Infinite Impulse Response (IIR) Filter Design

E

Objectives

1. To design Butterworth IIR filters using the Bilinear Transformation.
2. To design Chebyshev (Type – I and Type – II) IIR filters using the Bilinear Transformation.
3. To compare the performance of Butterworth and Chebyshev filters based on frequency response.

Apparatus

Software: MATLAB R2022b
Hardware: Personal Computer

Theory

Infinite Impulse Response (IIR) filters are characterized by a feedback mechanism, where the output depends not only on the current input but also on previous inputs and outputs. This feedback structure enables IIR filters to achieve a desired frequency response with lower filter orders compared to Finite Impulse Response (FIR) filters, making them computationally efficient for many applications. Unlike FIR filters, which have a finite-duration impulse response, IIR filters exhibit an impulse response that theoretically extends indefinitely due to their recursive nature. However, the infinite nature of the response can lead to stability challenges if the filter is not carefully designed.

The design of IIR filters typically begins with analog prototypes, such as Butterworth or Chebyshev filters. These analog filters are subsequently transformed into digital filters using techniques such as the bilinear transformation or impulse invariance. Among these methods, the bilinear transformation is particularly favored, as it effectively maps the analog frequency response to the digital domain while preserving essential characteristics, such as the overall shape of the filter's response.

Butterworth Filters

The Butterworth filter is known for its maximally flat response in the passband, ensuring no ripples and a smooth frequency response until it begins attenuating signals in the stopband. This smooth transition makes it ideal for applications that require a flat passband without any ripples or variations. The Butterworth filter features a gentle roll-off between the passband and stopband, with the roll-off becoming steeper as the filter order N increases.

Butterworth filters can be configured as low-pass, high-pass, band-pass, or band-stop filters. The magnitude responses for each configuration are summarized in Table 1.

Table 1. Magnitude responses of Butterworth Filters.

Configuration	Magnitude Response	Cutoff frequency
Low-pass	$\|\boldsymbol{H(j\omega)}\| = \frac{\mathbf{1}}{\sqrt{\mathbf{1} + (\boldsymbol{\omega/\omega_c})^{\mathbf{2N}}}}$	$\boldsymbol{\omega_c}$
High-pass	$\|\boldsymbol{H(j\omega)}\| = \frac{\mathbf{1}}{\sqrt{\mathbf{1} + (\boldsymbol{\omega_c/\omega})^{\mathbf{2N}}}}$	$\boldsymbol{\omega_c}$
Band-pass	$\|\boldsymbol{H(j\omega)}\| = \frac{\mathbf{1}}{\sqrt{\mathbf{1} + \prod_{k=1}^{2} (\boldsymbol{\omega/\omega_{ck}})^{\mathbf{2N_k}}}}$	$\boldsymbol{\omega_{c1}}$, $\boldsymbol{\omega_{c2}}$
Band-stop	$\|\boldsymbol{H(j\omega)}\| = \frac{\mathbf{1}}{\sqrt{\mathbf{1} + \prod_{k=1}^{2} (\boldsymbol{\omega_{ck}/\omega})^{\mathbf{2N_k}}}}$	$\boldsymbol{\omega_{c1}}$, $\boldsymbol{\omega_{c2}}$

Chebyshev Filter Filters

The Chebyshev filter offers a steeper roll-off than the Butterworth filter but introduces ripples in the frequency response. The degree of ripple depends on the filter type: Type-I or Type-II Chebyshev filters. Both filter types support low-pass, high-pass, band-pass, and band-stop configurations.

1. Type-I Chebyshev filter

The Type-I Chebyshev filter has ripples in the passband, while the stopband remains monotonic. It is useful for applications requiring a sharp transition from passband to stopband.

2. Type-II Chebyshev filter

The Type-II Chebyshev filter (also known as the inverse Chebyshev filter) exhibits no ripples in the passband but introduces ripples in the stopband. These filters are useful when the stopband requires more attenuation, while the passband must remain flat.

