 .5 -70 2])
>> grid
```



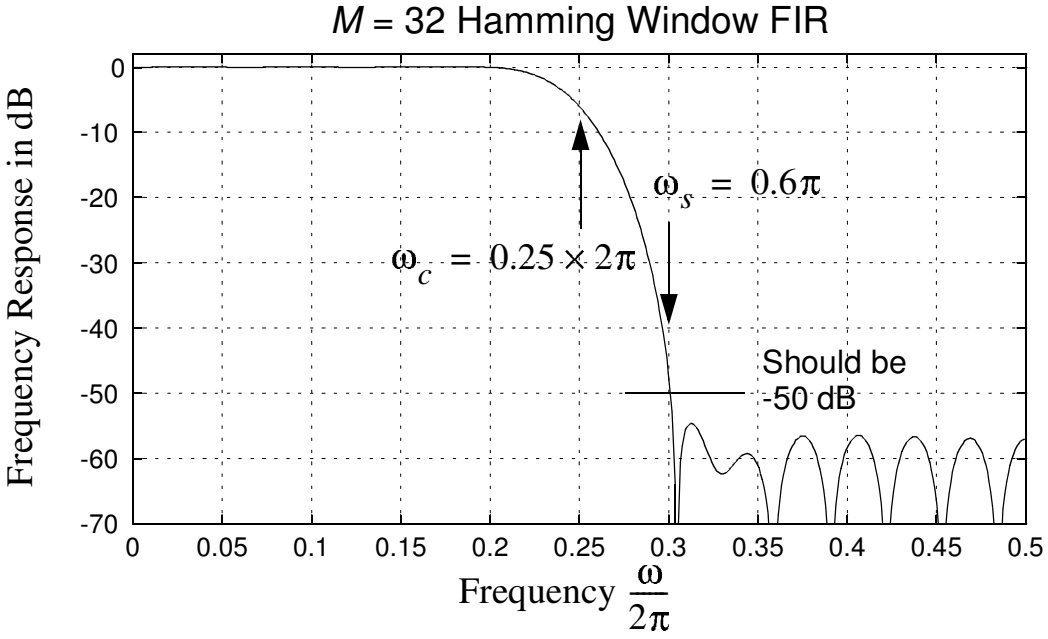
$M = 32$, Hamming window, impulse response



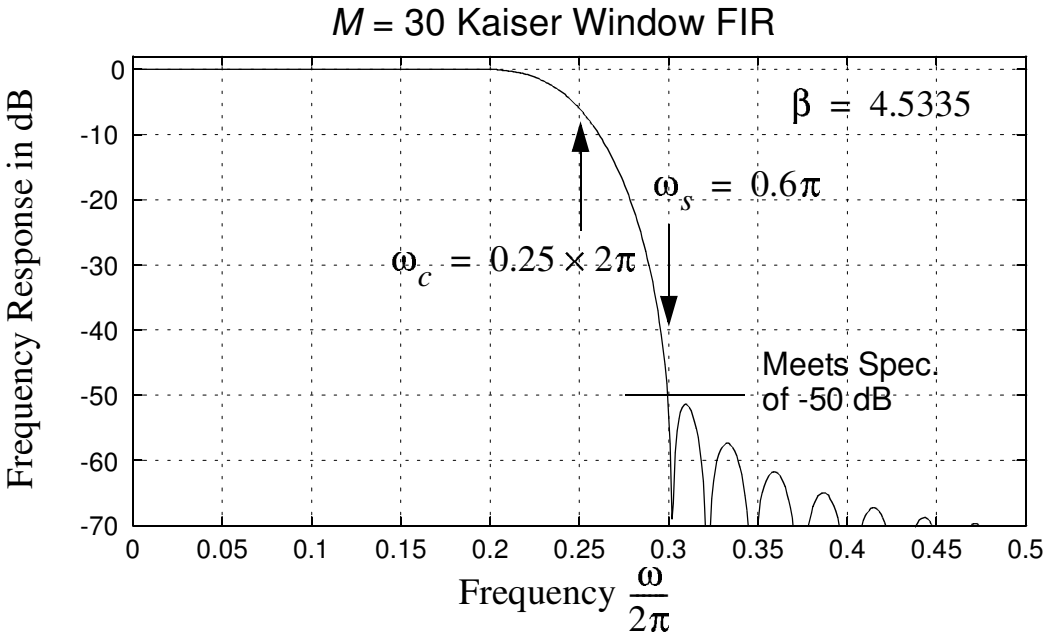
Note: There are few misplaced zeros in this plot due to MATLAB's problem finding the roots of a large polynomial.

Note: The many zero quadruplets that appear in this linear phase design.

$M = 32$, Hamming window, pole-zero plot



$M = 32$, Hamming window, frequency response



$M = 30$, Kaiser window ($\beta = 4.5335$), frequency response

9.10 Appendix: MATLAB s-Domain and z-Domain Filter Design Functions

9.10.1 Introduction

The intent of this appendix is to introduce some of the MATLAB functions that are useful for filter design. These functions are contained in the MATLAB signal processing toolbox. Many of them are also contained in the signals and systems toolbox of the old Prentice-Hall student edition of MATLAB.

9.10.2 Some Functions

The following function list is a subset of the functions contained in the MATLAB signal processing toolbox version 4.3, June 1999. The functions included here are those of most relevance to the current topics in the lecture notes. The function groupings match those of the toolbox manual.

