**University of Illinois at Urbana-Champaign**
**Department of Electrical and Computer Engineering**

**ECE 311: Digital Signal Processing Lab**
**Chandra Radhakrishnan**
**Peter Kairouz**

# LAB 4: **Difference Equations, z-Transforms, Pole-Zero Diagrams, BIBO Stability and Quantization Effects**

Summer 2011

## 1 Overview

In this lab we will use MATLAB to solve difference-equations, analyze the stability of LSI systems in terms of the pole-zero location of their transfer function, and study the effects of quantization on different filter structures.

## 2 The z-Transform

The bilateral $z$ transform of a discrete-time signal $x[n]$,

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

is a generalization of the discrete-time Fourier transform that is useful for studying discrete-time signals and systems. Note that if $z = e^{j\omega}$, the $z$-transform reduces to the discrete-time Fourier transform. However, the bilateral $z$-transform exists for a broader class of signals than the discrete-time Fourier transform does, and is useful for understanding the behavior of both stable and unstable systems. For a large class of signals, the $z$-transform can be represented as a ratio of polynomials in $z$, i.e.,

$$X(z) = \frac{N(z)}{D(z)}$$

These transforms are called rational transforms and arise as the system functions of LTI systems which satisfy linear constant-coefficient difference equations. The locations of the roots $N(z)$ and $D(z)$, known as the zeros and poles of the system, respectively, determine to within a constant multiplicative factor the behavior of LTI systems with rational transforms. Therefore, plots of the pole and zero locations can be used to analyze system properties.

## 3 Discrete-Time Pole-Zero Diagrams

The poles and zeros of rational system function can be computed using the function `roots`. The function `roots` requires the coefficient vector to be in descending order of the independent variable. For example, consider the LTI system with system function,

$$H(z) = \frac{z^2 - z}{z^2 + 3z + 2} \tag{1}$$

The poles and zeros can be computed by executing

```
>> b = [1 -1 0]
>> a = [1 3 2]
>> zs = roots(b)
>> zs =
        0
        1
>> ps = roots(a)
ps =
     -2
     -1
```

It is often desirable to write discrete-time system functions in terms of increasing order of $z^{-1}$. The coefficients of these polynomials are easily obtained from the linear constant coefficient difference equation and are also in the form that `filter` requires. However, if the numerator and denominator do not have same order, some poles or zeros at $z = 0$ may be overlooked. Example (1) can be rewritten as,

$$H(z) = \frac{1 - z^{-1}}{1 + 3z^{-1} + 2z^{-2}} \tag{2}$$

If we now use the `roots` command we will get,

```
>> b = [1 -1]
>> a = [1 3 2]
>> zs = roots(b)
>> zs =
        1
>> ps = roots(a)
ps =
     -2
     -1
```

Note that $z = 0$ does not appear here. In order to find the complete set of poles and zeros when working with a system function in terms of $z^{-1}$, you must append zeros to the coefficient vector for the lower order polynomial such that the coefficient vectors are the same length. The function `dpzplot(a,b)` plots the poles and zeros of discrete-time systems.

```
function dpzplot(b,a)
%dpzplot(b,a)
%plots the pole-zero diagram for the discrete-time system function
%H(z)=b(z)/a(z) defined by numerator and denominator polynomials b and a.

la = length(a);
lb = length(b);
if(la>lb)
    b = [b zeros(1,la-lb)];
elseif (lb > la)
    a = [a zeros(1,lb-la)];
end
ps = roots(a);
zs = roots(b);
```

```
mx = max(abs([ps' zs' 0.95])) + 0.5;
clf
axis([-mx mx -mx mx]);
axis('equal');
hold on
w = [0 : 0.01:2*pi];
plot(cos(w),sin(w),'.');
plot([-mx mx],[0 0 ]);
plot([0 0], [-mx mx]);
text(0.1,1.1,'Im','sc');
text(1.1,0.1,'Re','sc');
plot(real(ps),imag(ps),'x');
plot(real(zs),imag(zs),'o');
numz=sum(abs(zs)==0);
nump=sum(abs(ps)==0);
if numz > 1
    text(-0.1,-0.1,num2str(numz));
elseif nump>1
    text(-0.1,-0.1,num2str(nump));
end
holdoff
```

A causal LTI system is one whose unit sample response $h[n]$ satisfies the condition

$$h[n] = 0, \quad n < 0$$

We also know that the ROC of the $z$-transform of a causal sequence is the exterior of a circle. Consequently, a linear time-invariant system is causal if and only if the ROC of the system function is the exterior of a circle or radius $r < \infty$ including the point $z = \infty$. The stability of a LTI system can also be expressed in terms the characteristics of the system function. The necessary and sufficient condition for a LTI system to be BIBO stable is

$$\sum_{n=-\infty}^{\infty} |h[n]| \leq \infty$$

In turn, this condition implies that $H(z)$ must contain the unit circle. Hence if the system is BIBO stable, the unit circle is contained in the ROC of $H(z)$. The converse is also true. Therefore, a linear time-invariant system is BIBO stable if and only if the ROC of the system function includes the unit circle. Note that ROC cannot contain any poles of $H(z)$ and if we also consider the ROC of a causal system then it follows that a causal LTI system is BIBO stable if and only if all the poles are inside the unit circle.

