Lab 8

Frequency Domain Analysis

8.1 Introduction

In the previous labs, we introduce simple time-domain techniques such as moving-average filters. Here in this lab, we will introduce the frequency domain techniques to analyze signals and systems.

8.1.1 The Z-Transform

The Z-transform of a signal $x[n]$ is defined as

$$X(z) = \sum_{n=-\infty}^{\infty} x[n]z^{-n} \tag{8.1}$$

where z is a complex variable.

The set of z values for which $X(z)$ exist is called the region of convergence. To recover $x[n]$ from $X(z)$, we use the inverse Z-transform.

8.1.2 The System Function

The output of a system, $y[n]$ in Z-domain is given by $Y(z)$. The system function $h[n]$ is given by $H(z)$, and the input $x[n]$ is given by $X(z)$. We can define the output as

$$Y(z) = H(z)X(z) \tag{8.2}$$

and the system function can be computed by

$$H(z) = \frac{Y(z)}{X(z)} \tag{8.3}$$

8.1.3 Poles and Zeros

The system function can be written as

$$H(z) = \frac{b_0(z - z_1)(z - z_2)...(z - z_{W-1})}{(z - p_1)(z - p_2)...(z - p_W)} \tag{8.4}$$

where z_i represent the zeros and p_m represent the poles of the system function.

To find the zeros and poles of a system, we can use the MATLAB's *roots*() function. For a causal system, the system is stable if and only if the transfer function has all its poles inside the unit circle. i.e.

$$|p_m| < 1, m = 1, 2, ..., W \tag{8.5}$$

The poles can be viewed via the MATLAB's *zplane*() command. The screen shot of location of poles and zeros for a moving-average filter is shown in Fig. 8.1. The unfilled circles represent zeros and the crosses represent the poles of the system.

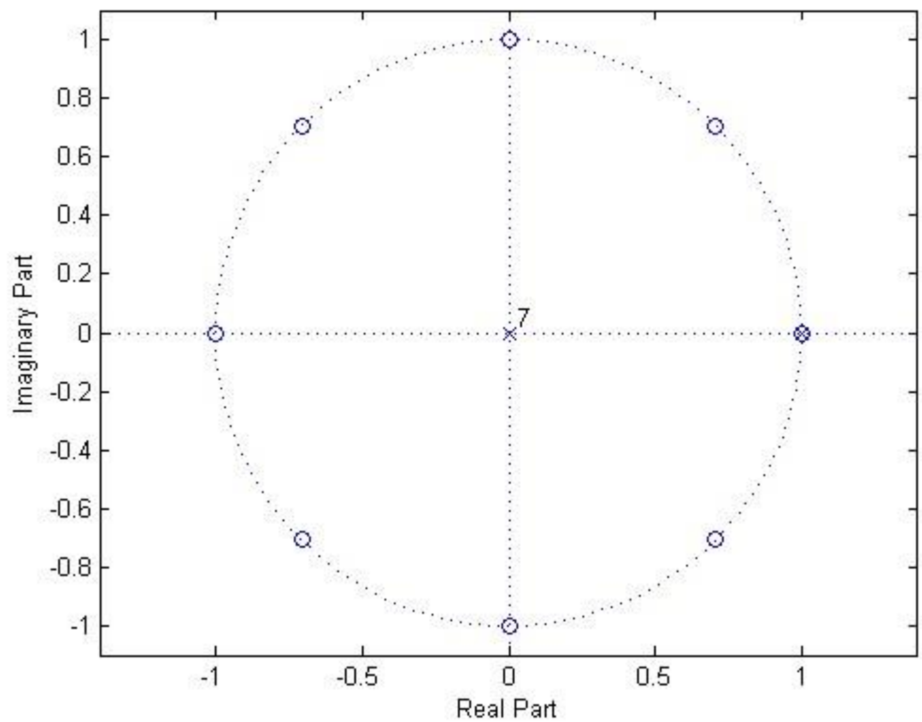

Figure 8.1: The pole zero plot of a moving-average filter

8.1.4 The Frequency Response

The frequency response of a system can be determined from its system function by replacing z with e^{jw} and then converting $H(z)$ to polar form.

$$H(z = e^{jw}) = H(e^{jw}) = |H(e^{jw})|\angle(H(e^{jw})) \tag{8.6}$$

The frequency response is then subdivided into magnitude and phase response. The frequency response of a digital system can be computed by the MATLAB's function *freqz*(). Example of a moving-average filter of length 8 is shown by Fig. 8.2.