Filter Analysis/Implementation	
`y = filter(b,a,x)`	Direct form II filter vector `x`
`[H,w] = freqs(b,a)`	s-domain frequency response computation
`[H,w] = freqz(b,a)`	z-domain frequency response computation
`[Gpd,w] = grpdelay(b,a)`	Group delay computation
`h = impz(b,a)`	Impulse response computation
`unwrap()`	Phase unwrapping
`zplane(b,a)`	Plotting of the z-plane pole/zero map

Linear System Transformations	
`residuez()`	z-domain partial fraction conversion
`tf2zp()`	Transfer function to zero-pole conversion
`zp2sos()`	Zero-pole to second-order biquadratic sections conversion

IIR Filter Design	
`[b,a] = besself(n,Wn)`	Bessel analog filter design. Near constant group delay filters, but if transformed to a digital filter this property is lost
`[b,a] = butter (n,Wn,'ftype','s')`	Butterworth analog and digital filter designs. Use 's' for s-domain design. 'ftype' is chooses bandpass, highpass or bandstop designs.
`[b,a] = cheby1 (n,Rp,Wn,'ftype','s')`	Chebyshev type I analog and digital filter designs. Use 's' for s-domain design. 'ftype' chooses bandpass, highpass or bandstop designs.
`[b,a] = cheby2 (n,Rs,Wn,'ftype','s')`	Chebyshev type II analog and digital filter designs. Use 's' for s-domain design. 'ftype' chooses bandpass, highpass or bandstop designs.
`[b,a] = ellip(n,Rp,Rs,Wn, 'ftype','s')`	Elliptic analog and digital filter designs. Use 's' for s-domain design. 'ftype' chooses bandpass, highpass or bandstop designs.

IIR Filter Order Selection	
`[n,Wn] = buttord (Wp,Ws,Rp,Rs,'s')`	Butterworth analog and digital filter order selection. Use 's' for *s*-domain designs.
`[n,Wn] = cheb1ord (Wp,Ws,Rp,Rs,'s')`	Chebyshev type I analog and digital filter order selection. Use 's' for *s*-domain designs.
`[n,Wn] = cheb2ord (Wp,Ws,Rp,Rs,'s')`	Chebyshev type II analog and digital filter order selection. Use 's' for *s*-domain designs.
`[n,Wn] = ellipord (Wp,Ws,Rp,Rs,'s')`	Elliptic analog and digital filter order selection. Use ŚsŠ for *s*-domain designs.

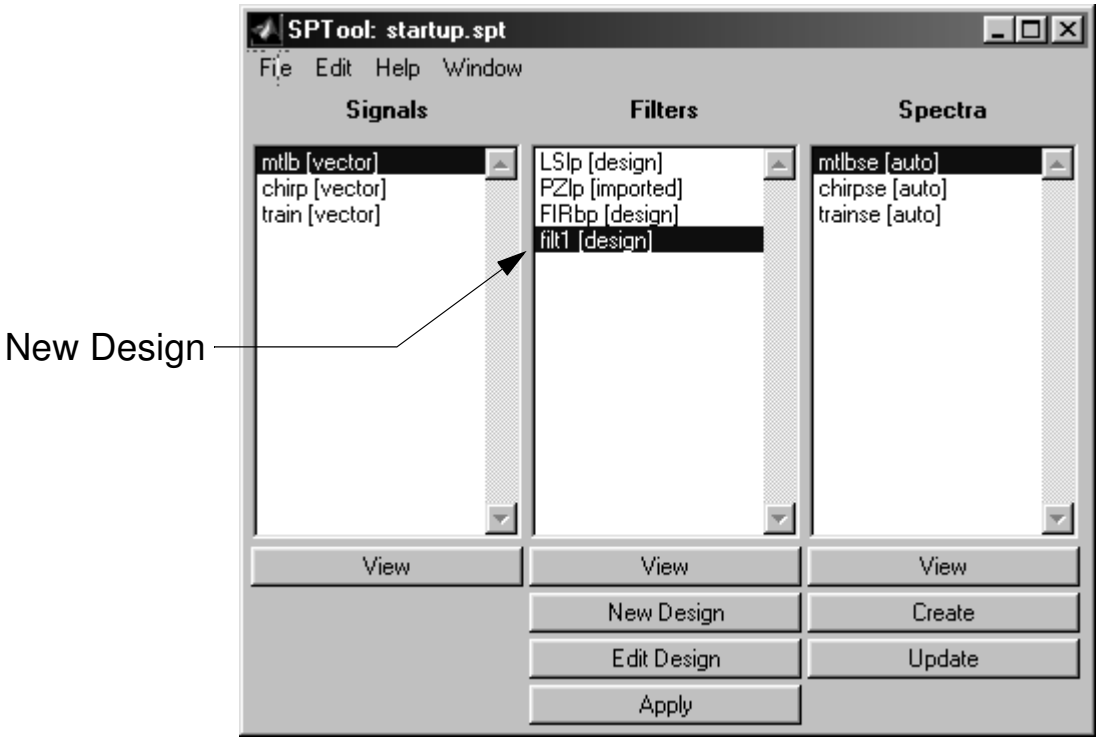
FIR Filter Design	
`fir1()`	Window-based FIR filter design with standard response.
`fir2()`	Window based FIR filter design with arbitrary response.
`remez()`	Parks-McClellan optimal FIR filter design
`remezord()`	Parks-McClellan filter order estimation.

Filter Discretization	