The magnitude responses for Type-I Chebyshev filters with Chebyshev polynomial of the N^{th} order (denoted as T_N) are summarized in Table 2. The magnitude responses for Type-II Chebyshev filters are similar to those of Type-I, but the ripples are shifted to the stopband. This swapping of ripples between the passband and stopband is reflected in the transformation of

terms such as ω/ω_c and ω_c/ω in low-pass and high-pass configurations, and similarly for band-pass and band-stop filters.

Table 2. Magnitude responses of Type-I Chebyshev filter.

Configuration	Magnitude Response	Cutoff frequency
Low-pass	$\|H(j\omega)\| = \frac{1}{\sqrt{1+\varepsilon^2 T_N^2\left(\frac{\omega}{\omega_c}\right)}}$	$\boldsymbol{\omega_c}$
High-pass	$\|H(j\omega)\| = \frac{1}{\sqrt{1+\varepsilon^2 T_N^2\left(\frac{\omega_c}{\omega}\right)}}$	$\boldsymbol{\omega_c}$
Band-pass	$\|H(j\omega)\| = \frac{1}{\sqrt{1+\varepsilon^2 T_N^2\left(\frac{\omega}{\omega_{c1}}\right).T_N^2\left(\frac{\omega_{c2}}{\omega}\right)}}$	$\boldsymbol{\omega_{c1}, \omega_{c2}}$
Band-stop	$\|H(j\omega)\| = \frac{1}{\sqrt{1+\varepsilon^2 T_N^2\left(\frac{\omega_{c1}}{\omega}\right).T_N^2\left(\frac{\omega}{\omega_{c2}}\right)}}$	$\boldsymbol{\omega_{c1}, \omega_{c2}}$

In both filter types, ε represents the ripple factor or ripple magnitude, which quantifies the variation in gain within the passband for Type-I filters or within the stopband for Type-II filters.

IIR Filter Design Using Bilinear Transformation (BLT)

The Bilinear Transformation is a widely used method for converting analog filter to a digital filter. The design procedure includes the following steps: (1) transforming digital filter specifications into analog filter specifications, (2) performing analog filter design, and (3) applying bilinear transformation and verifying the frequency response. The BLT is a mapping or transformation of analog frequency (points on the s-plane) to a digital frequency (the z-plane) represented as follows.

$$s = \frac{\mathbf{2}}{\boldsymbol{T}} \cdot \frac{\mathbf{1} - \boldsymbol{z^{-1}}}{\mathbf{1} + \boldsymbol{z^{-1}}}$$

Here z^{-1} represents the delay operator. The bilinear transformation preserves the shape of the filter's response but introduces a non-linear frequency distortion. This nonlinear frequency

mapping effect is called frequency warping. This can be corrected by pre-warping the critical frequencies. In practice, designing IIR filters often involves specifying parameters such as passband ripple, stopband attenuation, sampling frequency, and the cutoff frequencies. Meanwhile, the cutoff frequencies are the points where the magnitude response begins to transition between passband and stopband, typically defined at specific attenuation levels such as -3 db. These parameters guide the determination of the filter order and the exact cutoff frequencies that separate the passband and stopband regions.

MATLAB Code Overview

- Butterworth Filters

1. Filter specifications
   Defining filter parameters is a crucial step in the design of IIR filters. These parameters are essential to ensure the desired filter performance and characteristics.

   - MATLAB code for defining the design specification for Butterworth low-pass filter is,

```
alphap = 3;      % Passband ripple in dB
alphas = 15;     % Stopband attenuation in dB
fp = 500;        % Passband frequency
fs = 750;        % Stopband frequency
f = 2000;        % Sampling frequency
```

2. Normalized frequency
   The passband (fp) and stopband (fs) frequencies are normalized by dividing them by the Nyquist frequency (f/2). Here f is the sampling frequency. This is necessary because MATLAB functions such as '`buttord`' and '`butter`' operate on normalized frequencies in the range of 0 to 1 (corresponding to 0 to π in radians).

   - MATLAB code for determining the order and normalized cut off frequency is,

```
% Normalize the frequencies
omp = 2 * fp / f;  % Normalized passband frequency
oms = 2 * fs / f;  % Normalized stopband frequency
```

3. The filter order and cutoff frequency
   The '`buttord`' function is used to calculate the minimum order N of the Butterworth filter and the normalized cutoff frequency Wn, to meet the given design specifications.

