David Hou

Table of Contents

1. 3rd Order Elliptic Filter

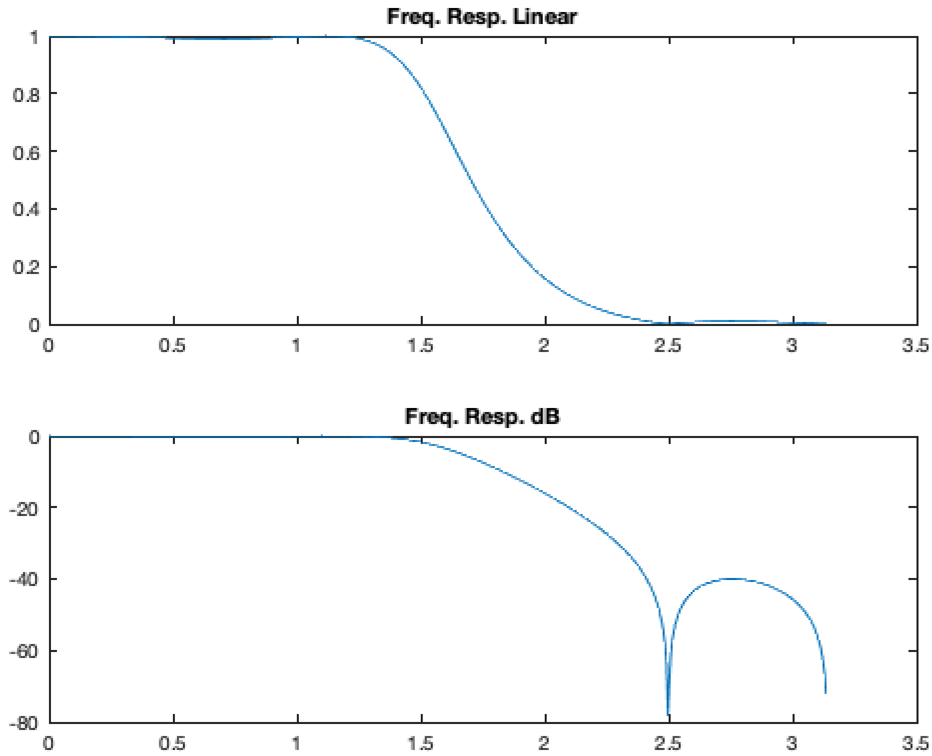
We use ellip function to generate parameters for the desired third order filter

```
dp = 0.01;
ds = 0.01;
Rp = -20*log10(1-dp);
Rs = -20*log10(ds);
[B, A] = ellip(3, Rp, Rs, 0.4);
% H(z) = (0.1256 + 0.3021z^-1 + 0.3021z^-2 + 0.1256z^-3) / (1 -
 0.6303z^-1 + 0.6550z^-2 - 0.1693z^-3)
```

- Frequency Response

We plot frequency response using freqz function, both in linear and dB scale.

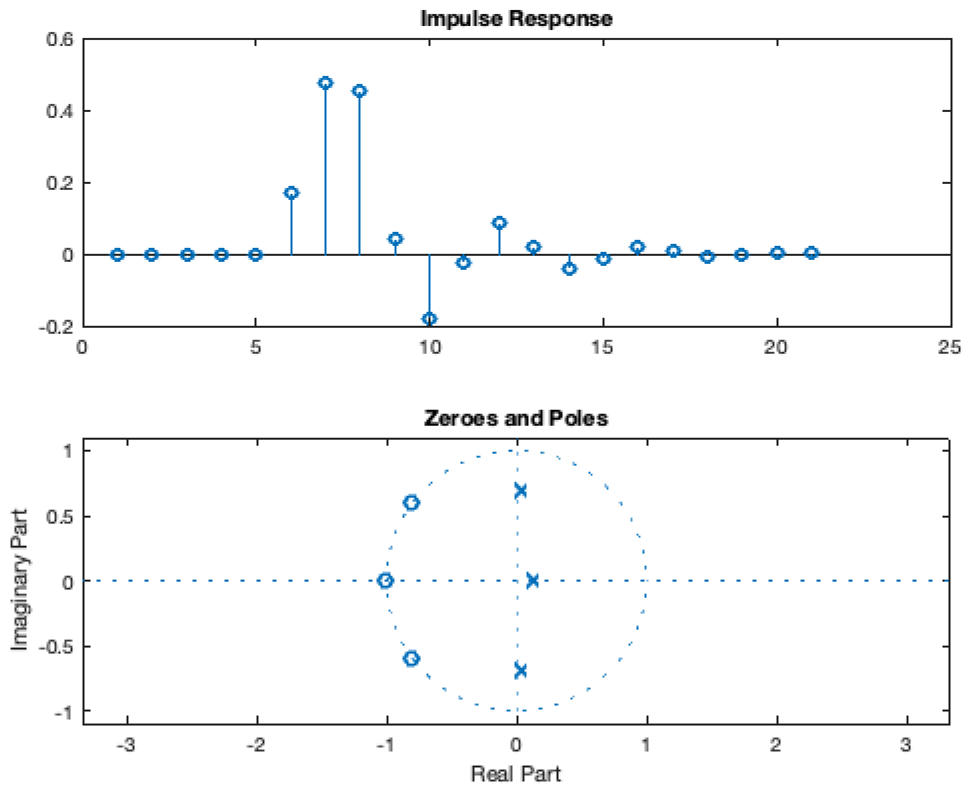
```
[H, w] = freqz(B,A);
subplot(2,1,1), plot(w, abs(H)), title('Freq. Resp. Linear');
subplot(2,1,2), plot(w, mag2db(abs(H))), title('Freq. Resp. dB');
```



- Impulse Response, Zeros and Poles

We also plot the impulse response and the poles and zeroes of the system.

```
n = -5:15;
hx = filter(B,A,(n==0));
subplot(2,1,1), stem(hx), title('Impulse Response');
subplot(2,1,2), zplane(B,A), title('Zeroes and Poles');
```