`[bz,az] = bilinear (bs,as,Fs,Fp)`	Maps s-domain transfer function to z-domain using the bilinear transformation. `Fs` is sampling rate in Hz. The optional parameter `Fp`, in Hz, is a matching frequency.
`[bz,az] = impinvar (bs,as,Fs)`	Maps s-domain transfer function to z-domain using impulse invariance. `Fs` is sampling rate in Hz.

9.11 Appendix: Using MATLAB `sptool` for Filter Design

The MATLAB signal processing toolbox contains a graphical user interface (GUI) application known as `sptool`, short for Signal Processing Tool. With the `sptool` environment there are windows for analyzing and manipulating signals, designing digital filters, and performing spectral analysis. Signals may be imported and exported from `sptool` and filter designs can be exported in a data structure format which will be described below. In this appendix we are concerned primarily with the filter design capabilities of `sptool`.

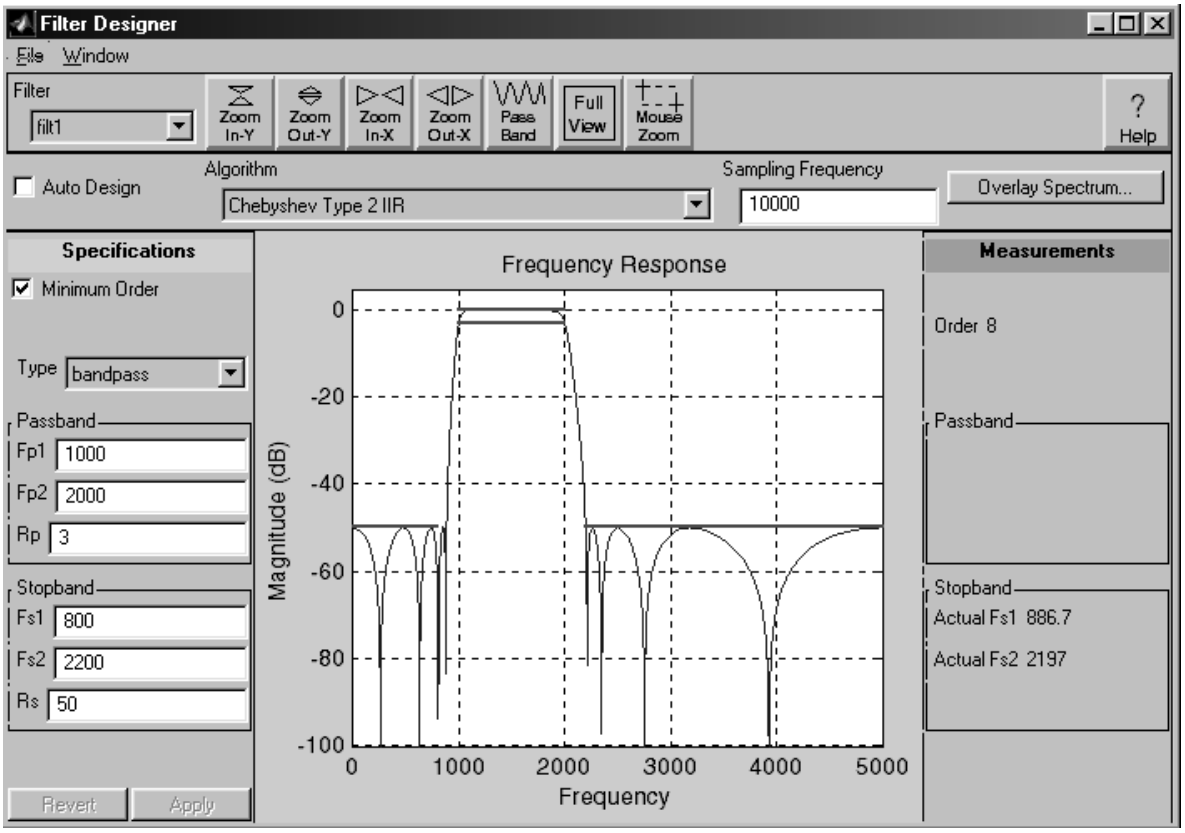
To start `sptool` simply type `sptool` at the MATLAB prompt. The window that opens is the main window which shows the current signals, filters, and spectra loaded into the `sptool` environment.



SPTool startup dialog showing predefined signal, filter, and spectra

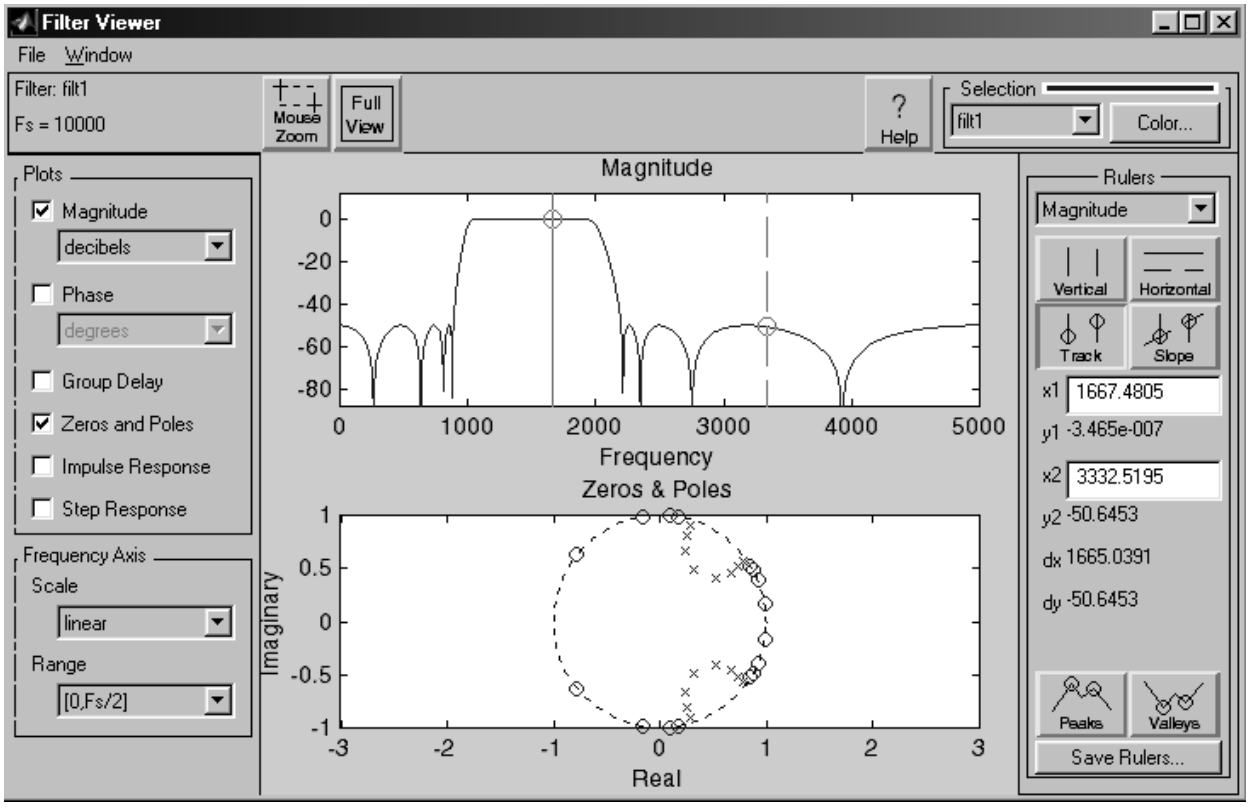