## 4 Quantization

So far we have assumed that we are dealing with discrete-time systems characterized by linear difference equations with constant coefficients, where both the coefficients and signal variables have infinite precision. However, when implemented in either software form in general purpose computer or in special purpose hardware form, the system parameters along with the signal variables can take only discrete values within specified range since the registers of the digital machine where they are stored are of finite length.

The function `ellip` can be used to design a discrete-time elliptic filter. Elliptic frequency selective filters have a frequency response magnitude which is equiripple in the pass-band, i.e., the frequency response magnitude oscillates between $1 \pm \delta_1$ in the pass-band and between $\delta_2$ in the stop band. Consider an eighth order elliptic filter returned by the call,

```
>>[b,a] = ellip(4,0.2,40,[0.41 0.47]);
```

This filter has 0.2 dB ripple in pass-band, $0.41\pi \leq |\omega| \leq 0.47\pi$ and has 40 dB attenuation in the stopband. When discrete-time filters are implemented with quantized coefficients the resulting systems are called digital filters. Digital filters are generally implemented using fixed-point arithmetic on integer digital signal processing (DSP) chips. The coefficients of digital filter must be quantized to the number of bits available on the DSP chip. The function `quant` below will quantize the coefficients in the vector `x` to `N` bits, where `M` is the maximum possible amplitude of each element.

```
function qc = quant(x,N,M)
% QUANT Q = QUANT(x,N,M) quantizes the values of x(n) into
%           2^N values. The argument M is the value of the
%           maximum amplitude of x[n]

[mm, nn] = size(x);
qc = zeros(mm,nn);
levels = 2^(N-1);
maxlevel = 2^(N-1);

for  k = 1 : mm
    tmp = fix((x(k,:)+M)./(M/levels));
    q = zeros(1,nn);
    q(tmp <= maxlevel) = tmp(tmp <= maxlevel);
    q(tmp > maxlevel) = maxlevel * ones(1,length(tmp(tmp>maxlevel)));
    q(tmp < 0) = zeros(size(tmp(tmp < 0)));
    q = (q- levels)*M/levels;
    qc(k,:)=q;
end
```

Coefficient quantization can have an impact on the frequency response magnitude of the elliptic filter. In problems 4 and 5 we will look at two different implementations (Direct Form II and Cascade form) of filters and study the effects of quantization on these implementations. Note: the function `filter` in MATLAB uses direct form structure.

## 5 Homework - Due 07/12/2011 at 5:00 PM

1. The difference equation of an LTI system can be written as,

$$\sum_{k=0}^{K} a_k y[n-k] = \sum_{m=0}^{M} b_m x[n-m] \tag{3}$$

where $x[n]$ is the system input and $y[n]$ is the system output. The coefficients $a_k$ and $b_m$ can be expressed as vectors `a` and `b` respectively, such that

$$\sum_{k=0}^{K} a(k+1) y[n-k] = \sum_{m=0}^{M} b(m+1) x[n-m] \tag{4}$$

(a) Define coefficient vectors `a` and `b` to describe the causal LTI system described by,

$$y[n] = 0.5x[n] + x[n-1] + 2x[n-2].$$

(b) Define coefficient vectors `a` and `b` to describe the causal LTI system described by,

$$y[n] = 0.8y[n-1] + 2x[n].$$

(c) Define coefficient vectors `a` and `b` to describe the causal LTI system described by,

$$y[n] - 0.8y[n-1] = 2x[n-1].$$

For each of these systems use the `filter` command to compute the response $y[n]$ for the input signal $x[n] = u[n], 0 \leq n \leq 3$. Also plot the response for each of the above cases.

2. Consider the following difference equation

$$y[n] = ay[n-1] + x[n]$$

(a) Write a function `y=diffeqn(a,x,yn1)` which computes the output $y[n]$ of the causal system determined by the given equation. The input vector `x` contains $x[n]$ for $0 \leq n \leq N-1$ and `yn1` supplies the values of $y[-1]$. The output vector `y` contains $y[n]$ for $0 \leq n \leq N-1$. The first line of your M-file should read,

```
function y = diffeqn(a,x,yn1)
```

(b) Assume $a = 1$, $y[-1] = 0$, and that we are only interested in the output over the interval $0 \leq n \leq 30$. Use your function to compute the response due to $x_1[n] = \delta[n]$ and $x_2[n] = u[n]$, the unit impulse and unit step, respectively. Plot each response using `stem`.

(c) Assume again that $a = 1$, but that $y[-1] = -1$. Use your function to compute $y[n]$ over $0 \leq n \leq 30$ when the inputs are $x_1[n] = u[n]$ and $x_2[n] = 2u[n]$. Define the outputs produced by the two signals to be $y_1[n]$ and $y_2[n]$, respectively. Use `stem` to plot $(2y_1[n] - y_2[n])$. Given that the equation is a linear difference equation, why isn't this difference identically zero?