2. 4th Order Elliptic Filter

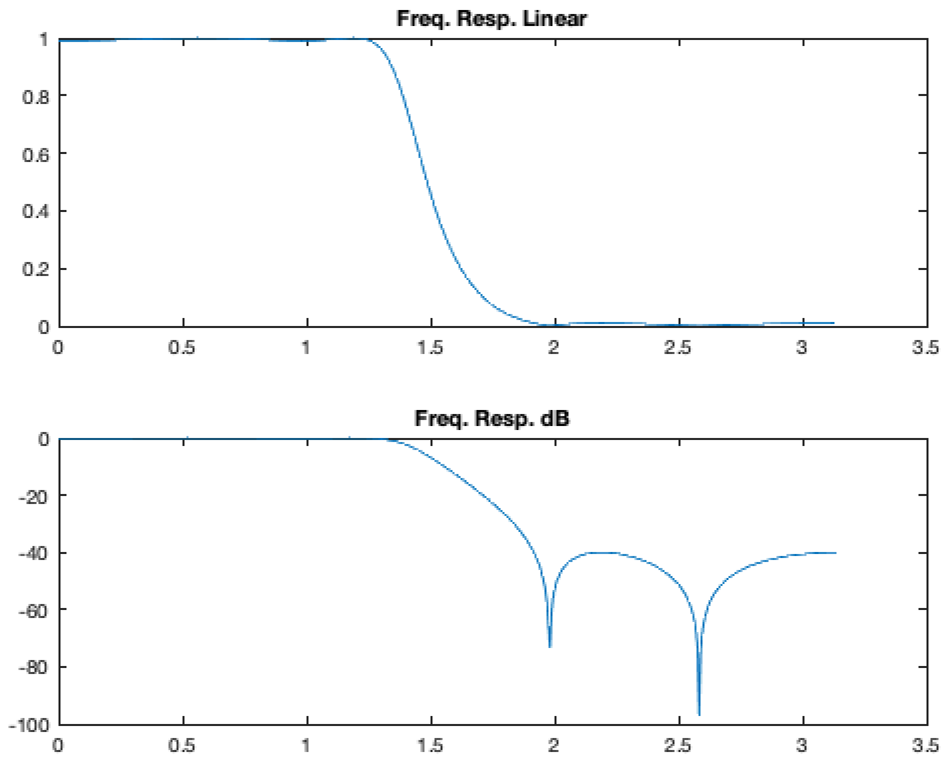
Same as above, but this time using a 4th order elliptic filter.

```
dp = 0.01;
ds = 0.01;
Rp = -20*log10(1-dp);
Rs = -20*log10(ds);
[B, A] = ellip(4, Rp, Rs,0.4);
```

- Frequency Response

We plot frequency response using freqz function, both in linear and dB scale.

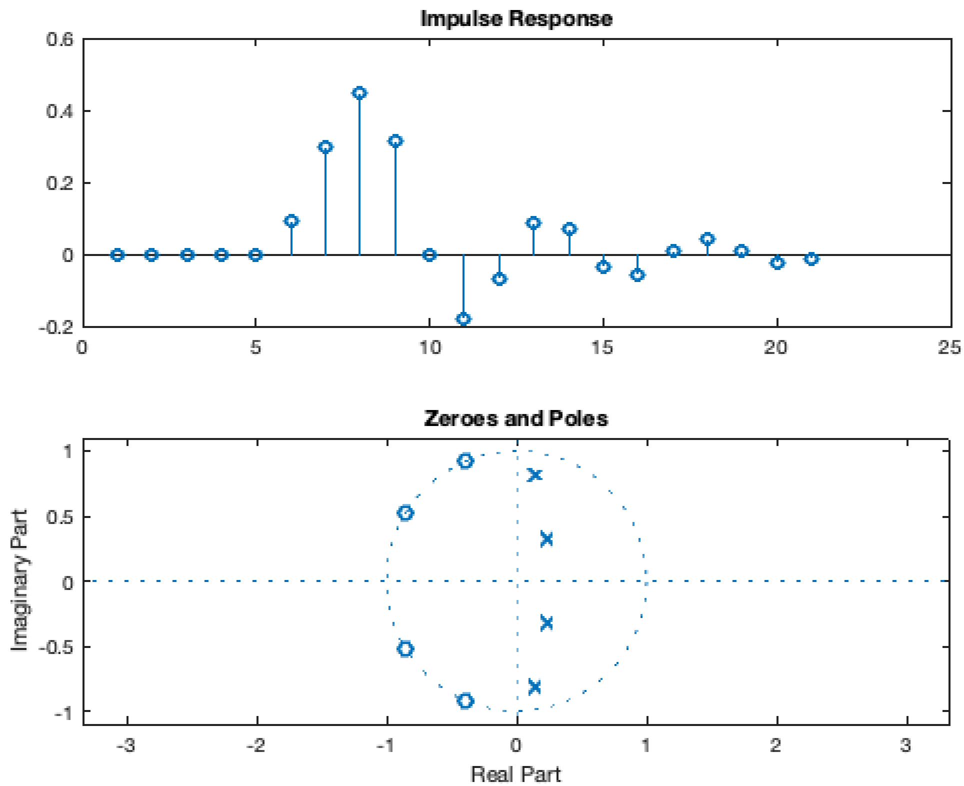
```
[H, w] = freqz(B,A);
subplot(2,1,1), plot(w, abs(H)), title('Freq. Resp. Linear');
subplot(2,1,2), plot(w, mag2db(abs(H))), title('Freq. Resp. dB');
```



- Impulse Response, Zeroes and Poles

We also plot the impulse response and the poles and zeroes of the system.

```
n = -5:15;
hx = filter(B,A,(n==0));
subplot(2,1,1), stem(hx), title('Impulse Response');
subplot(2,1,2), zplane(B,A), title('Zeroes and Poles');
```