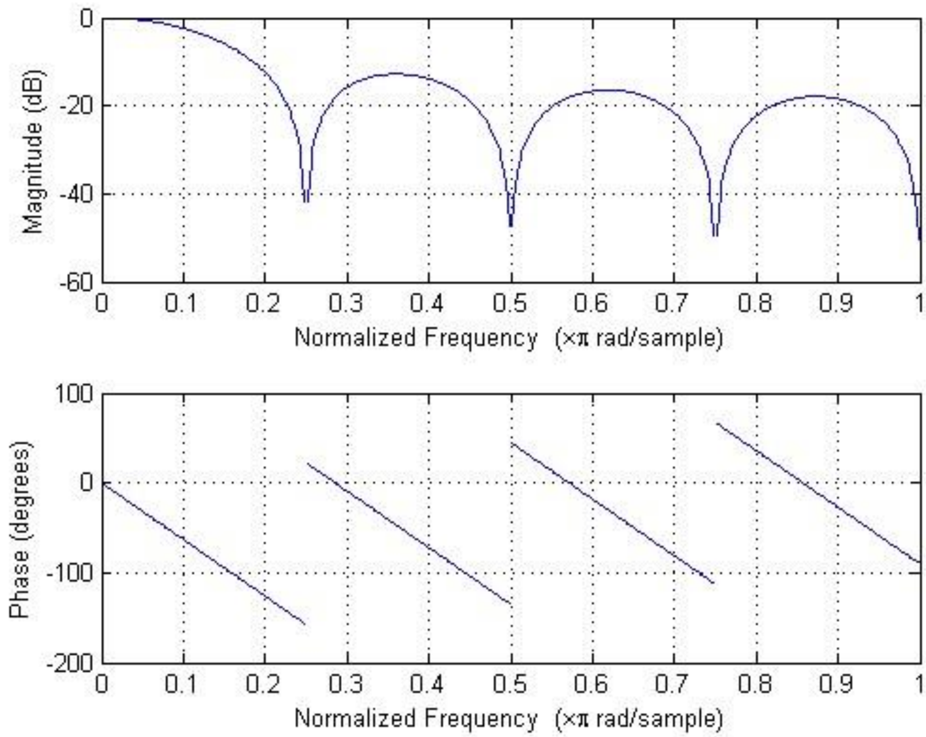

Figure 8.2: The magnitude and phase response of moving-average filter

8.1.5 MATLAB's FVTool

The MATLAB's fvtool can also be used to compute the response of a digital filter. Not only the magnitude and phase responses are provided to the user, but quantities like group delay and phase delay can also be dealt with easily using this GUI. The Fig. 8.3 shows the screen shot of the fvtool.

We see that the quantities that fvtool can provide us are:

- The Magnitude Response
- The Phase Response
- Superimposes of magnitude and phase response
- Group delay of the filter
- Phase delay of the system
- Impulse response of the system
- Step response of the current filter

- Pole-zero plot of the filter
- The filter coefficients of the current filter

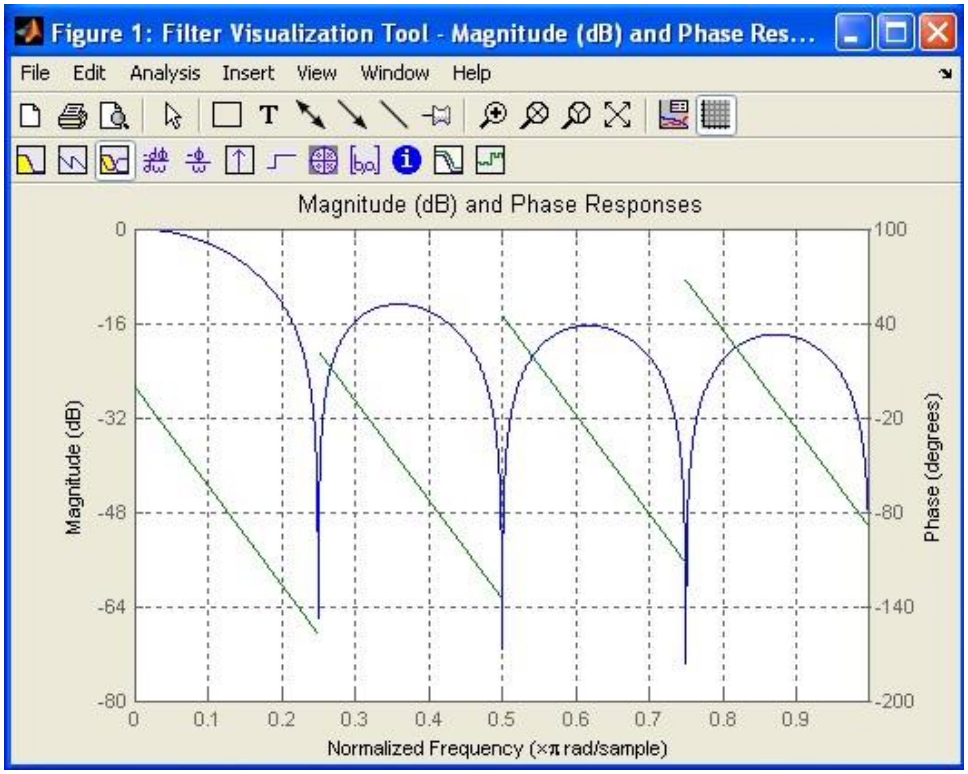

Figure 8.3: The MATLAB's fvtool for a specific filter

Thus, we can see that the fvtool is a powerful tool for analyzing the digital filters.

8.2 Practicals

1. Simulate the *zplane*() of a moving-average filter of lengths 5 and 8. Also determine the magnitude and phase response using the *freqz*() command.
2. Derive the transfer function of digital system defined by the equation:

$$y[n] - 2y[n-1] + y[n-2] = 0.5(x[n] + x[n-1]) \tag{8.7}$$

   Is the system stable? Find the magnitude and phase response of the system. Also display the pole-zero plot and the impulse response of the current system.

3. Find the transfer function of the following system:

$$y[n] = 0.1x[n] + 0.2x[n-1] + 0.1x[n-2] \tag{8.9}$$

   Is the system stable? Plot the magnitude and phase response.

8.3 Hints

To use fvtool for the digital filters, find the coefficients of the numerator and denominator of the filter function. Then invoke the fvtool by typing:

fvtool(< *numerator* >,< *denominator* >)