Chapter 3 MATLAB Frequency Response Example

A couple years ago one student asked if I could put together some of the MATLAB commands I used in obtaining the discrete-time $G(z)$ using the integration rules, and for finding the frequency response (magnitude and phase). I'll even show a little more than that. So here it is...

1. System under Consideration. Let's consider a second-order system with damping ratio $\zeta = 0.1$ and natural frequency $f_n = 5$ Hz. The low damping ratio will give us a resonant peak in the frequency response (as well as much overshoot in the step response if we were interested in that). We know that the continuous-time transfer function will be

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{1}$$

Note that this $G(s)$ has a DC gain of 1.

1.1 Continuous System. First we'll construct the continuous-time system $G(s)$ in MATLAB (one way to do it):

```
>> zeta = 0.1;  % Define damping ratio
>> fn = 5;  % Define natural frequency (Hz)
>> wn = 2*pi*fn;  % Convert to rad/s
>> numGc = [0 0 wn^2];  % Create numerator of G(s)
>> denGc = [1 2*zeta*wn wn^2];  % Create denominator of G(s)
>> Gc = tf(numGc,denGc)  % Create transfer function G(s); call it Gc

Transfer function:
        987
-------------------
s^2 + 6.283 s + 987
```

1.2 Discrete System. Next we'll find the *trapezoidal* discretization; that one is supported by MATLAB. But before that we have to select a sampling frequency. An old "rule of thumb" is to sample 6–20 times faster than the natural frequency, so let's choose $f_s = 50$ Hz ($T = 0.02$ sec).

```
>> fs = 50;  % Sampling frequency (Hz)
>> T = 1/fs;  % Sampling period (sec)
>> Gtrap = c2d(Gc,T,'tustin')  % Find trapezoidal (Tustin) discretization

Transfer function:
0.08497 z^2 + 0.1699 z + 0.08497
--------------------------------
     z^2 - 1.552 z + 0.8918

Sampling time: 0.02
```

You might be interested in the *poles and zeros* of the $G(z)$; the `zpk()` function helps a little:

```
>> zpk(Gtrap)

Zero/pole/gain:
```

```
    0.084971 (z+1)^2
-----------------------
(z^2  - 1.552z + 0.8918)

Sampling time: 0.02
```

So this $G(z)$ has two zeros at $z = -1$, but we *still* don't know where the poles are! The `zpk` function won't factor a complex denominator (or numerator)! I have no idea why not, but what you must do is extract the numerator and denominator using the `tfdata()` function, then use fundtion `roots()` to find the roots (poles). Like this:

```
>> [numtrap,dentrap] = tfdata(Gtrap,'v')  % The 'v' MUST be present or you get a cell array

numtrap = 0.0850    0.1699    0.0850
dentrap = 1.0000   -1.5519    0.8918

>> poles = roots(dentrap)  % Find the roots of the denominator (poles)

poles = 0.7760 + 0.5382i
        0.7760 - 0.5382i
```

You don't really have to find the poles in this case, but that's how.

2. Frequency Response. Let's find the frequency response (mag, phase) at 75% of the natural frequency, which is $f = 3.75$ Hz. The magnitude $|G|$ is the steady-state amplitude ratio, while the phase $\angle G$ is the phase shift. MATLAB returns magnitude as a "pure" number (not dB), and returns phase in degrees. We'll convert magnitude to dB, and we'll do this for both the original continuous system $G(s)$ and the discrete system $G(z)$.

2.1 Continuous system. Here we work with $G(s)$:

```
>> f = 3.75;  % Frequency of interest (Hz)
>> w = 2*pi*f;  % Convert to rad/s
>> [magc,phasec] = bode(Gc,w)  % Find the frequency response at f = 3.75 Hz

magc = 2.1622  % Magnitude of G(s) as a pure number, somewhat greater than 1 (DC value)
phasec = -18.9246  % Phase lag of G(s) is around 19 degrees

>> magcdB = 20*log10(magc)  % Remember that log10 is the base 10 log function

magcdB = 6.6978  % Magnitude of G(s) is around 7 dB (DC value is 0 dB)
```

2.2 Discrete system. Here we work with $G(z)$:

```
>> [magtrap,phasetrap] = bode(Gtrap,w)  % Same function, but G(z) = Gtrap

magtrap = 2.2563  % Both the magnitude and phase are
phasetrap = -20.1731  % pretty close to G(s)

>> magtrapdB = 20*log10(magtrap)

magtrapdB = 7.0681  % Magnitude of G(z) in dB...pretty close
```