3. Higher Order Elliptic Filter

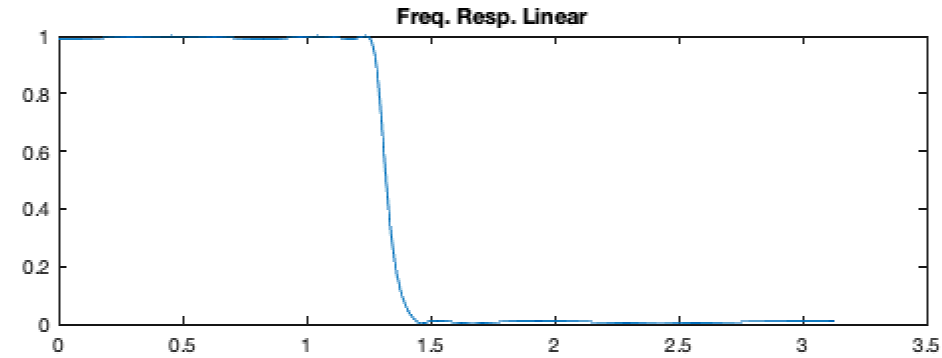
As above, but using a 6th order elliptic filter.

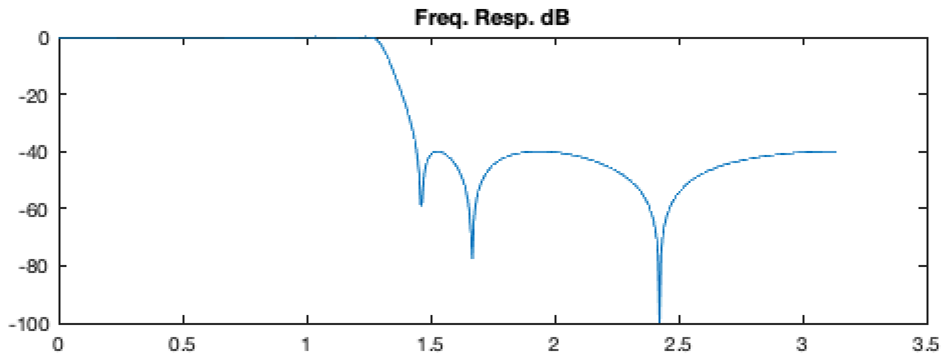
```
dp = 0.01;
ds = 0.01;
Rp = -20*log10(1-dp);
Rs = -20*log10(ds);
[B, A] = ellip(6, Rp, Rs,0.4);
```

- Frequency Response

We plot frequency response using freqz function, both in linear and dB scale.

```
[H, w] = freqz(B,A);
subplot(2,1,1), plot(w, abs(H)), title('Freq. Resp. Linear');
subplot(2,1,2), plot(w, mag2db(abs(H))), title('Freq. Resp. dB');
```



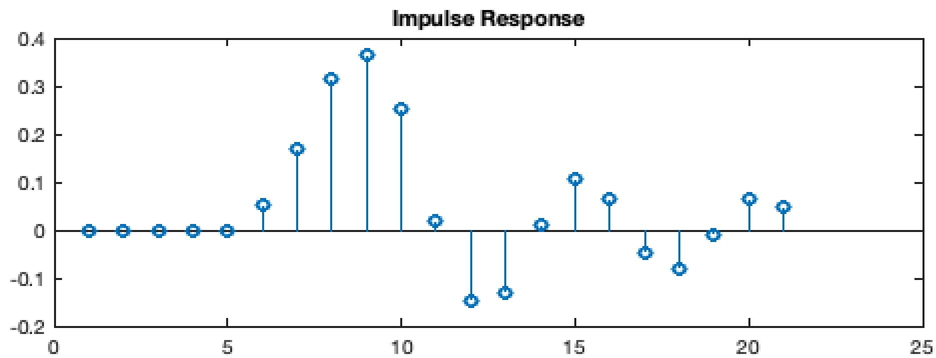


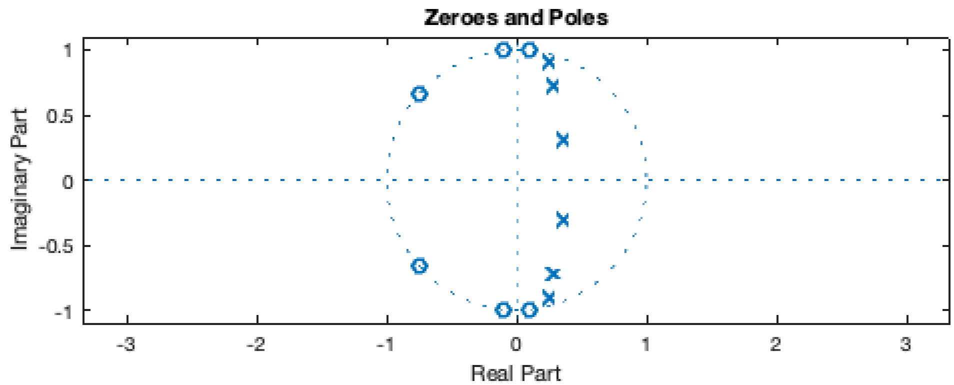
- Impulse Response, Zeroes and Poles

We also plot the impulse response and the poles and zeroes of the system.

```
n = -5:15;
hx = filter(B,A,(n==0));
subplot(2,1,1), stem(hx), title('Impulse Response');
subplot(2,1,2), zplane(B,A), title('Zeroes and Poles');

% Overall, we observe that increasing the order of the elliptic filter
% reduces the transition window between the passband and stopband.
 Though
% the ripples in the stopband and passband are kept the same,
 increasing
% the order of the elliptic filter made the filter behave more closely
 to
% an ideal filter, more quickly attenuating the signal as the
 frequency
% passes the passband threshold.
```





