ELEC6410 Project 5

Answers to Digital Signal Processing Project #5

Michael J. Carroll

October 29, 2010

Question 1

From the project description, I was given the transfer function.

$$H(z) = \frac{1 + 0.2z^{-1} - 0.8z{-2}}{1 + 0.7z^{-1} + 0.64z^{-2}}$$

From this, I found the pole and zero locations of the system and generated the pole-zero plot included in Figure 1.

```
num = [1, 0.2, -0.8]; den = [1, 0.7, 0.64];
[z,p,k] = tf2zp(num,den)
```

```
z =                     p =
   -1.0000                  -0.3500 + 0.7194i
    0.8000                  -0.3500 - 0.7194i
```

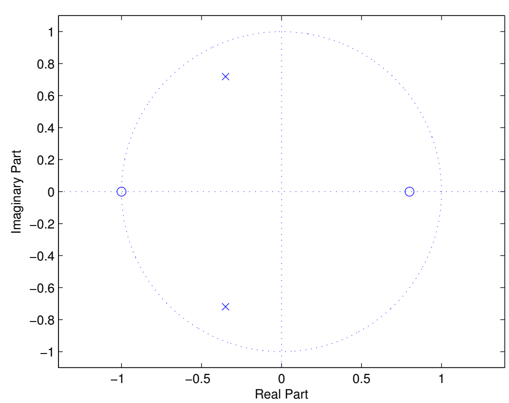

Figure 1: Pole-Zero Plot of H(z)

I then sketched the estimated frequency response from looking at the pole-zero plot. This is included in Figure 2. I then compared this with MATLAB's output from the freqz command included in Figure 3

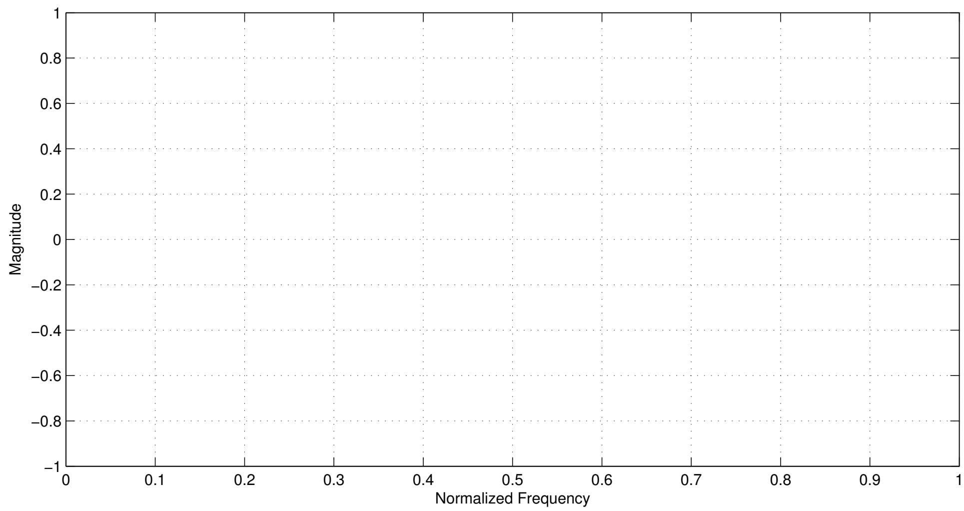

Figure 2: Sketched magnitude response of H(z)

