

### Plotting with *Matlab* and Bode Plots

You previously had to find the transfer function for the circuit defined in problem 14.20 of the textbook. The transfer function is

$$H(s) = \frac{V_o(s)}{V_i(s)} = \frac{s/RC}{s^2 + s/R_{eq}C + 1/LC} \quad (1)$$

where

$$R_{eq} = \frac{RR_l}{R + R_l}. \quad (2)$$

Equation (1) can be rearranged slightly to obtain

$$H(s) = \frac{R_l}{R + R_l} \frac{s/R_{eq}C}{s^2 + s/R_{eq}C + 1/LC} = \frac{R_l}{R + R_l} \frac{s\beta}{s^2 + s\beta + \omega_o^2}. \quad (3)$$

The last form of the equation is in the standard form for a bandpass filter where the bandwidth  $\beta$  is  $1/R_{eq}C$ , the center frequency  $\omega_0$  is  $\sqrt{1/LC}$ , and the maximum magnitude of the transfer function is  $R_l/(R + R_l)$ . Letting  $s = j\omega$ , the magnitude of the transfer function can be written

$$|H(j\omega)| = \frac{R_l}{R + R_l} \frac{\omega\beta}{([\omega_o^2 - \omega^2]^2 + \omega^2\beta^2)^{\frac{1}{2}}}. \quad (4)$$

Bode plots are plots of the magnitude and phase of the transfer function versus frequency. However, the horizontal (frequency) axis always uses a log scale and the vertical axis always uses a linear scale. This statement is a bit misleading in that when we plot the magnitude of the transfer function, we actually plot 20 times the log base 10 of the magnitude (so in some sense we are using a logarithmic vertical scale too).

Assume we want to use *matlab* to make a Bode plot of the magnitude of the transfer function given above. In a moment we will show a much easier way to generate this plot, but, to illustrate how one can plot an arbitrary function in *matlab*, we will first do things the hard way.

One can obtain a plot in *matlab* using the `plot()` command which, in one of its more basic forms, takes two arguments: an  $x$  and a  $y$  vector which give the  $x$  and  $y$  values to be plotted. Assume we want to plot the transfer function over a range of frequencies going from one to 100 kilo radians per second ( $10^3 \leq \omega \leq 10^5$  rad/s). We can create a vector of values for  $\omega$  using a command such as

```
omega = linspace(1e3,1e5,101);
```

---

File: bode-n-such.tex

Thus, the vector `omega` would contain values

```
[1000 2000 3000 .... 100000]
```

There is nothing inherently wrong with this, but for a Bode plot the horizontal axis will be logarithmic. Thus it is usually better to use logarithmic spacing between the values of  $\omega$ . We can get values which are equally spaced in the logarithmic sense by using the command `logspace()` instead of `linspace()`. Thus, the following command

```
omega = logspace(3,5,101);
```

sets `omega` equal to the vector

```
[1000 1047.1 1096.5 ... 91200 95500 100000]
```

These values are equally spaced in the logarithmic sense in that

$$\frac{\omega_{n+1}}{\omega_n} = \text{constant}. \quad (5)$$

Here that constant happens to be 1.0471 (this is  $100^{1/100}$ ).

The vector `omega` is our “ $x$ ” vector of values. Now we need to construct the “ $y$ ” vector. To do that, we need to select values for the circuit elements. Assume they are given by  $R = 200 \Omega$ ,  $R_l = 1000 \Omega$ ,  $C = 2.5 \mu\text{F}$ , and  $L = 1.0 \text{ mH}$ . Thus the transfer function is

$$H(s) = \frac{2000s}{s^2 + 2400s + 4 \times 10^8} \quad (6)$$

giving a bandwidth of 2400 rad/s and a center frequency of 20,000 rad/s. The magnitude of  $H(j\omega)$  is given by

$$|H(j\omega)| = \frac{2000\omega}{\left([4 \times 10^8 - \omega^2]^2 + \omega^2 2400^2\right)^{\frac{1}{2}}}. \quad (7)$$

To generate these values in *matlab* we have to do things like square the values of the  $\omega$ . Keep in mind that multiplication of vectors does not make sense. We can take the cross product of vectors or the dot product, but we cannot multiply them in the usual sense. It does make sense to multiple vectors by a scalar. Here we have  $2000\omega$  in the numerator. *Matlab* allows us to carry out that operation by simply writing

```
2000*omega
```

The result of this operation is simply the scaling of each value of the original  $\omega$  vector by the value 2000. What do we do about a term like  $\omega^2$ ? We have to keep in mind what this is implying. We want to square each individual value of the frequency. In other words, we want to obtain a vector which contains the square of the first value of the  $\omega$  vector, and then the square of the second value of the  $\omega$  vector, and so on. *Matlab* allows us to multiply vectors on a term-by-term basis (note that this operation is not something defined in a standard math class and you should not confuse it with standard vector manipulation—this is merely a programming convenience). Assume vectors  $a$  and  $b$  are given by

$$a = [a_1 \ a_2 \ \cdots \ a_n], \quad (8)$$

$$b = [b_1 \ b_2 \ \cdots \ b_n]. \quad (9)$$

In *matlab* we cannot write  $a*b$ , but we can write  $a.*b$ . Adding a dot ( $.$ ) before most operators means to do the operation on a term-by-term basis. Thus, if we issue the command

```
c = a.*b
```

then the vector  $c$  will have the values

$$c = [a_1b_1 \ a_2b_2 \ \cdots \ a_nb_n]. \quad (10)$$

Returning to the transfer function, we can get  $\omega^2$  by writing  $\omega.*\omega$ . However, this term-by-term type of operation even works for the exponentiation operator (the “ $\wedge$ ” operator) so we could also write  $\omega.^2$ .