4. Elliptic High-Pass Filter

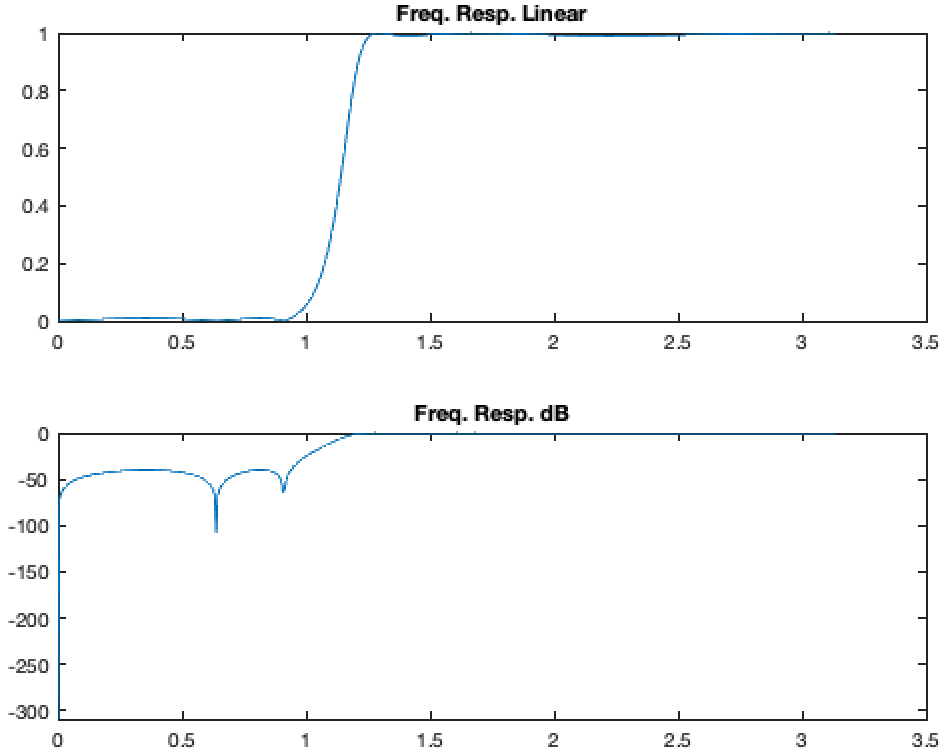
We can also use the ellip function to design highpass filters.

```
dp = 0.01;
ds = 0.01;
Rp = -20*log10(1-dp);
Rs = -20*log10(ds);
[B, A] = ellip(5, Rp, Rs, 0.4, 'high');
```

- Frequency Response

We plot frequency response using freqz function, both in linear and dB scale.

```
[H, w] = freqz(B,A);
subplot(2,1,1), plot(w, abs(H)), title('Freq. Resp. Linear');
subplot(2,1,2), plot(w, mag2db(abs(H))), title('Freq. Resp. dB');
```

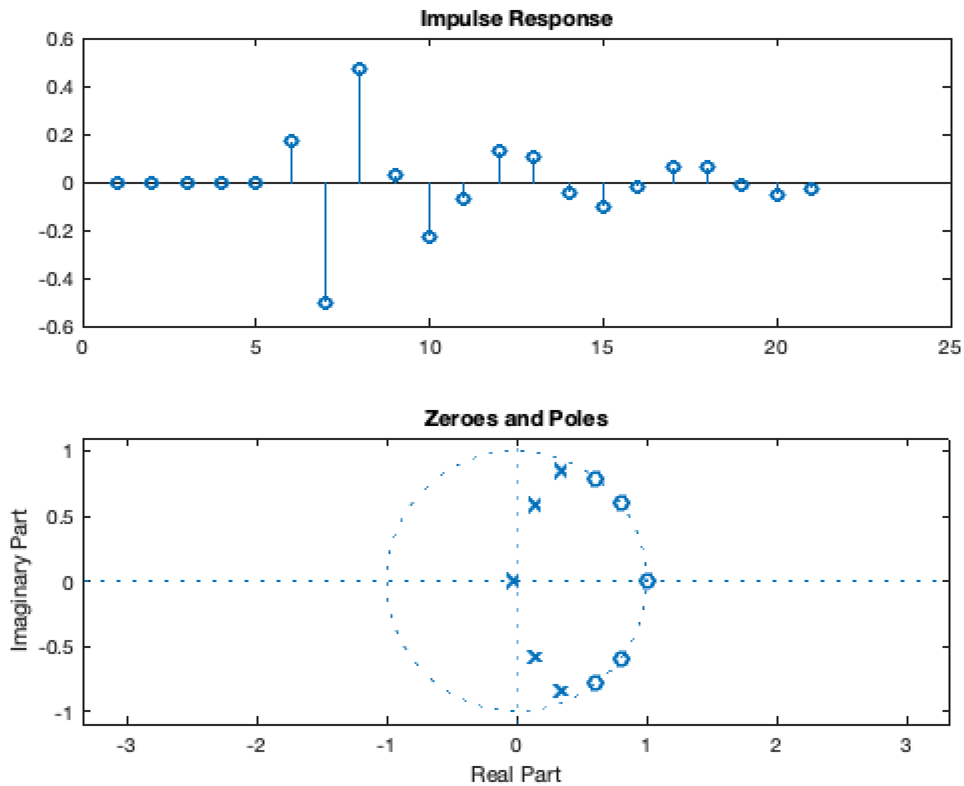


- Impulse Response, Zeroes and Poles

We also plot the impulse response and the poles and zeroes of the system.

```
n = -5:15;
hx = filter(B,A,(n==0));

subplot(2,1,1), stem(hx), title('Impulse Response');
subplot(2,1,2), zplane(B,A), title('Zeroes and Poles');
```



