Laplace and Z-Transform Analysis and Design Using Matlab

Harold L. Broberg
Indiana University - Purdue University, Fort Wayne

I. INTRODUCTION

The Electrical Engineering Technology (EET) curriculum at IPFW requires an understanding of Laplace and z-transforms and their use in circuit analysis and design. This is emphasized in junior level courses [1, 2] which focus on analog and digital circuits and systems. Senior level electives, including courses in automatic control systems and digital filters [3, 4] build on this knowledge. The computer program used in these courses must enable students to easily calculate and plot time and frequency responses using Laplace and z-transform equations. The program must also be easily programmable, allow simple conversion between domains, and enable students to mathematically predictor verify Pspice circuit simulations. It must also be useable in as many courses as possible, for continuity, and be widely used in industry.

The Professional version of Matlab meets these requirements at a relatively low cost for a networked version. It also has many available Toolboxes, and a simple programming language. Simulink (a graphical simulation tool), the Control System Toolbox, and the Signal Processing Toolbox were also purchased with Matlab and are used extensively in these courses. The Fuzzy Logic Toolbox, another of the many available toolboxes, is used in another senior level course. The networked version of Matlab, with these toolboxes, is also used by Mechanical Engineering Technology and Electrical Engineering students, and these departments shared the cost of the software. The Matlab "notebook", which comes with the professional version of Matlab, was used for this document. This "notebook" enables the user to embed Matlab programs, solutions, and graphs into a Microsoft Word 6.0 document and is an excellent tool for instructors.

A linear algebra course is not required in EET, but students are familiar with many of the basic concepts by their junior year. This familiarity, along with the primarily vector (or list) characteristics of most Matlab commands enable students to learn to use the vector/matrix based format with little difficulty. Matlab is powerful, has an extensive help facility, and is useful for introducing and using mathematical methods in continuous and discrete circuit analysis, control systems, and filters. It has an extensive system simulation capability, using the optional Simulink, and its power and graphical capabilities are appreciated by students. The examples described below focus on time and frequency response using the Laplace and z-domain and use a small subset of the Matlab commands. The Bookware Companion series [5] contains many excellent Matlab applications.

II. LAPLACE CIRCUIT ANALYSIS AND DESIGN

The use of Matlab to augment Pspice for the mathematical portions of circuit analysis was well received by the students. Student use of Matlab provides a better understanding of the mathematics involved

[1]The MathWorks, Inc. 24 Prime Park Way, Natick, MA 01760

in the design and functioning of circuits in both the time and the frequency domains, and of the complex relationships between the time and frequency domains. Students learn more because they can graphically analyze equations and simulate circuits using Laplace and z-transforms, and quickly see the result of changing a parameter in an equation representing a circuit or system.

One of the requirements in Laplace analysis is partial fraction expansion of a Laplace polynomial representing the output of a circuit. Inversion of the partial fraction expansion gives the time domain expression for the output. The algebraic manipulation of the Laplace transformed circuit and the subsequent partial fraction calculations provide needed practice for students. There is, however, a limit to the time that can be devoted to this algebra practice. A simple method of partial fraction expansion, to check the students work or do the work for them, allows the students to learn more about how mathematical expressions represent the circuits operation.

In Matlab, the ***residue(N,D)*** command provides a partial fraction expansion of a polynomial with numerator, N, and denominator, *D*. For instance, consider a simple series RLC circuit with the output taken across the capacitor. After transformation of this circuit into the Laplace domain and a few calculations, Equation [1] is found to represent the Laplace polynomial of the output of the circuit, where $V_0(S)$ is the output across the capacitor and VI(S) is the generalized input waveform.

$$V_0(S) = \frac{1/LC}{S^2 + (R/L)*S + (1/LC)} * V_I(S) \qquad [1]$$

A common method of using this circuit for illustration of time and frequency domain effects is to discuss the effect of changing the resistance value, or damping, of the circuit. With a unit step input, zero initial conditions, an inductance of 10mH, and capacitance of 1 μF, we find Equation [2]:

$$V_0(S) = \frac{10^8}{S^2 + 100R*S + 10^8} * \frac{1}{S} * \frac{10^8}{S^3 + 100R*S^2 + 10^8 S} \qquad [2]$$

A few calculations show that over-, critical, and under-damping occur for values of R = 500, 200, and 100 ohms respectively.

