HW 9 Solution

6.34 The z-transform of a right-sided sequence $h[n]$ is given by

$$H(z) = \frac{z + 1.7}{(z + 0.3)(z - 0.5)}.$$

Find its inverse z-transform $h[n]$ via the partial-fraction approach. Using MATLAB verify the partial fraction expansion.

Solution: Let

$$H(z) = \frac{z + 1.7}{(z + 0.3)(z - 0.5)} = \frac{z^{-1} + 1.7z^{-2}}{(1 + 0.3z^{-1})(1 - 0.5z^{-1})} = \frac{A}{1 + 0.3z^{-1}} + \frac{B}{1 - 0.5z^{-1}}$$

Then,

$$A = 6.8333$$
$$B = 5.5$$

Hence,

$$h[n] = 5.8333(-0.3)^n\mu[n] + 5.5(0.5)^n\mu[n]$$

MATLAB:

```matlab
num = [0, 1, 1.7];
den = [1, -0.2, -0.15];
[r, p, k] = residuez(num, den)
```

the output parameter r is the value for B and A above.

$r, p, k = \text{residues}(b, a)$ finds the residues, poles, and direct terms of a partial fraction expansion of the ratio of two polynomials, $b(z)$ and $a(z)$. Vectors $b$ and $a$ specify the coefficients of the polynomials of the discrete-time system $b(z)/a(z)$ in descending powers of $z$.

$$B(z) = b_0 + b_1 z^{-1} + b_2 z^{-2} + \ldots + b_m z^{-m}$$
$$A(z) = a_0 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_n z^{-n}$$

If there are no multiple roots and $a > n-1$,

$$\frac{B(z)}{A(z)} = \frac{r(1)}{1 - p(1)z^{-1}} + \ldots + \frac{r(n)}{1 - p(n)z^{-n}} + k(1) + k(2)z^{-1} + \ldots + k(m - n + 1)z^{-(m-n)}$$

The returned column vector $r$ contains the residues, column vector $p$ contains the pole locations, and row vector $k$ contains the direct terms. The number of poles is

$$n = \text{length}(a) - 1 = \text{length}(z) = \text{length}(p)$$
2. [20 points] Assume one needs to process a recorded signal by filtering out some high frequency noise. In particular, assume that the desired frequency band is from 0 to 2000Hz. This signal is sampled at a sampling frequency \( f_s = 8192 \text{Hz} \). Assume that the desired lowpass filter has a ripple peak no larger than 0.002 and the transition band less than 409.6Hz. Use the Kaiser window to design a low pass filter that satisfy the above specification.

(a) Specify the \( \beta \) and \( N \) for the Kaiser window.
(b) Specify the cutoff frequency in your filter design.
(c) Using MATLAB, plot the impulse response of the FIR filter that you design in time domain. You can use \textit{stem} function.
(d) Using MATLAB, plot the magnitude of the frequency response of the corresponding FIR filter.

Basic:

– Thus the transition width is defined as \( \Delta \omega = \omega_s - \omega_p \) and the cutoff frequency of the underlying (ideal) low pass filter is

\[
\omega_c = \frac{\omega_p + \omega_s}{2}.
\]

– Define

\[
\alpha_s = -20 \log_{10} \delta
\]

Hence, \( \alpha_s \) is the value of \( \delta \) in the unit of dB.

– Then Kaiser window satisfying the \( \delta \) tolerance is given by

\[
\beta = \begin{cases} 
0.1102(\alpha_s - 8.7) & \text{if } \alpha_s > 50 \\
0.5842(\alpha_s - 21)^{0.4} + 0.07886(\alpha_s - 21) & \text{if } 21 \leq \alpha_s \leq 50 \\
0 & \text{if } \alpha_s < 21
\end{cases}
\]

– Thus if the sidelobe suppression is only 21 dB or less, one can use a rectangular window!