5. Elliptic Band-Pass Filter

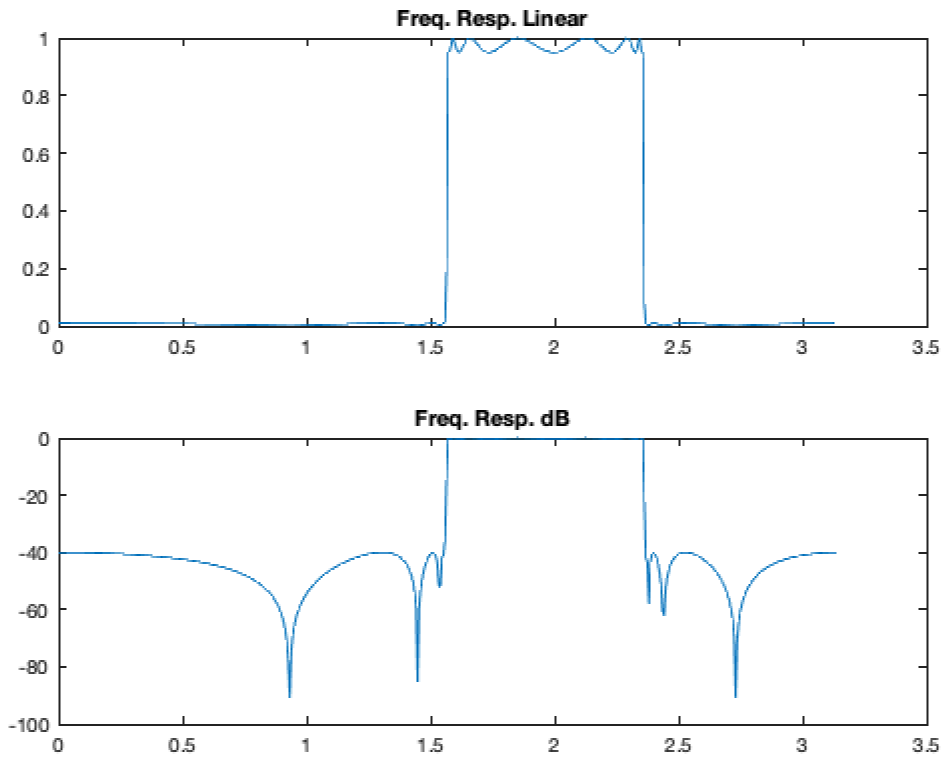
We can also use the ellip function to design bandpass filters.

```
dp = 0.05;
ds = 0.01;
Rp = -20*log10(1-dp);
Rs = -20*log10(ds);
[B, A] = ellip(8, Rp, Rs, [500 750]/1000, 'bandpass');
```

- Frequency Response

We plot frequency response using freqz function, both in linear and dB scale.

```
[H, w] = freqz(B,A);
subplot(2,1,1), plot(w, abs(H)), title('Freq. Resp. Linear');
subplot(2,1,2), plot(w, mag2db(abs(H))), title('Freq. Resp. dB');
```

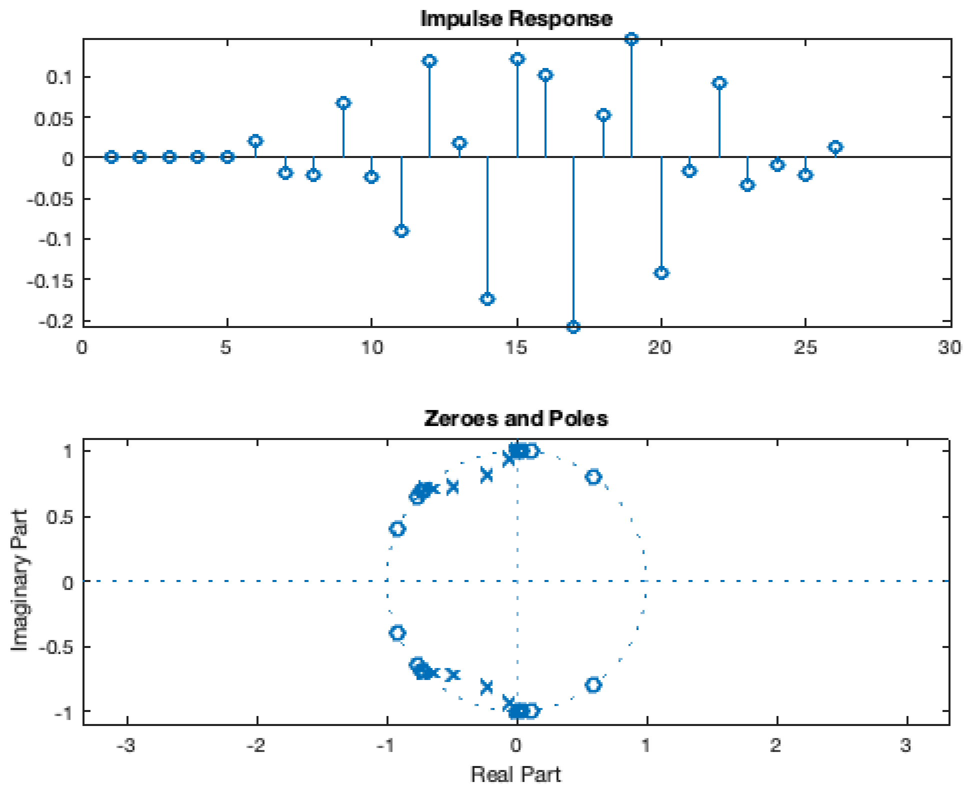


- Impulse Response, Zeroes and Poles

We also plot the impulse response and the poles and zeroes of the system.

```
n = -5:20;
hx = filter(B,A,(n==0));

subplot(2,1,1), stem(hx), title('Impulse Response');
subplot(2,1,2), zplane(B,A), title('Zeroes and Poles');
```