(d) When is the causal system described by the given difference equation BIBO stable? Assume $a = 1/2$ and that `x` contains $x[n] = u[n]$ for $0 \leq n \leq 30$. Assuming both $y[-1] = 0$ and $y[-1] = 1/2$, compute the two output signals $y[n]$ for $0 \leq n \leq 30$. Use `stem` to display both responses. How do they differ?

3. In this problem we will use `dpzplot` function to plot the poles and zeros of some rational functions.

(a) Use `dpzplot` to plot the poles and zeros for,

$$H(z) = \frac{z^2 - z}{z^2 + 3z + 2}$$

(b) Use `dpzplot` to plot the poles and zeros which satisfy the following difference equation

$$y[n] + y[n-1] + 0.5y[n-2] = x[n]$$

(c) Use `dpzplot` to plot the poles and zeros which satisfy the following difference equation

$$y[n] - 1.25y[n-1] + 0.75y[n-2] - 0.125y[n-3] = x[n] + 0.5x[n-1]$$

4. In this problem we will examine the effect of coefficient quantization on different filter structures. Consider the $8^{th}$ order elliptic filter we designed in the example earlier. Store in the vectors `a` and `b` the coefficients for the eighth-order elliptic bandpass filter. Use `[H,w] = freqz(b,a,4096)` to compute 4096 samples of the frequency response of the filter for $0 \leq \omega \leq \pi$. Plot the `log` magnitude in decibels (dB) of `H` versus `w/pi` by executing:

```
>> plot(w/pi,20*log10(abs(H)));
>> axis([0 1 -80 10]);
```

Use `axis` to zoom in on the passband.

(a) Use `filter` to compute 4096 samples of the impulse response of the filter, storing the result in the vector `h`. Plot the first 200 samples of the impulse response.

(b) Set `M=max(abs([b a]))` and use `quant` to quantize the coefficients in `b` and `a` to 16 bits storing the results in `a16` and `b16`. What is the maximum amount any of the coefficients is changed by quantization? Use `freqz` to plot the frequency response magnitude of the quantized filter. Do you see any difference?

(c) Use the function `dpzlot` to generate a pole-zero plot for the filter described by the coefficients `a16` and `b16`.

(d) Repeat parts (a) and (b) quantizing `a` and `b` to 12 bits and store the results in `a12` and `b12`. Based on the pole-zero plots is the frequency response you have plotted both causal and stable? Why or why not?

(e) Use `filter` to compute 4096 samples of the impulse response of the filter described by `a12` and `b12`. Is the filter impulse response you have plotted stable? Why or why not?

5. In this problem you will examine the sensitivity of a cascade of four second-order direct form subsections to coefficient quantization. You will also compare this sensitivity to that of the direct form implementation you examined in the previous problem.

(a) The MATLAB function `[bc, ac] = df2cf(b,a)`, which is listed below, transforms the filter described by `a` and `b` into a cascade of second-order subsections. Each row of `ac` and `bc` contains the coefficients for one of the second-order subsections. Use `df2cf` with the original unquantized coefficients in `b` and `a` to create the cascade system parameters and store them in `bc` and `ac`.

```
function [bc, ac] = df2cf(b,a)
%[bc, ac] = df2cf(b,a)
% convert form direct from to a cacade of
% second order subsections.

N= length(b)-1;

z = roots(b);
p = roots(a);

for k=1:N/2
   bc(k,:) = poly(z(2*k-1:2*k));
   ac(k,:) = poly(p(2*k-1:2*k));
end
bc(1,:) = b(1)*bc(1,:);
```

(b) Use successive calls to `filter` with each row of `ac` and `bc` to generate 4096 samples of the impulse response of the cascade system; store the result in `hc`. Plot the first 200 samples of `hc` and compare this result to the plot of `h` (in problem 4) to verify that you have implemented the filter correctly. If you have, then `max(hc-h)` should be roughly `3e-13`.

(c) Use `quant` to quantize the coefficients in `bc` and `ac` to 16 bits and store the results in `bcq16` and `acq16`. Repeat part (b) to generate 4096 samples of the impulse response of the quantized cascade system and store the result in `hc16`.

(d) Use `freqz(hc16,1,4096)` to generate plots of the frequency response magnitude for the cascade system and zoom in on the pass-band. Do you see any difference compared to Problem 3. Also use `dpzplot` to make a pole-zero plot for each of the second -order subsections described by the rows of `bcq16` and `acq16`. Are each of the causal second order subsections stable? How do the poles and zeros of the second-order subsections correspond to the pols and zeros of the overall cascade system? How does the magnitude response of the 16-bit quantized cascade filter compare to that of the 16-bit quantized direct form implementation?

(e) Repeat parts (b)-(e) quantizing `bc` and `ac` to 12 bits. Comment on the new results.

## Deliverables

- Email your code, figures, calculation and answers as a *.pdf* or *.doc* file to **ece311lab.uiuc@gmail.com**. Be sure to name your document in the form- **ECE311Lab4_firstname_lastname.doc/pdf**.
- Late reports will reduce the grade by 20% per day.
- Make sure to present a clear and concise report having figures labeled and centered.
- **Reminder: Homework is due on 07/12/2011**