– The window size \( N \) is determined correspondingly as

\[
N = \frac{\alpha_s - 8}{2.285 \Delta \omega}
\]

to also satisfy the transition region requirement. Notice that the smaller \( \Delta \omega \) is (sharper transition), the larger \( N \) needs to be.
\[ \Delta \omega = \frac{409.6 \times \pi}{8192} = 0.314 \]

\[ \alpha_0 = -20 \log_{10} 0.002 = 53.98 \]

since \( \alpha > 50 \)

\[ \beta = 0.1102 (\alpha_0 - 8.7) = 0.1102 (53.98 - 8.7) = 4.99 \]

\[ N = \frac{\alpha_0 - 8}{2.285 \Delta \omega} = \frac{53.98 - 8}{2.285 \times 0.314} = 64.08 \rightarrow 65 \text{ odd number} \]

(b) \[ \omega_c = \frac{\omega_0}{f_s} = \frac{2\pi \times 2000}{8192} = 1.533 \]

(c) \[ w[n] = \frac{I_0 \left[ \beta \sqrt{1 - \left(\frac{n}{M}\right)^2} \right]}{I_0(\beta)}, \quad -M \leq n \leq M \]

Then

(c) Ideal filter:

\[ h[n] = \frac{\sin \omega_c n}{n \pi} \]

Kaiser window:

\[ w_k[n] = \frac{I_0(4.99) \sqrt{1 - \left(\frac{n}{M}\right)^2}}{I_0(4.99)} \]

matlab:

\[ w_k = \text{kaiser}(65, 4.99) \]

Designed filter:

\[ h'[n] = h[n] w_k[n] \]

(d) use fft to plot \( h'[n] \)
clear all;
wc = 1.533;
n = -32:32;
hn = sin(wc * n) ./ n / pi;
hn(33) = wc / pi;
wn = kaiser(65, 4.99);

%gn = hn .* wn';
%gn = gn / norm(gn);
stem(n, gn);

Hk = fft(gn);
figure;
stem(abs(Hk));
3. [20 points] We have learned in class that the following FIR system
\[ y[n] = \frac{1}{2}(x[n] + x[n - 1]) \]
acts as a lowpass filter. Consider now the following IIR filter
\[ y[n] = 0.05x[n] + 0.05x[n - 1] + 0.9y[n - 1] \]
The difference lies in that there is also a recursive term \( 0.9y[n - 1] \).

(a) Find the transfer function of both systems and the corresponding zeros and poles.
(b) Find the frequency responses \( H(e^{j\omega}) \) of both systems in the form
\[ H(e^{j\omega}) = \frac{Y(e^{j\omega})}{X(e^{j\omega})} \]

(c) Plot the frequency responses in MATLAB for both systems. You can use any method you like, i.e., direct plot or using freqz command in MATLAB to compute the frequency responses. Use `subplot(2,1,1)`
   `plot( ... )`
   `subplot(2,1,2)`
   `plot( ... )`
to line up the two plots.
(d) Based on the frequency response plot, indicate what type of filter each of the two is. Compare the two frequency responses, comment on the difference between the two and explain using the poles and zeros of the two systems.
2. (a) \( y[n] = \frac{1}{2} (x[n] + x[n-1]) \)

\[ H_1(z) = 0.5 + 0.5 z^{-1} = \frac{0.5 z + 0.5}{z} \]

Zero: \( z = -1 \)

Pole: \( z = 0 \)

\[ y[n] = 0.5 x[n] + 0.5 x[n-1] + 0.9 y[n-1] \]

\[ H_2(z) = \frac{0.05 + 0.05 z^{-1}}{1 - 0.9 z^{-1}} \]

Zero: \( z = -1 \)

Pole: \( z = 0.9 \)

(b) \( H_1(e^{j\omega}) = 0.05 + 0.05 e^{-j\omega} \)

\( H_2(e^{j\omega}) = \frac{0.05 + 0.05 e^{-j\omega}}{1 - 0.9 e^{-j\omega}} \). 

(c)
(d) Both are lowpass, because both have zero at $z = -1$ corresponding to $w = \pi$.

Difference lies in that the second filter is sharper and cuts off nonzero frequency components more quickly, thus having a narrow transition band. This is because the second filter also has a pole at $z = 0.9$, which is close to 1, and hence boosts frequency components around $w = 0$. 

```matlab
1 clear all;
2 w = -pi:0.01:pi;
3 h1 = 0.5 + 0.5 * exp(-j * w);
4 h2_num = 0.5 + 0.5 * exp(-j * w);
5 h2_den = 1 - 0.9 * exp(-j * w);
6 h2 = h2_num ./ h2_den;
7 subplot(2,1,1);
8 plot(w / pi, abs(h1));
9 subplot(2,1,2);
10 plot(w / pi, abs(h2));
```