9.11.1 A Chebyshev Type II Bandpass Design

Starting with a sampling rate of $F_s = 10,000$ Hz we set out to design a bandpass filter with the specifications. By choosing the button New Design under the filters heading in the `SPTool` main dialog, we obtain the Filter Designer dialog as shown below.



The filter design dialog configured for a type II Chebyshev bandpass design

Once a design has been created we can return to the main dialog and choose View to open the Filter Viewer dialog as shown in the following figure.



The filter viewer dialog configured to show the magnitude response in dB and the pole-zero plot

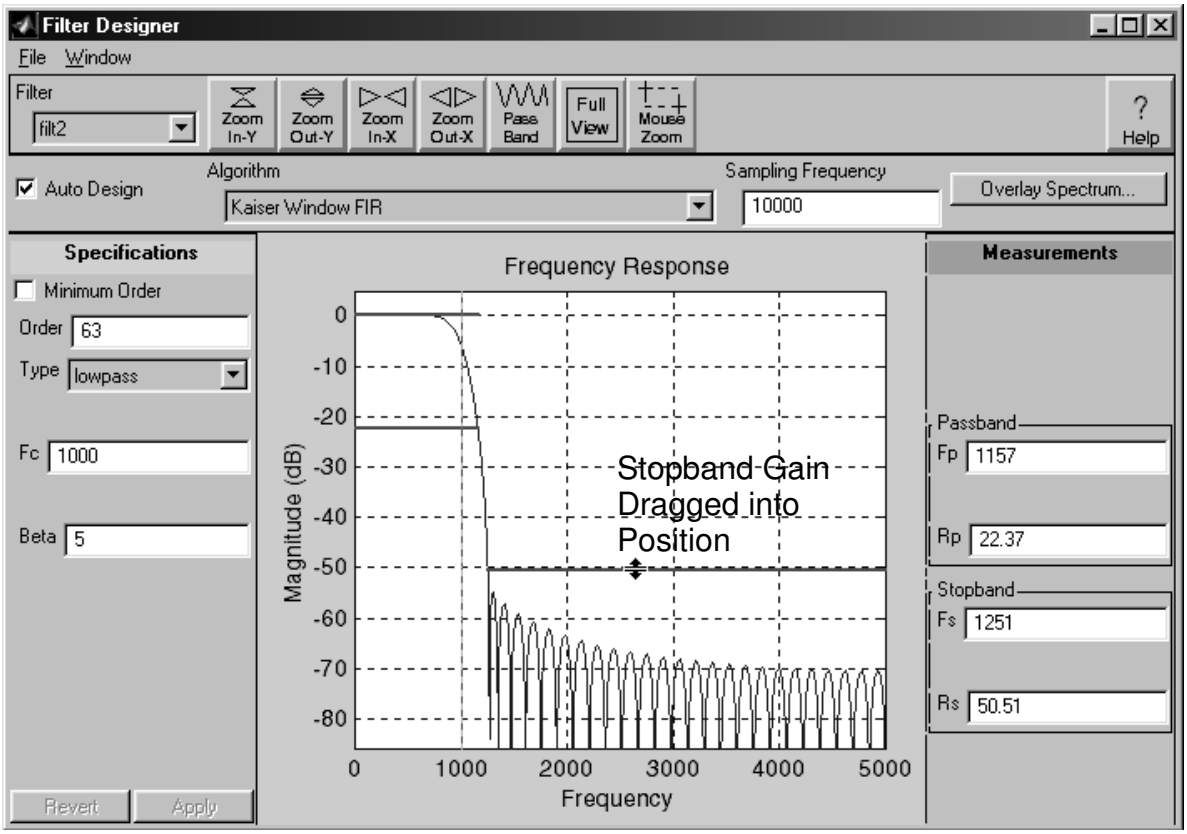
- The filter design we have just created is stored locally to `sptool` in the data structure `filt1`
- The filter data structure can be exported to the MATLAB workspace by using the Export option under the menu File in the `sptool` main dialog window
- The expansion of the data structure contents is given in the following listing from the command window of MATLAB
- Note that not all of the elements are filled, simply because windows displaying that particular plot type were never requested when using the Filter Viewer