- MATLAB code for determining and displaying the order and cutoff frequency is,

```
% Determine the order and cutoff frequency
[N, Wn] = buttord(omp, oms, alphap, alphas);

% Display the order and cutoff frequency
disp('Order of the filter n =');
disp(N);
disp('Cutoff frequency Wn = ');
disp(Wn);
```

4. Butterworth low-pass filter designing and displaying filter coefficients
   The 'butter' function designs the Butterworth filter using the order N and the normalized cutoff frequency Wn. It outputs the filter coefficients, '*b*' and '*a*', which correspond to the numerator and denominator of the filter's transfer function, respectively. These coefficients are used to express the digital filter's transfer function, *H(z)*, in its standard form as follows.

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} + \cdots}{a_0 + a_1 z^{-1} + a_2 z^{-2} + \cdots}$$

   In digital filters, the numerator coefficients control the zeros, determining which frequencies are attenuated. A low-pass filter typically has positive *b* values, while a negative *b* value indicates a high-pass filter. Meanwhile, the denominator coefficients control the poles, influencing the filter's stability and frequency response.

   - MATLAB code for designing the filter and displaying the filter coefficients is,

```
% Design Butterworth low-pass filter
[b, a] = butter(N, Wn, 'low');
% Display the filter coefficients
disp('Filter coefficients b =');
disp(b);
disp('Filter coefficients a =');
disp(a);
```

5. Frequency response
   The '`freqz`' function calculates the frequency response *h* of the filter over a range of frequencies *w*. This step can be used to plot the gain and phase responses of the filter.

   - MATLAB code for the determining and plotting the frequency response is,

```
w = 0:0.01:pi;  % Frequency ranges from 0 to π (normalized)
[h, om] = freqz(b, a, w, 'whole');% The frequency response
m = abs(h);     % Magnitude of the frequency response
an = angle(h);  % Phase of the frequency response
```

6. Bilinear Transformation
   This function converts the analog filter defined by *b* and *a* into a digital filter using the bilinear transformation method. As we discussed earlier, the bilinear transformation maps the s-plane to the z-plane, allowing the design of digital filters based on analog prototypes.

   • MATLAB code for Bilinear Transformation is,

```
[bz,az] = bilinear(b,a,f);
```

Procedure

1. Open MATLAB and create a new script by navigating to **Home → New → Script**
2. Begin your script by defining the filter specifications.
3. Normalize the passband and stopband frequencies for digital filter design.
4. Use the '`buttord`' function to determine the order (N) and cutoff frequency (Wn) of the filter.
5. Use the '`butter`' function to generate the filter coefficients.
6. Set the frequency range over which the filter's frequency response will be visualized.
7. Calculate the frequency response of the filter using '`freqz`' function.
8. Use the bilinear transformation to convert the analog filter to a digital filter.
9. Save your script with an appropriate name, "low_pass_butterworth_filter.m" and run the script to generate low-pass Butterworth filter responses as shown in Figure 1.
10. Save the figures by selecting **File → Save As** in the Figure window.
11. Design a high-pass Butterworth filter, open a new script, modify the code to use the butter function with the '`high`' option, and repeat the above steps to obtain the filter responses as shown in Figure 2.