A. MATLAB FOR PARTIAL FRACTION EXPANSION: To use Matlab for partial fraction expansion, first enter the numerator and denominator polynomials. I the following example, the variable **num** is used for the numerator and the variable denl represents the denominator. The command to find the residues is then shown on the next line.

The Matlab commands for R = 500 (overdamped) are:

```
num = [1e8];    den1=[1 5e4 1e8 O] ;
[residues ,poles,k]=residue(num,den1)
```

The Matlab output for these commands is:

```
residues =  0.0455, -1.0455,  1.0000
poles =  1. 0e+004 *   -4.7913,   -0.2087,  0
```

The residues are the numerators of the partial fraction expansion and the poles are the denominators. From this, the partial fraction expansion is found to be as shown in Equation [3].

$$\frac{108}{S(S^2 + 5X10^4 S + 10'S)} = \frac{.0455}{S + 47913} - \frac{1.0455}{S + 2087} + \frac{1}{S} \qquad [3]$$

For R=200, **and R=100. The Matlab commands are the same except for the denominator polynomial. Finding these partial fraction expansions using Matlab yields Equations [4] and [5].**

R = 200 (critically damped)

```
residues =   -1,   -10000,  1
poles =  -10000,   -10000,  0
```

$$\frac{10^8}{S(S^2 + 2x10^4S + 10^8S)} = \frac{-S}{S + 10000} - \frac{10000}{S + 10000} + \frac{1}{S} \qquad [4]$$

R = 100 (underdamped)

```
residues = -0.5000 + 0.2887i,  -0.5000 - 0.2887i,  1.0000
poles =  1.0e+003 *  -5.0000 + 8.6603i,  -5.0000 - 8.6603i,    0
```

$$\frac{10^8}{S(S^2 + 1x10^4S + 10^8S)} = \frac{-.5 + j.2887}{S + 5000 - j8660} + \frac{-.5 - j.2887}{S + 5000 + 8660} + \frac{1}{S} \qquad [5]$$

B. MATLAB FOR TIME DOMAIN ANALYSIS: To observe the effect of changing the resistance on the time domain output of the circuit, the Matlab *step(N,D, t)* command provides a graph of the unit step response of the transfer function, Equation [1], with numerator, N, denominator, *D* and time axis *t*. Since a single graph is better to compare the three time domain responses of the circuits, the following commands can be used:

First determine the time axis and plot from O to .003 seconds with a 1 µsec interval.

```
t=O: 1e-6: .003;
```

Next calculate 3001 points for each of the three equations.

The numerator is the same for equations and the change in resistance is reflected in the coefficient of the middle term. The left side of the Matlab commands produce vectors (lists) of the time axis, *t*, they axis (variables selected are *yover*, *ycrit* and *yunder* to refer to over, critical, and underdamping). The x variable on the left side produces the state response of the transfer function which is not of interest here.

```
[yover, xl, t]=step([1e8] , [1 5e4 1e8],t) ;
[ycrit, x2, t]=step([1e8] , [1 2e4 1e8],t) ;
[yunder, x3, t]=step([1e8] , [1 le4 le8] ,t) ;
```