To get the log base 10 in *matlab*, one uses the command  $\log_{10}()$ . We now can generate the necessary  $y$  values for the plot. They can be obtained using commands such as the following (where we’ve repeated the creation of the  $\omega$  vector here for the sake of completeness):

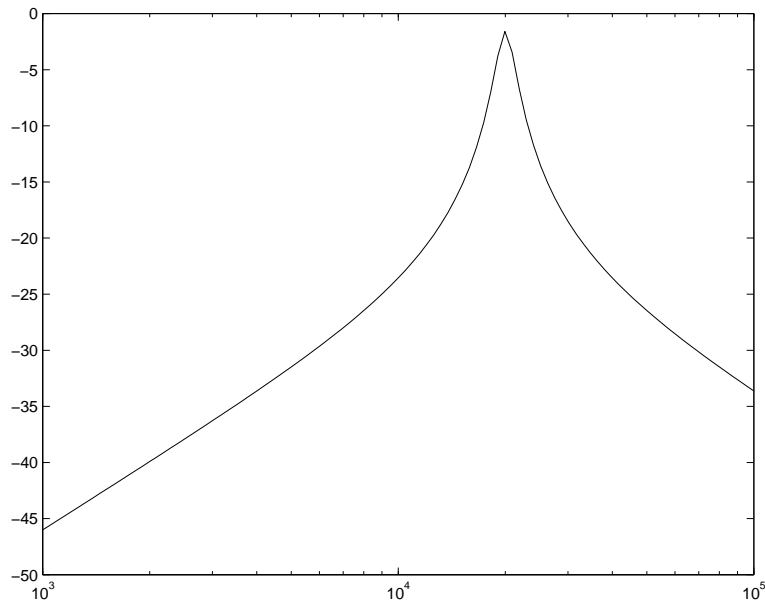
```
omega = logspace(3,5,101);
beta=2400;
omegaZero=20000;
hmag=20*log10( ...
    2000*omega ./ ...
    sqrt((omegaZero^2 - omega.^2).^2 + beta^2*omega.^2))
```

This could now be plotted using the command  $\text{plot}(\omega, \text{hmag})$ , but this generates a linear scale for both the horizontal and vertical axes while Bode plots require a logarithmic scale

for the horizontal axis. The command which generates a logarithmic  $x$  axis and linear  $y$  axis is `semilogx()`. Thus, the desired plot can be obtained using

```
semilog(omega, hmag)
```

The plot generated by this is shown below.



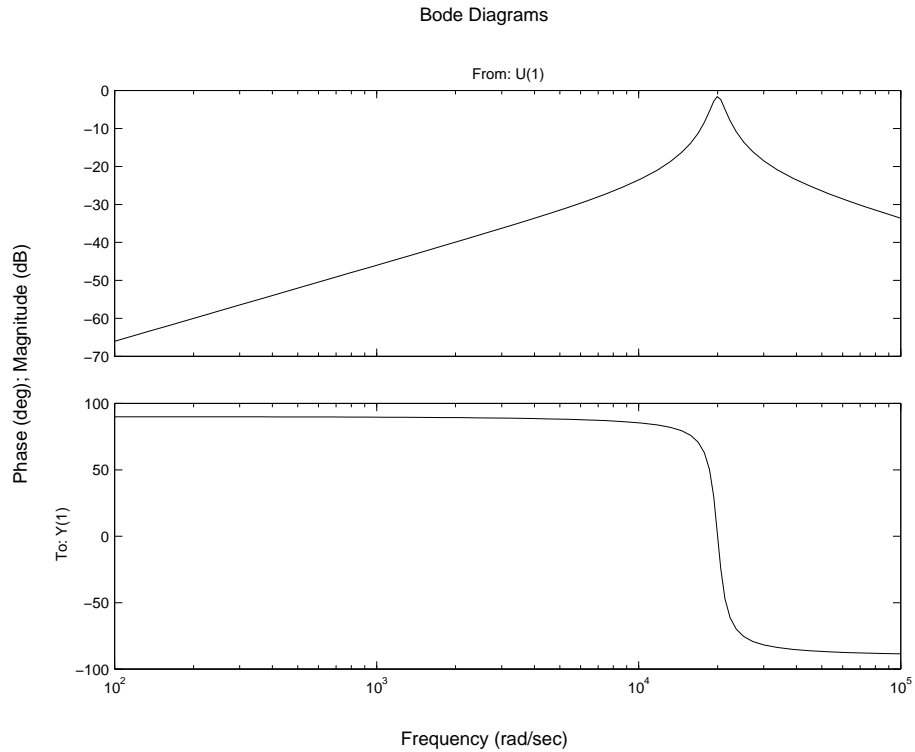
Note that the peak is at 20,000 rad/s as it should be.

### Letting *Matlab* Do All the Work

That was a fair amount of work to generate the magnitude of the transfer function. We might be interested in the phase as well and that would require a similar effort. Fortunately *matlab* makes the generation of Bode plots incredibly easy. One can simply issue the command `bode(num, den)` where `num` and `den` are vectors containing the coefficients for all the powers of  $s$  in the numerator and denominator of the transfer function. For the example being considered here, the command would be

```
bode([2000 0],[1 2400 4e8])
```

This produces the following plot:



Here we get the magnitude and the phase, as well as labeling, “for free.”

Note that `bode ( )` will accept various arguments. Of particular interest, when dealing with state-space problems, is giving the **A**, **B**, **C**, and **D** matrices as arguments. Assuming these matrices have been properly defined, the following will generate the appropriate Bode plots for the system

```
bode(a,b,c,d)
```