Results

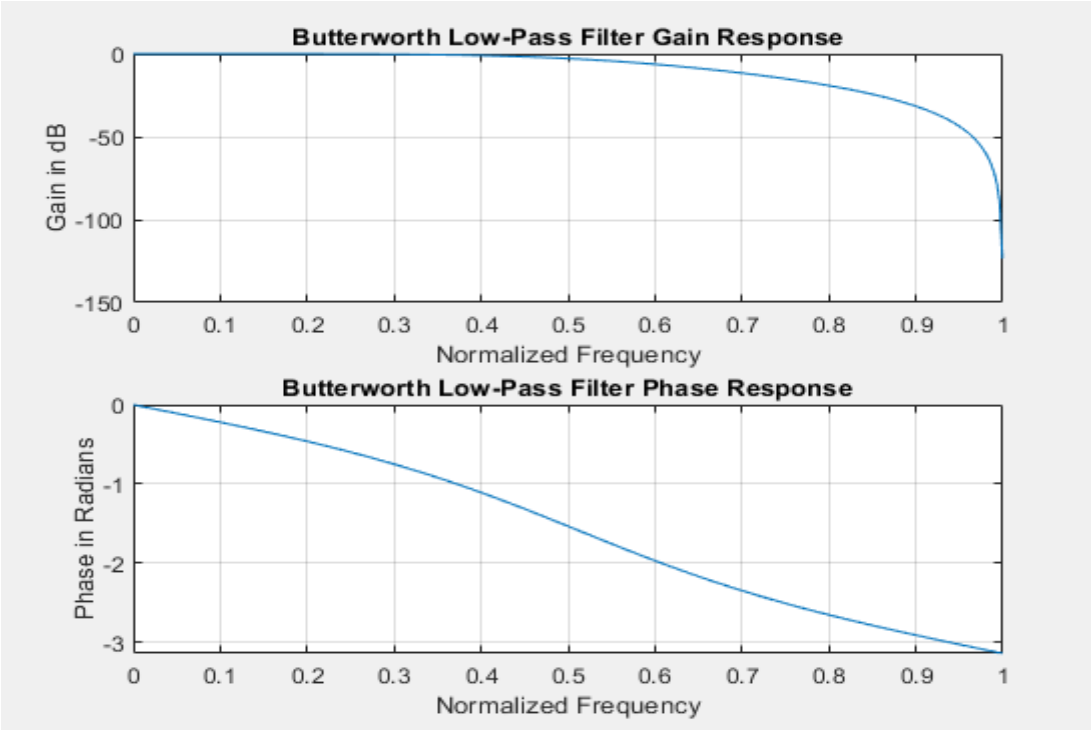

Figure 1.Butterworth Low-Pass Filter design.

