

Chapter 3 HW Hints

Problem 1. Here you're given a *lead network*—often used in control systems to improve the transient response—which adds *approximately* 60° of phase angle at $\omega_1 = 3$ rad/s. The lead network transfer function is

$$H(s) = \frac{s + 1}{0.1s + 1}$$

Note that the DC gain of $H(s) = 1$. Use $T = 0.25$ seconds. You must do the two rectangular integration rules and the pole-zero mapping “manually,” but you can use the MATLAB `c2d()` function to find trapezoidal and prewarped (the `c2d` function does not do *pole-zero mapping* (`'method'='matched'`) in the way that I've shown (it doesn't handle zeros the same way)):

```
>> sysd = c2d(sysc,T,'tustin')      % Get the trapezoidal (Tustin) discretization
>> sysd = c2d(sysc,T,'prewarp',w); % Get the prewarped discretization (w = prewarp freq)
```

(a) I advise you to arrange your $H(z)$ transfer functions in “positive powers of z ” form, with a numerator coefficient factored out so both numerator and denominator polynomials are monic. The phase angle of the continuous system at 3 rad/sec is 54.87° (you should show this). Some of my phase angle results are:

- (i) Forward rectangular rule: check the stability!
- (ii) Backward rectangular rule: $\phi_F = 38.92^\circ$.
- (iv) Prewarped trapezoidal is perfect, as expected.
- (v) Pole-zero mapping: $\phi_{PZ} \approx 47^\circ$.

(b) You can use the MATLAB `bode(sys,w)` function to draw the Bode plots. You should prepare frequency vector w using the `logspace(w1,w2,N)` function with $N=100$ points. If you want more control over the plots, you can invoke the function with returned parameters like `[mag,phase] = bode(sys,w)` and then plot using `semilogx(w,mag)`. Note that `mag` and `phase` will be returned as “pathological” 3-D arrays—you should correct this by invoking:

```
>> mag = squeeze(mag) to “collapse” them into vectors (same for phase).
```

Also, the `mag` will be returned as a “pure” magnitude, not in decibels (dB). Plotting in dB is traditional; you can convert from magnitude to dB by `>> magdB = 20*log10(mag)`. You might wish to plot multiple cases in a single command, or use the `hold on` function to overlay successive plots. If you use different linestyles in a single plot command, you might want to use the `legend('label1','label2',...)` to indicate the different plots.

We will go through some of this in class.

- (c) I have not yet looked at developing a “parabolic rule”...update when I have a result.

Problem 2. This is similar to the previous problem, except with a second-order system. As before, you can use MATLAB to do the *trapezoidal* and *prewarped* cases; however the *pole-zero mapping* should be done manually.

(a) Express your discrete transfer functions in the same form as Problem 1—both numerator and denominator polynomials monic in positive power form, with a numerator coefficient factored out.

(b) You can use MATLAB `bode()` here as well. Note that I specified the frequency range in Hz, but MATLAB will require it in rad/sec.

(c) Here use MATLAB `step()` with your $G(s)$ and $G(z)$ in MATLAB LTI system format.

Problem 3. This problem is again similar to the first two; the time response in (d) is similar to Problem 4 of the Chapter 2 homework. Your notch filter should be effective at filtering out the 60-Hz frequency component.

We will do a demo of a notch filter in the lab on Monday, February 22 (along with aliasing demos).