6. Butterworth Low-Pass Filter

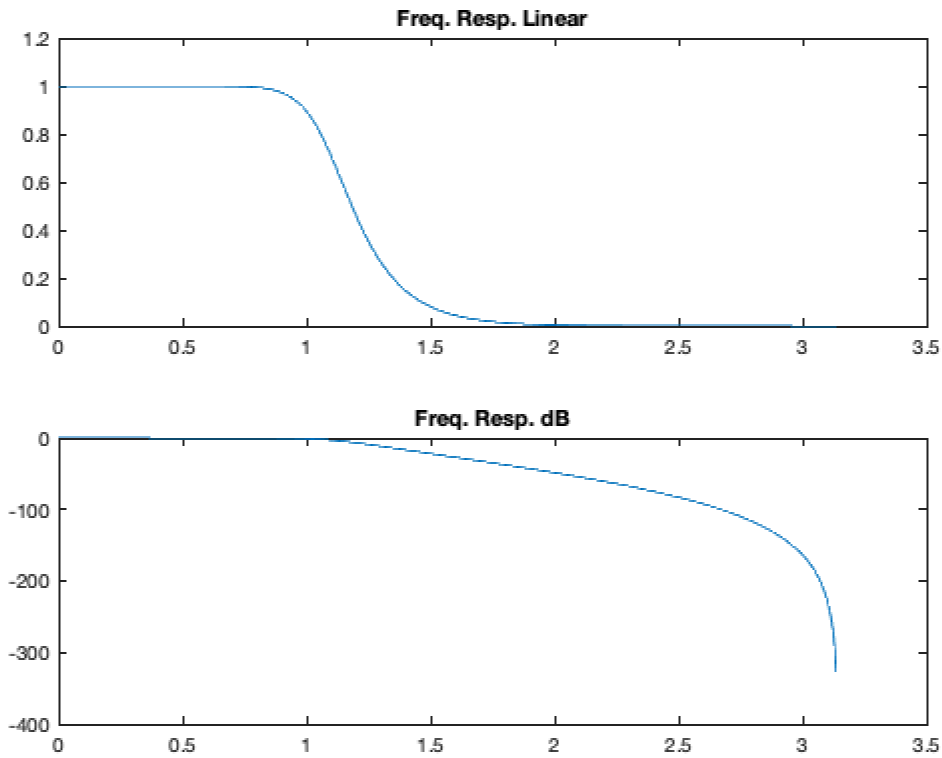
Matlab can also implement other types of filters: here, a 6th order Butterworth filter.

```
cutoff = 350/1000; % Freq 350Hz sampled at 1000 times a second.
[B,A] = butter(6, cutoff);
```

- Frequency Response

We plot frequency response using freqz function, both in linear and dB scale.

```
[H, w] = freqz(B,A);
subplot(2,1,1), plot(w, abs(H)), title('Freq. Resp. Linear');
subplot(2,1,2), plot(w, mag2db(abs(H))), title('Freq. Resp. dB');
```

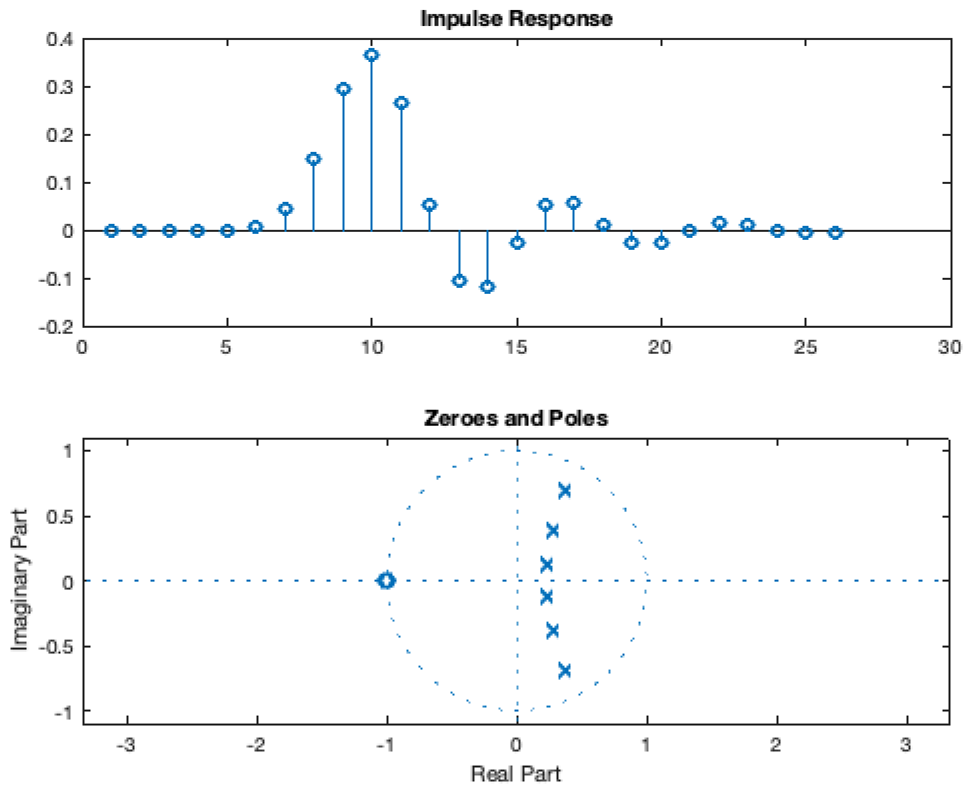


- Impulse Response, Zeroes and Poles

We also plot the impulse response and the poles and zeroes of the system.

```
n = -5:20;
hx = filter(B,A,(n==0));

subplot(2,1,1), stem(hx), title('Impulse Response');
subplot(2,1,2), zplane(B,A), title('Zeroes and Poles');
```