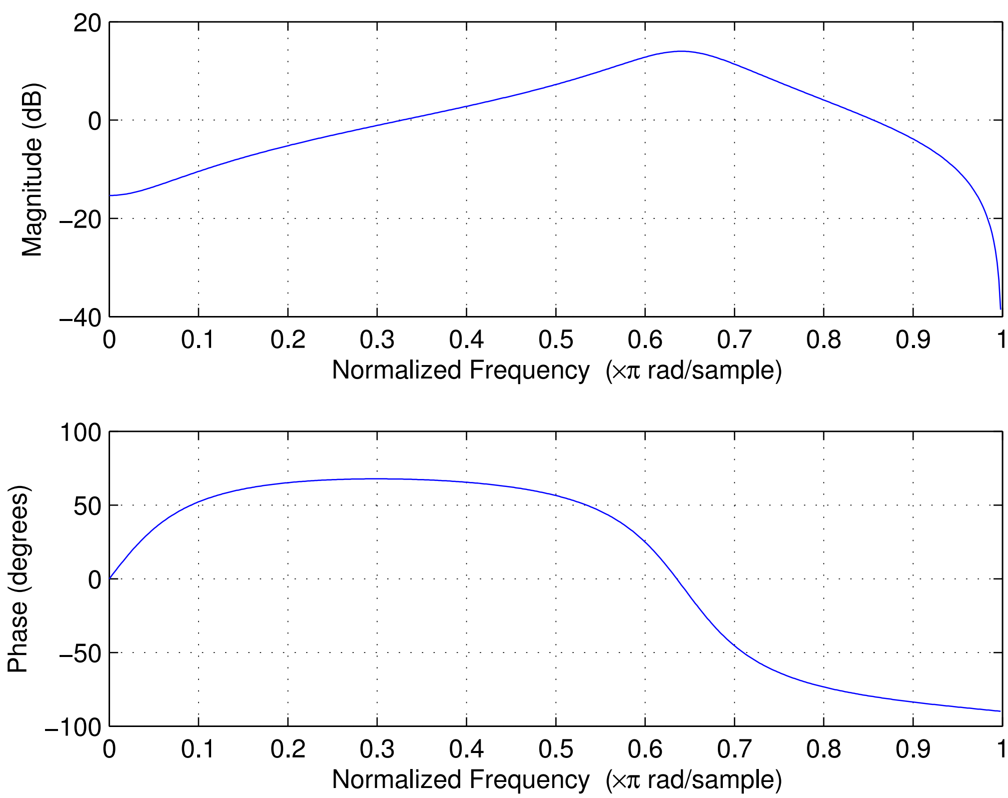

Figure 3: Frequency response of H(z)

Question 2

I then used MATLAB's fdatool to design a high-pass filter.

The transfer function of my designed filter is:

$$HP(z) = \frac{1 - 1.957z^{-1} + 0.9571z^{-2}}{1 + 0.25z^{-2}}$$

The magnitude response plot is included in Figure 4 and the zero-pole plot is included in Figure 5

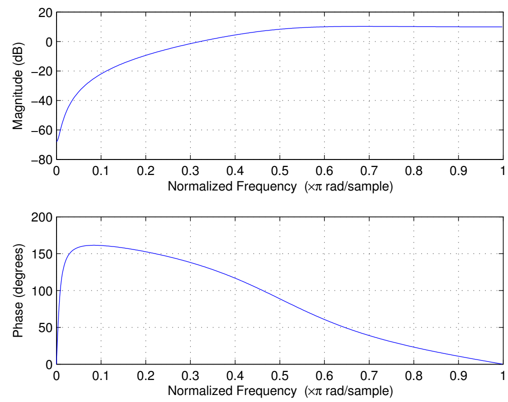

Figure 4: fdatool high-pass filter magnitude response

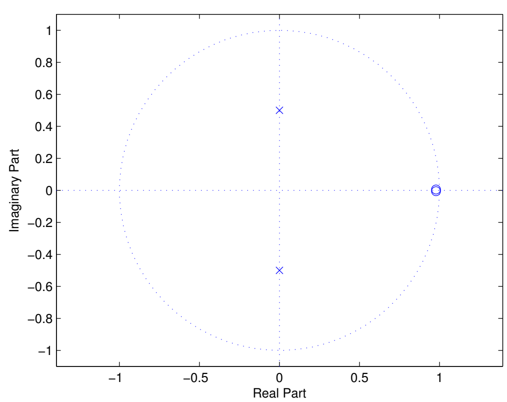

Figure 5: Pole-zero plot of fdatool high-pass filter

Question 3

The difference equation for the filter is then:

$$y[n] = -0.25y[n-1] + x[n] - 1.957x[n-1] + 0.9571x[n-2]$$

To test the filter, I generated an input signal and used the filter command. The results are in Figure 6. The filter that I created successfully attenuates the lower frequency. It is not easy to tell directly from the output of the filter, but the results are reflected better in the FFT of the output. I have included the FFT in Figure 7. The lower frequency is clearly attenuated in the output of the filter.

An interesting thing to note is that peaks appear in the output signal in places where there were no peaks in the input. This may be due to the highly nonlinear phase response of the high-pass filter.

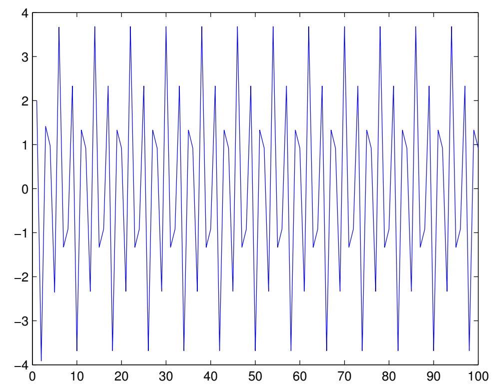

Figure 6: Filter response to sinusoidal inputs

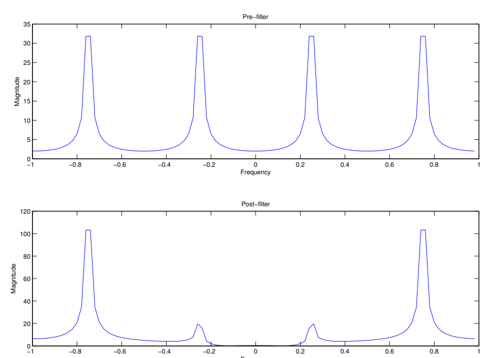

Figure 7: FFT of filter output

Question 4

I then generated the impulse response of the high-pass filter. It is included in Figure 8.

The impulse response appears to be that of a high-pass filter. A high-pass filter impulse response is characterized by a sharp "point" (or delta) at the origin, with negative samples around it. When the impulse response is convolved with an input signal, the parts of the input signal with sharp changes (high-frequency regions) will be amplified, while those with wide, sweeping changes (low-frequency regions) will be attenuated.

This is opposite of the impulse response of a low-pass function, which is characterized by a wide, sweeping appearance. A low-pass filter would be ideally characterized by a wide pulse or a sinc function.

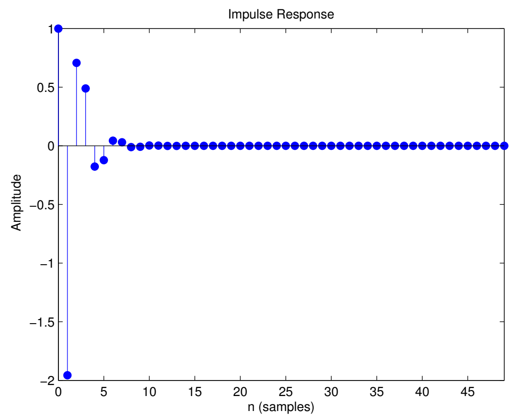

Figure 8: Impulse response of fdatool high-pass filter

Question 5

I loaded the fanfare.au music sample into MATLAB. The unfiltered file has a full range of audio, from bass notes to high notes. Once passed through the filter, the song then becomes ”tinny,” and loses all of the low-end ”punch.” This agrees with the fact that the lower frequencies have been filtered out of the song.

```
fanfare = auread('/Users/mjcarroll/Downloads/fanfare.au')
sound(fanfare)
sound(filter(HPnum,HPden,fanfare))
```