```
>> % Data structure passed out of sptool
>> filt1

filt1 =

            tf: [1x1 struct]
            ss: []
            zpk: [1x1 struct]
            sos: []
            imp: [331x1 double]
            step: []
                t: [331x1 double]
                H: [1x2048 double]
                G: [1x2048 double]
                f: [1x2048 double]
            specs: [1x1 struct]
            Fs: 10000
            type: 'design'
        lineinfo: [1x1 struct]
    SPTIdentifier: [1x1 struct]
            label: 'filt1'

>> % Getting access to the transfer function element
>> filt1.tf

ans =

    num: [1x17 double]
    den: [1x17 double]

>> % Getting access to the numerator coefficients
>> filt1.tf.num

ans =

Columns 1 through 6

7.3772e-003 -4.3836e-002  1.3043e-001 -2.6554e-001  4.1999e-001 -5.4723e-001

Columns 7 through 12

6.1629e-001 -6.3818e-001  6.4144e-001 -6.3818e-001  6.1629e-001 -5.4723e-001

Columns 13 through 17

4.1999e-001 -2.6554e-001  1.3043e-001 -4.3836e-002  7.3772e-003
```

```
>> % Getting access to the denominator coefficients
>> filt1.tf.den

ans =

Columns 1 through 6

1.0000e+000 -7.6586e+000  3.0264e+001 -8.0426e+001  1.5955e+002 -2.4910e+002

Columns 7 through 12

3.1538e+002 -3.2944e+002  2.8649e+002 -2.0795e+002  1.2560e+002 -6.2522e+001

Columns 13 through 17

2.5204e+001 -7.9830e+000  1.8856e+000 -2.9976e-001  2.4771e-002
```

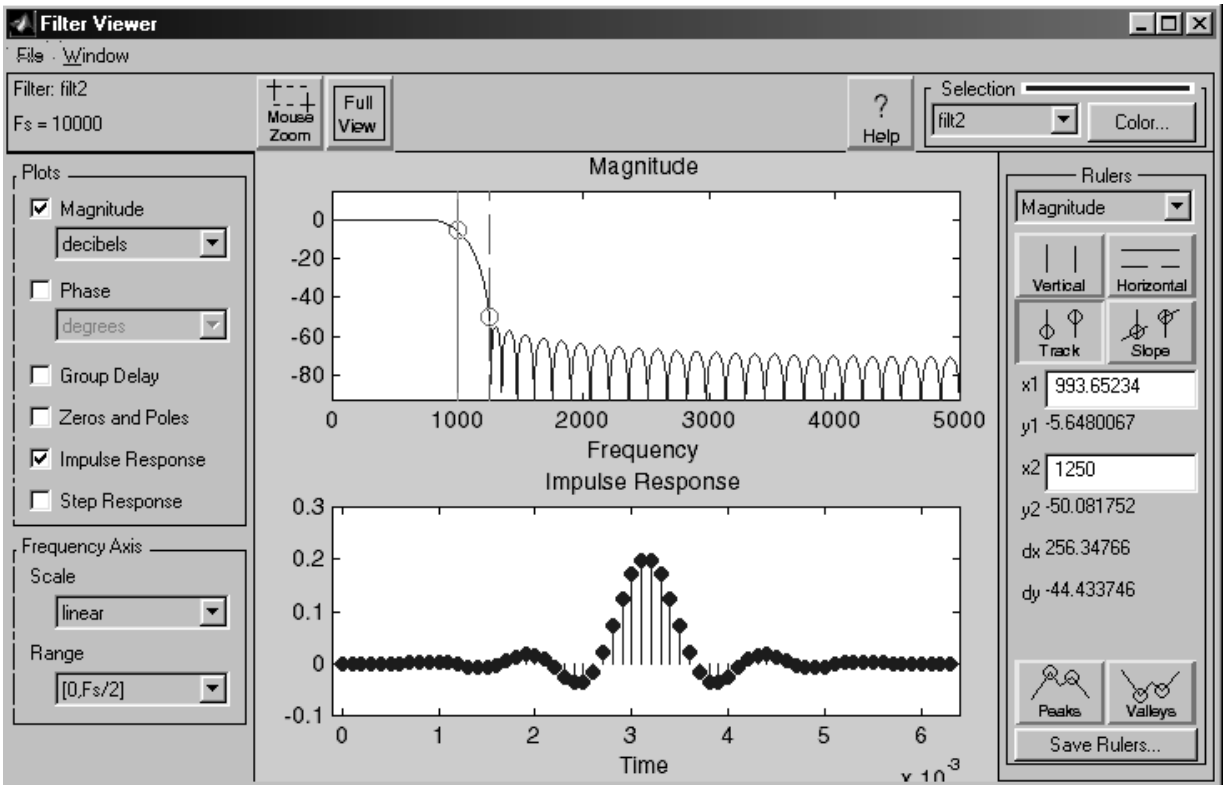
9.11.2 A Windowed FIR Design