7. Chebyshev Type I Low-Pass Filter

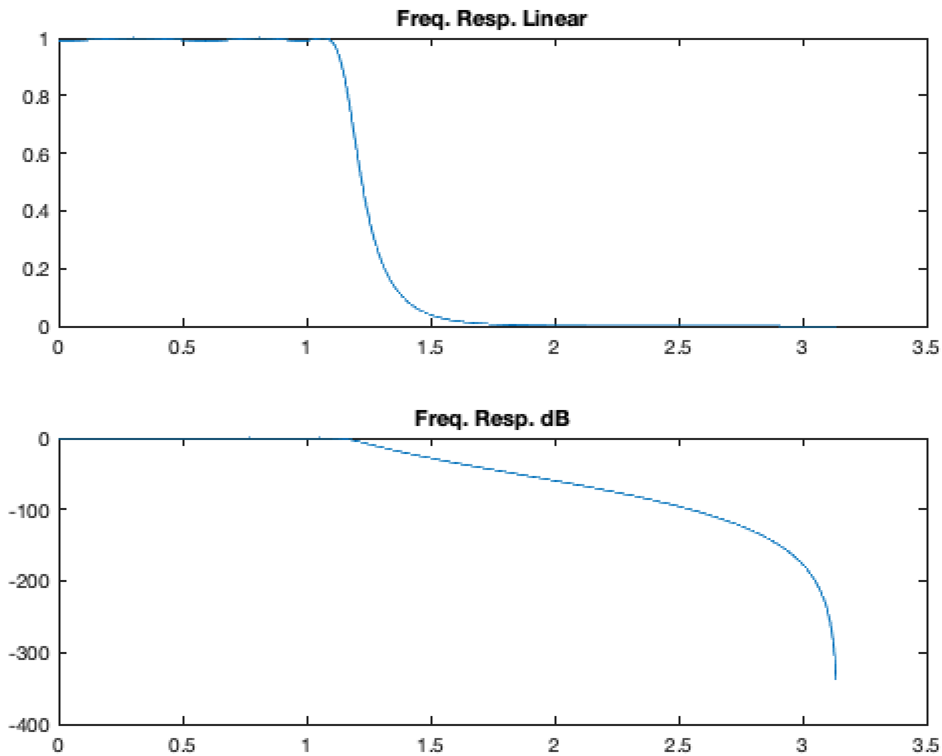
Cheby1 is used to implement a 6th order Chebyshev lowpass filter

```
dp = 0.01;
Rp = -20*log10(1-dp);
cutoff = 350/1000; % Freq 350Hz sampled at 1000 times a second.
[B, A] = cheby1(6, Rp, cutoff);
```

- Frequency Response

We plot frequency response using freqz function, both in linear and dB scale.

```
[H, w] = freqz(B,A);
subplot(2,1,1), plot(w, abs(H)), title('Freq. Resp. Linear');
subplot(2,1,2), plot(w, mag2db(abs(H))), title('Freq. Resp. dB');
```

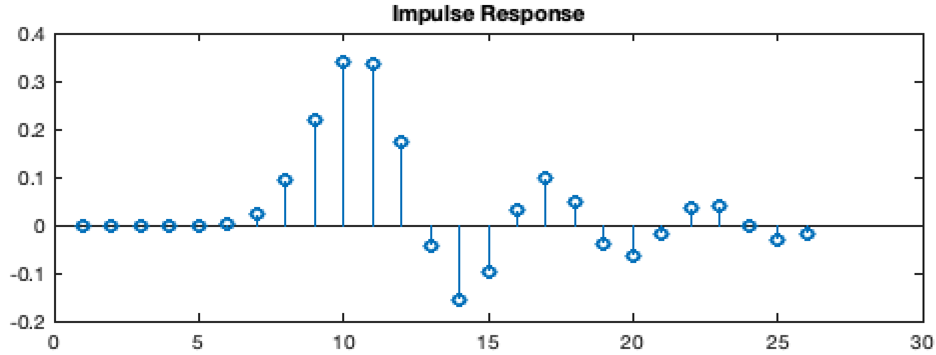


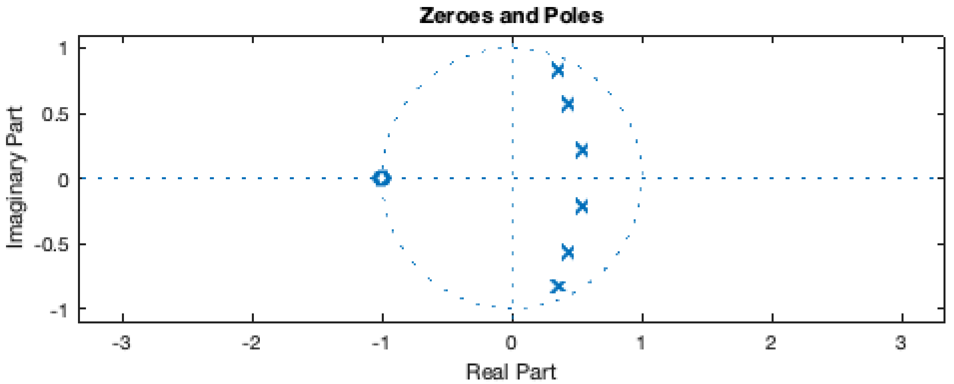
- Impulse Response, Zeroes and Poles

We also plot the impulse response and the poles and zeroes of the system.

```
n = -5:20;
hx = filter(B,A,(n==0));

subplot(2,1,1), stem(hx), title('Impulse Response');
subplot(2,1,2), zplane(B,A), title('Zeroes and Poles');
```