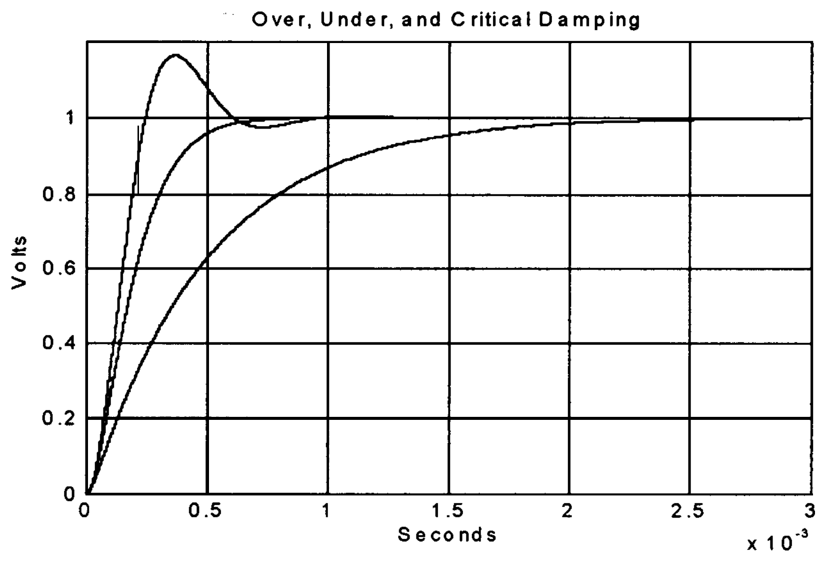
Next plot and label the graph for the three outputs:

The *plot* command allows the user to plot as many graphs as needed as long as each has a vector for the x-axis and one for the y-axis. The *grid* command places a grid on the plot and the *axis* command enables the user to control the axes boundaries. Here, the plot is from t=O to .003 and from y=O to 1.2. The *xlabel*, *ylabel* and *title* commands enable the user to label the graph.

```
plot (t, yover,t,ycrit,t,yunder) ,grid, axis( [0 .003 0 1.2])
xlabel ( ' Seconds ' ) , ylabel(' Volts ' ), title(' Over ,Under,and Critical Damping' )
```

The resulting graph, "Over, Under, and Critical Damping", shown below, provides a clear understanding of the effect of varying the resistance of the circuit on the circuit output. This mathematical approach, combined with Pspice simulations gives the student a better understanding of the effect of his/her algebraic manipulations.

C. MATLAB FOR FREQUENCY DOMAIN ANALYSIS: Bode analysis of the Laplace equations is just as straightforward as time domain response, and can provide an appreciation of the important relationship between the time and frequency domain. Using the same circuit with the same numerator and denominator functions shown above, the Matlab *bode(N, D)* command provides a graph of the frequency response of a transfer function with numerator, N, and denominator, *D*.



For Bode analysis, first generate 1000 points between 102 and 105 radians/see:

```
w=logspace(2, 5, 1000) ;
```

Next, calculate the magnitude and phase of the circuit with each of the values of R. The magnitude vectors of 1000 points each are called *magover, magcrit,* and *magunder* and the phase vectors of 1000 points each are called *phaseover, phasecrit* and *phaseunder.* Note that the same transfer functions are used.

```
[magoverrphaseover, w]=bode ( [1e8] , [1 5e4 1e8] ,w) ;
[magcrit, phasecrit, w]=bode ( [1e8] , [1 2e4 1e8] ,w) ;
[magunder,phaseunder, w]=bode ( [1e8] , [1 le4 1e8] ,w) ;
```

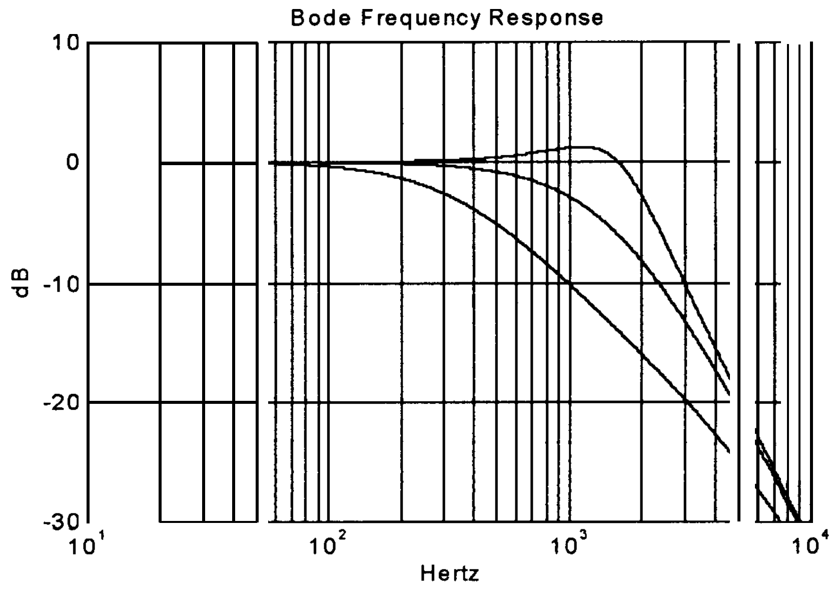
Then plot the magnitude, in dB, and the frequency, in Hertz, on a semilog axis. The *semilogx* command plots a logarithmic x-axis. the dB of the y-axis must be calculated for each graph. The *grid* and *axis* commands are the same as for the earlier graph. Plotting the phase response is also simple, but has not been illustrated here.

```
semilogx(w/(2*pi) ,20*log10 (magover) , w/ (2*pi) ,20*log10 (magcrit) ,
w/ (2*pi),20*log10(magunder)) ,grid, axis ( [0 le4 -30 10] )
xlabel ( ' Hertz ' ) ,ylabel ( ' dB' ) , title ( ' Bode Frequency Response ' )
```