Again starting with a sampling rate of $F_s = 10,000$ Hz we set out to design an FIR lowpass filter using a Kaiser window. The design tool allows all of the parameters to be varied by filling in quantities in the dialog box as well as dragging the filter gain and critical frequencies around.



The filter designer configured for a Kaiser window lowpass design

Uisng the Filter Viewer we now plot the magnitude response in dB and the impulse response.



Filter characteristics displayed in the Filter Viewer

```
>> % Data structure passed out of sptool
>> filt2.tf.num

ans =

Columns 1 through 6

3.0012e-004  1.6934e-004 -2.3616e-004 -8.2882e-004 -1.3339e-003 -1.3738e-003

Columns 7 through 12

-6.5604e-004  8.0810e-004  2.5737e-003  3.8295e-003  3.6959e-003  1.6710e-003

Columns 13 through 18

-1.9648e-003 -6.0141e-003 -8.6494e-003 -8.1097e-003 -3.5787e-003  4.1252e-003

Columns 19 through 24

1.2432e-002  1.7683e-002  1.6476e-002  7.2632e-003 -8.4152e-003 -2.5681e-002
```

```
Columns 25 through 30

-3.7334e-002 -3.5992e-002 -1.6699e-002  2.0880e-002  7.1570e-002  1.2554e-001

Columns 31 through 36

1.7081e-001  1.9662e-001  1.9662e-001  1.7081e-001  1.2554e-001  7.1570e-002

Columns 37 through 42

2.0880e-002 -1.6699e-002 -3.5992e-002 -3.7334e-002 -2.5681e-002 -8.4152e-003

Columns 43 through 48

7.2632e-003  1.6476e-002  1.7683e-002  1.2432e-002  4.1252e-003 -3.5787e-003

Columns 49 through 54

-8.1097e-003 -8.6494e-003 -6.0141e-003 -1.9648e-003  1.6710e-003  3.6959e-003

Columns 55 through 60

3.8295e-003  2.5737e-003  8.0810e-004 -6.5604e-004 -1.3738e-003 -1.3339e-003

Columns 61 through 64

-8.2882e-004 -2.3616e-004  1.6934e-004  3.0012e-004
```