8. Chebyshev Type II Low-Pass Filter

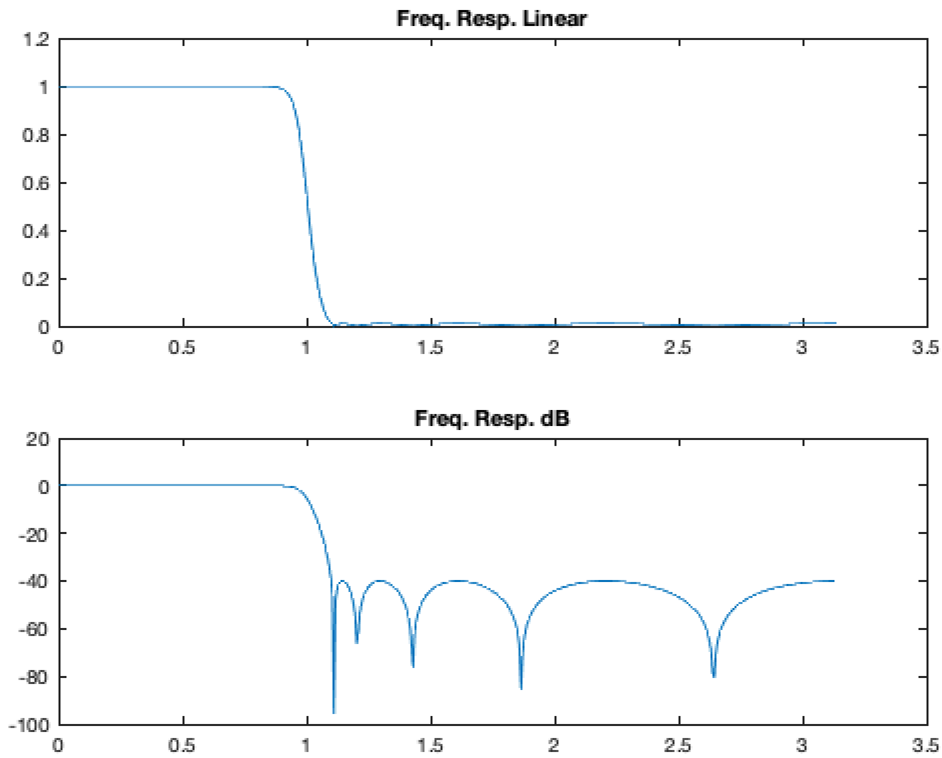
Cheby2 is used to implement a 10th order Chebyshev lowpass filter

```
ds = 0.01;
Rs = -20*log10(ds);
cutoff = 350/1000; % Freq 350Hz sampled at 1000 times a second.
[B, A] = cheby2(10, Rs, cutoff);
```

- Frequency Response

We plot frequency response using freqz function, both in linear and dB scale.

```
[H, w] = freqz(B,A);
subplot(2,1,1), plot(w, abs(H)), title('Freq. Resp. Linear');
subplot(2,1,2), plot(w, mag2db(abs(H))), title('Freq. Resp. dB');
```

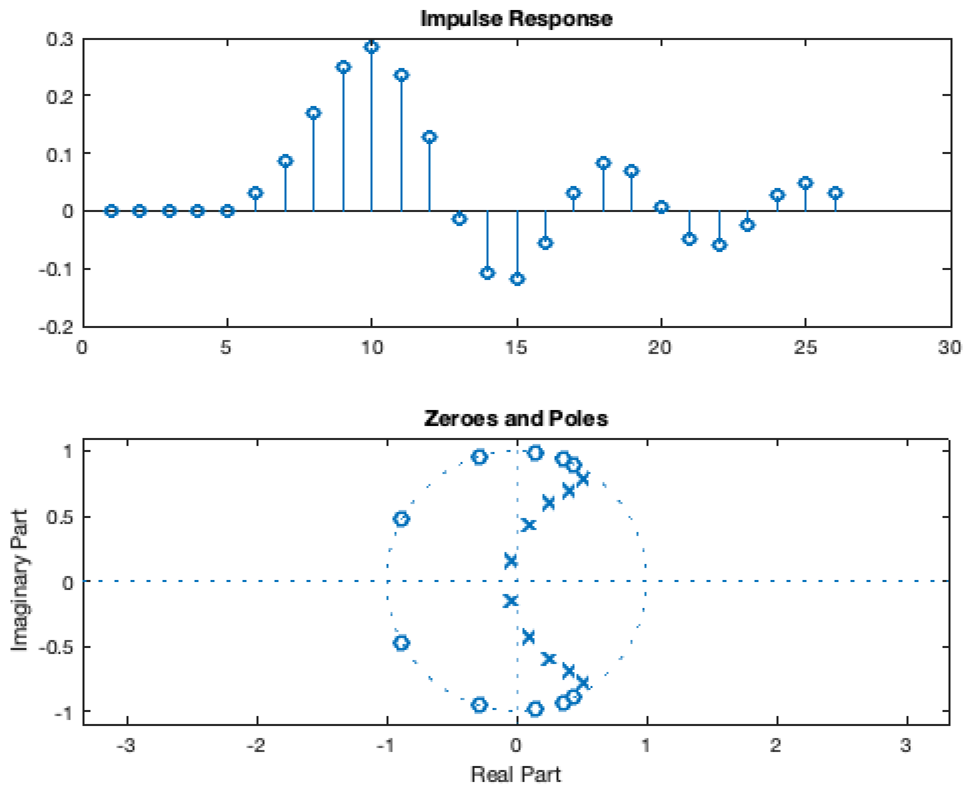


- Impulse Response, Zeroes and Poles

We also plot the impulse response and the poles and zeroes of the system.

```
n = -5:20;
hx = filter(B,A,(n==0));

subplot(2,1,1), stem(hx), title('Impulse Response');
subplot(2,1,2), zplane(B,A), title('Zeroes and Poles');
```



DISCUSSION:

```
%{
Each type of filter takes differing parameters to generate the filter,
 each
allowing different constraints to be defined. For example, the most
elaborate function of the ones explored in this lab is the elliptical
filter. It allows ripple limites to be defined for both the passband
 and
the stopband, while allowing different filter types, like highpass,
lowpass, or bandpass if the passband frequency is defined as two
 numbers.

On the otherhand, Butterworth filters are simple. Defining a low-pass
filter using the butter function requires only only a cutoff frequency
 and
the order of the filter. By passing in different Wn parameters to
 butter()
and various ftypes, it is possible to define bandpass, bandstop, and
 other
filters. However, ripple cannot be constrained in butterworth filters.

Chebyshev filters are in between the above two filter families. Type I
 has
```

```
the filter being defined by a frequency threshold and a peak-to-peak
 ripple
limit. Type II is the reverse, allowing the minimum attenuation in the
stopband to be defined.

In general, increasing the order of the filter decreases the
 transition
between the stopband(s) and passband(s), though I have observed some
 noise
around the threshold frequency when the order is very high. That said,
 as a
whole, it can be said that higher order filters behave closer to ideal
filters than lower order filters.
%}
```

Published with MATLAB® R2019a