Looking at the resulting graph, “Bode Frequency Response”, shown below, all of the circuits look like 2nd order low-pass filters in the frequency domain. Using some simplification, the underdamped circuit shows a resonance effect and has Chebychev type characteristics in the frequency domain, because of the resonance, which is called underdamping in circuits and controls. The critically damped circuit is relatively flat up to the cutoff frequency and is a Butterworth type low-pass filter. The overdamped circuit falls off slowly and is an example of a poor low-pass filter. A classroom discussion of the time and frequency response characteristics of this circuit is normal in most Laplace circuit analysis courses. If this is followed by an assignment where the students calculate and plot the time and frequency responses of a similar circuit, with a varying parameters, using Matlab, it provides them with a better understanding of the mathematical relationships.

III. Z-TRANSFORM ANALYSIS AND DESIGN

Like Laplace analysis, z-transform analysis and design is based on time and frequency domain concepts. Similar Matlab tools are available in the z domain to those shown above in the Laplace domain for finding and plotting time and frequency response. A useful example is conversion of a polynomial from the Laplace to the z-domain. This is a tedious algebraic procedure that contains many opportunities for error. The following routine enables Matlab to perform the conversion automatically. As an example we will convert the underdamped, R = 100, Laplace circuit transfer function used above to its z-transform using a zero-order hold with a sampling period of 10 μsec.



First, input the sampling period, T, of 10 μsec, which is a sampling rate of 100 KHz:

```
T=1e-5;
```

Next, write a brief program to convert from a Laplace to a z-transform polynomial. The first equation converts from transfer function to state-space form. The second converts the result from continuous state-space to discrete state-space. The third converts from discrete state-space to a z-transform transfer function. This program is used like a subroutine and the student does not need to know state-space concepts.

```
[A, B, C, D]=tf2ss(num,den3);
[ad, bd]=c2d(A,B,T);
[numz,denz]=ss2tf (ad, bd,C,D,1)
```

The Matlab result is:

```
numz =  0 4.8334 e-003 4. 6749e-003
denz =  1. 0000e+000 -1.8953 e+000 9. 0484e-001
```

The z-transform representation of this underdamped circuit is shown as Equation [6]