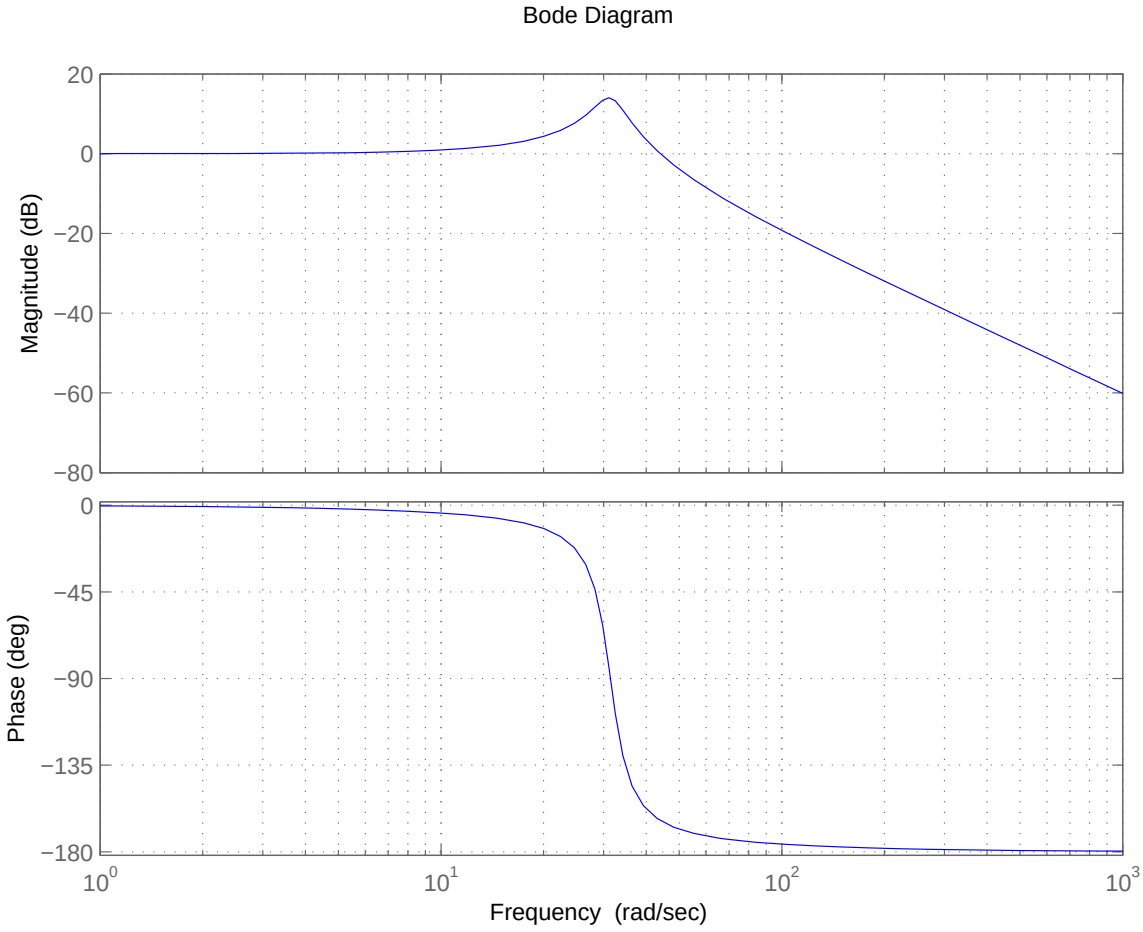
So that's how to perform the discretization and find the frequency response at a single frequency point.

However, we usually want to make Bode plots of the frequency response data. This can be done with the same `bode` function in MATLAB.

The simplest way is to let MATLAB select the frequency range (we'll use $G(s)$ here):

```
>> bode(Gc); grid;  % I like to put a grid on the plot
```

This will produce the plot shown below.



This plot is nice (note that the magnitude is plotted in dB) but if *you* wish to select the frequency range, you must pass a vector of frequencies to the `bode` function. The vector of frequencies should be formed using the `logspace()` function to get logarithmic spacing. The `logspace()` function accepts powers of 10 to determine this range, as well as a number of points. There's also the issue of Hz *vs* rad/s (Problem 3c).

To construct a frequency range of $1 \leq f \leq 1000$ Hz with 200 points (just as an example), then convert it to rad/sec and pass it to `bode()` you would do the following (I'm not going to show the plot):

```
>> f = logspace(0,3,200);  % Construct vector of frequencies (Hz) from 10^0 -> 10^3 w/200 pts
>> w = 2*pi*f;  % Convert to rad/s
>> bode(Gtrap,w);  % Find Bode plots of G(z)...this might look a little strange!
```

3. Conclusion. So there is an example of the MATLAB functions used to find and plot the frequency response characteristics of both a continuous system $G(z)$ and its discrete equivalent $G(z)$. Hope this helps!

NOTE: If you use the `bode()` function with returned arguments, like

```
>> [mag,phase] = bode(Gtrap,w);
```

you will need to remove the "singleton" extra dimension from returned arrays `mag` and `phase`. Like this:

```
>> mag = squeeze(mag);  % Remove singleton dimension
>> phase = squeeze(phase);  % Here, too...
```

This didn't used to be necessary, but that's progress.