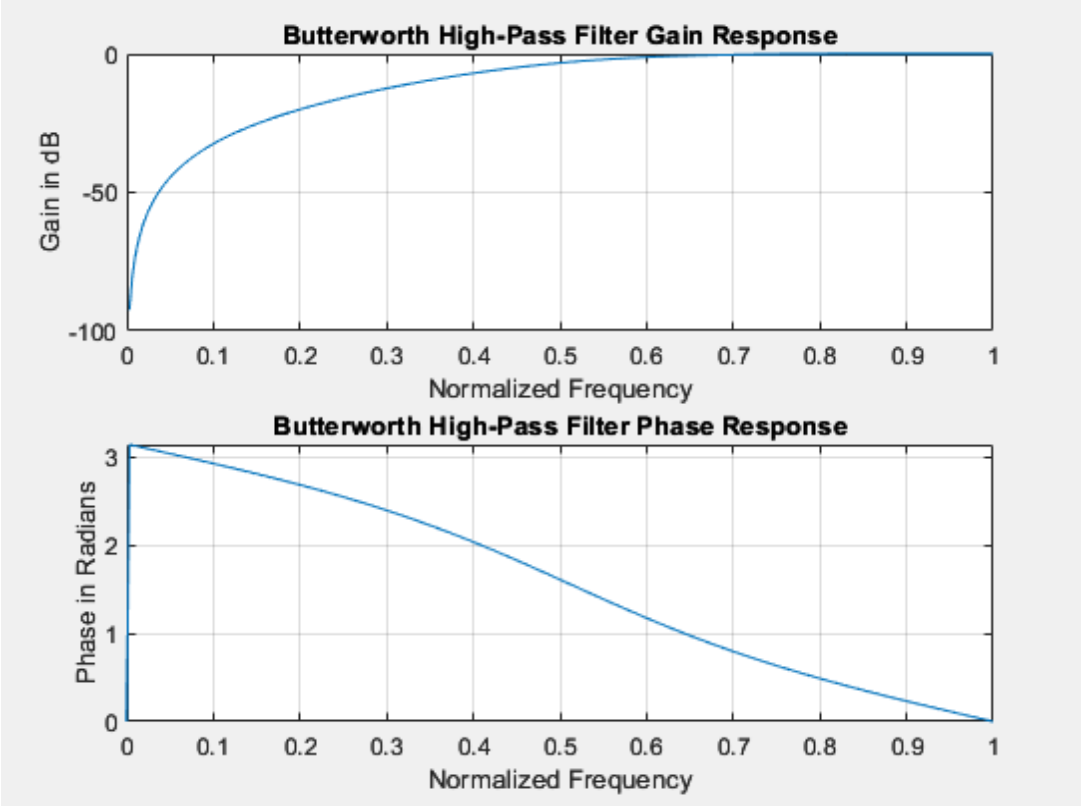

Figure 2. Butterworth High-Pass Filter design.

Exercise:

1. Design a low-pass Chebyshev Type I filter using MATLAB based on the following specifications:passband ripple (αp) of 2.5 dB, stopband attenuation (αs) of 40 dB, passband frequency (fp) of 500 Hz, stopband frequency (fs) of 900 Hz, and sampling frequency (f) of 3000 Hz.

   (a) Follow the same procedure as for the Butterworth filter design: to normalize the frequencies, determine the filter order (N) and cutoff frequency (Wn) using the cheb1ord function. Then, design the low-pass filter using the cheby1 function to calculate the filter coefficients.
   (b) Analyze the frequency response using the freqz function to plot the gain (magnitude) response in decibels and the phase response over the normalized frequency range. Display the filter order and cutoff frequency in the command window.
   (c) Design a high-pass Chebyshev Type I filter using the same specifications. For this, modify the design step by specifying 'high' in the cheby1 function.

2. Design a low-pass Chebyshev Type II filter using MATLAB based on the following specifications: stopband ripple (α_s) of 25 dB, passband attenuation (α_p) of 1.2 dB, passband frequency (f_p) of 600 Hz, stopband frequency (f_s) of 1000 Hz, and sampling frequency (f) of 5000 Hz

   (a) Follow the same procedure as for the Chebyshev Type I filter design to normalize the frequencies, determine the filter order (N) and cutoff frequency (W_n) using the cheb2ord function. Then, design the low-pass filter using the cheby2 function to calculate the filter coefficients.
   (b) Use the freqz function to analyze the frequency response by plotting the gain (magnitude) in decibels and phase response over the normalized frequency range. Use 'y' (yellow) for magnitude and 'b' (blue) for phase plots. Display the filter order and cutoff frequency in the command window.
   (c) Design a high-pass Chebyshev Type II filter with the same specifications by modifying the design step to specify 'high' in the cheby2 function. Use 'g' (green) for magnitude and 'r' (red) for phase plots.

3. Design a Butterworth band-stop filter using MATLAB to effectively eliminate interference at 150 Hz from a measurement signal sampled at 1200 Hz.

   (a) Use the butter function with stopband frequencies 148 Hz & 152 Hz, passband frequencies 140 Hz & 160 Hz, passband ripple (α_p)=1.5 dB, and stopband attenuation (α_s)=45 dB.
   (b) Plot the frequency response using freqz.Use yellow dash-dot lines ('y-.') for magnitude and blue dashed lines ('b--') for the dB plot.

4. Design a band-pass Butterworth filter for marine sonar signal analysis to isolate frequencies between 800 Hz and 1800 Hz.Frequencies below 500 Hz and above 2500 Hz should be attenuated with a stopband attenuation of 40 dB.The passband ripple is 1 dB, and the sampling frequency is 8000 Hz.

   (a) Normalize the frequencies, determine the filter order, and cutoff frequencies using MATLAB.

   (b) Implement the filter using the butter function and plot its gain response using a yellow solid line ('y').Generate a synthetic sonar signal with tones at 300 Hz, 1200 Hz, and 3000 Hz, then apply the designed filter.Plot the original signal in blue ('b') and the filtered signal in yellow ('y') to analyze the filter’s band-pass characteristics.