$$H(Z) = \frac{.004833Z + .004675}{Z^2 - 1.8953Z + .9048} \qquad [6]$$

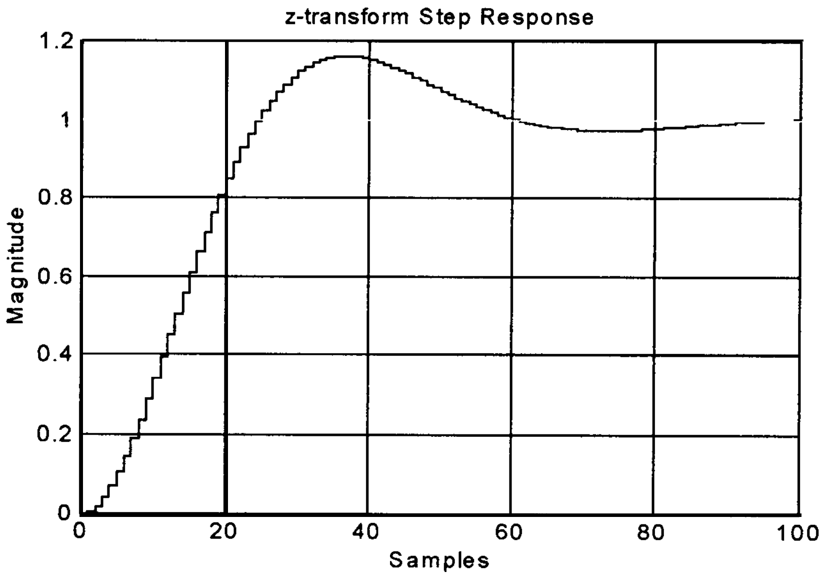
To plot the step response of this transfer function, the Matlab *dstep(N, D)* command is used. The result is a graph of the unit step response of the transfer function with numerator, N, and denominator, *D*. For a single plot, the following simple commands which also provide automatic graph scaling can be used.

```
dstep (numz, denz ) , grid
xlabel ( ' Samples' ) ,ylabel ( 'Magnitude' ) , title ( ' z-transform Step Response' )
```

The graph is shown below as "z-transform Step Response". The relatively slow sampling rate used here makes the discrete nature of the A-D and D-A process that produces the discrete time waveform readily apparent. With a shorter sample time (e.g. 1 μsec), the time response of this sampled-time system would be virtually identical to the continuous-time system from which it was derived.

Another Matlab command, *dbode(N, D)* provides a bode magnitude and phase plot of the discrete transfer function with numerator, N, and denominator, *D*. Bode plots of the discrete transfer function (z-domain) can be produced using the same Bode plotting techniques used earlier with the *dbode(N,D)* command used instead of the *bode(N,D)* command.

The ease of transfer between the Laplace and z-domains, and the readily available time and frequency domain plots, make the transition to z-domain functions smooth and relatively painless when compared to the algebraic nightmare of calculating 2nd and higher order z-transforms by hand. Additionally, when a z-domain transfer function is converted to a difference equation, a simple Matlab program (using a For... .End loop) can be used to plot the result of the difference equation implementation.



IV. CONCLUSION

The professional version of Matlab enables the student to easily plot time and frequency domain waveforms after determining the Laplace transfer function of a circuit. Conversion of Laplace polynomials to z-transform polynomials is also easily accomplished. This allows the instructor to concentrate on important relationships between the time and frequency domains and between continuous and discrete time circuits. Matlab is a widely used tool in industry and academia and is a standard in control system analysis and design. It is powerful and relatively inexpensive for academic institutions and is highly recommended for junior and senior EET students.

REFERENCES:

[1] Stanley, W. D.; “Transform Circuit Analysis for Engineering and Technology”, 2nd Edition, Prentice-Hall, 1989

[2] Strum, R. D.; and Kirk, D.E.; “First Principles of Discrete Systems and Digital Signal Processing”, Addison-Wesley, 1989

[3] Stefani, R. T.; Savant, C. J.; Shahian B.; Hostetter, G. H.; Design of Feedback Control Systems, 3rd Edition, Saunders College Publishing, 1994

[4] Ingle, V. K.; and Proakis J. G.; “Digital Signal Processing Laboratory using the ADSP- 2101 Microcomputer”, Prentice-Hall, 1991

[5] Bookware Companion Series, Robert D. Strum, Series Editor, PWS Publishing Company

HAROLD L. BROBERG: A retired Lt.Col of Marines, Hal received his P.E. in 1988, and his PhD (EE) in August, 1993. His dissertation involved improvements to the GOES weather satellite, which is used for all TV weather satellite photos. He is a consultant for ITT on weather satellite servos and uses Matlab extensively. He is Acting Chair of the Electrical Engineering Technology Department, President of the Anthony Wayne Chapter of the Indiana Society of Professional Engineers, and a senior member of IEEE.