EE -213
BASIC CIRCUIT ANALYSIS
LAB MANUAL

EE 213
Fall 2009

LABORATORY #1
INTRODUCTION TO MATLAB

INTRODUCTION

The purpose of this laboratory is to introduce you to Matlab and to illustrate some of its circuit analysis applications. Matlab is an interactive, high level, programming language for general scientific and technical computation. You will find that the language is structured so that matrix manipulations are especially easy, efficient, and reliable. Computations with Matlab are easier than programming in C.

To begin Matlab, select the Matlab program from a menu in your operating system, or by entering Matlab from the keyboard, You should see the Matlab prompt EDU> or >> which tells you that Matlab is waiting for you to enter a command. At this point you can type demo and observe demonstrations of Matlab as described on the screen that pops up.

Another possibility is to type help; then from the list that pops up on the screen, you can pick a more specific help, such as help elfun; this list is similar to the index of a textbook. From the next list that pops up on the screen you can pick a more specific help, such as help sin. In this manner you can wander, perhaps for hours, through Matlab commands.

Yet another approach is to just type out Matlab commands followed by commas and Matlab will print out the results that you have asked it to compute. An example of this is the Matlab script:

```
EDU>t=1:3:15,
        t =1     4     7    10    13
EDU>t'
        ans =
                 1
                 4
                 7
                10
                13

EDU>t*t'
        ans =335

EDU>t'*t
        ans =  1     4     7    10    13
                 4    16    28    40    52
                 7    28    49    70    91
                10    40    70   100   130
                13    52    91   130   169
EDU>t.*t
```

ans =1 16 49 100 169

The first command generates a row vector with values between 1 and 15 with components separated by 3. The second command generates a column vector that is the transpose of the first vector. The third and fourth commands multiply these two vectors using the usual matrix multiplication rules; the final command multiplies the vector t times itself component by component. If you type the same commands followed by a semicolon, Matlab will not print the results; they are just stored by Matlab. The semicolon is very useful when the vectors have many components; the comma is very useful for debugging Matlab when the vectors are short.

In this laboratory, you will be asked to practice plotting some typical functions that arise in circuit analysis and to analyze some problems that you solved in EE 211.

PLOTTING WITH MATLAB

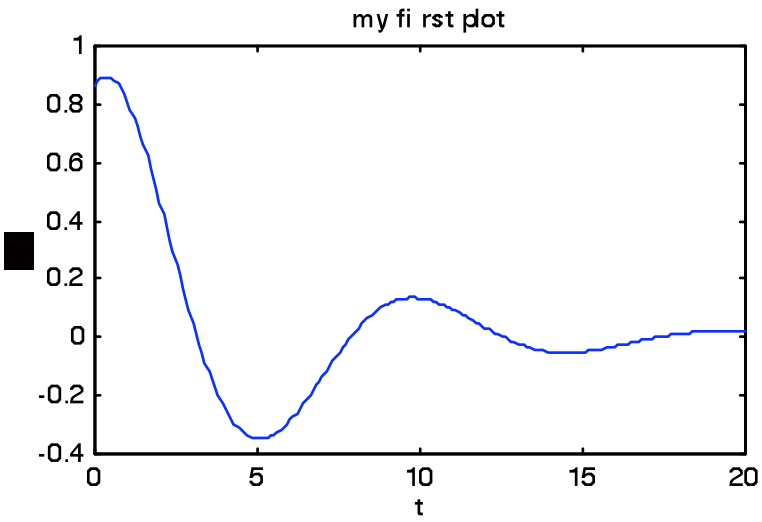
We first illustrate the plotting capabilities of Matlab with the following example**.** Consider plotting the voltage

$$v(t) = e^{-t/5}\sin\left(2t/3 + \pi/3\right)$$

for times between zero and twenty seconds with 0.1 second increments. The Matlab script to accomplish this is:

```
EDU>a=-1/5;w=2/3;phi=pi/3;t=0:.1:20;
EDU>v=exp(a*t).*sin(w*t+phi);
EDU>plot(t,v),ylabel('v(t)'),xlabel('t'),title('my first plot')
```

The resulting plot is shown below.



The first line of the Matlab script generates a set of parameters for the problem; the second command generates the vector v that is to be plotted; the last line plots and labels the function. By using the arrow keys on the keyboard you can scroll through previous commands and change the parameters and plot the function again.

Plot and label each of the functions below on the indicated time interval; choose the time increment so that the resulting plot is smooth:

$$a(t) = 40\cos(4t/5 + 2\pi/7), \quad -3 < t < 3;$$
$$b(t) = a(t) + 30\sin(t/5 + \pi/6), \quad -3 < t < 3;$$
$$c(t) = 25\cos(2(t-2)^2), \quad -5 < t < 5$$
$$d(t) = \exp(t/5)\cos(t/6), \quad -1 < t < 35$$
$$e(t) = a(t)c(t), \quad -1 < t < 5$$
$$f(t) = \begin{cases} 14\sin(4\pi t + \pi/7), & t < 2 \\ 15\sin(2\pi t - \pi/5), & t > 2 \end{cases}, \quad -4 < t < 25$$
$$g(t) = \begin{cases} e^{-|2t+5|}\sin(4\pi t), & t < 5 \\ e^{3|t|}\cos(2\pi^2 t), & t > 5 \end{cases}, \quad -5 < t < 25$$
$$h(t) = f(t) + g(t), \quad -4 < t < 25.$$

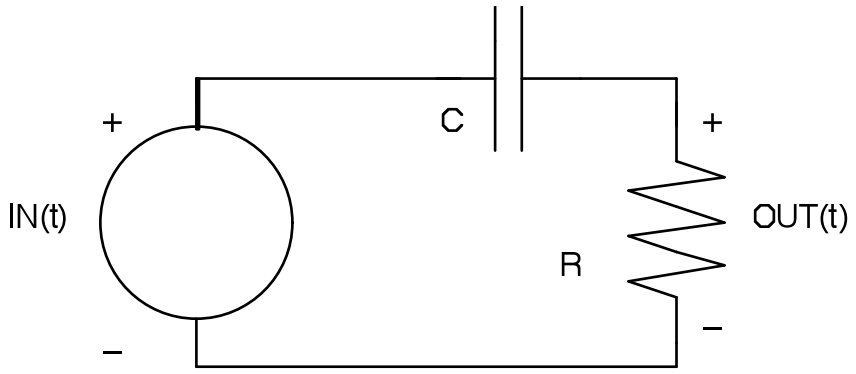
Use Matlab to plot the signals a(t) and b(t) on the same set of axes with different types of lines; carefully label your plot using the legend command.

Use the subplot command to plot the signal $y(t) = \text{real}(Xe^{st})$ and illustrate the effect of the phasor X and the complex frequency s on the signal y(t).

THE STEP RESPONSE OF CIRCUITS

In EE 211 the step responses of several simple circuits were found; in this laboratory we use the Matlab command step(.,.) to find and plot the step responses of some circuits.

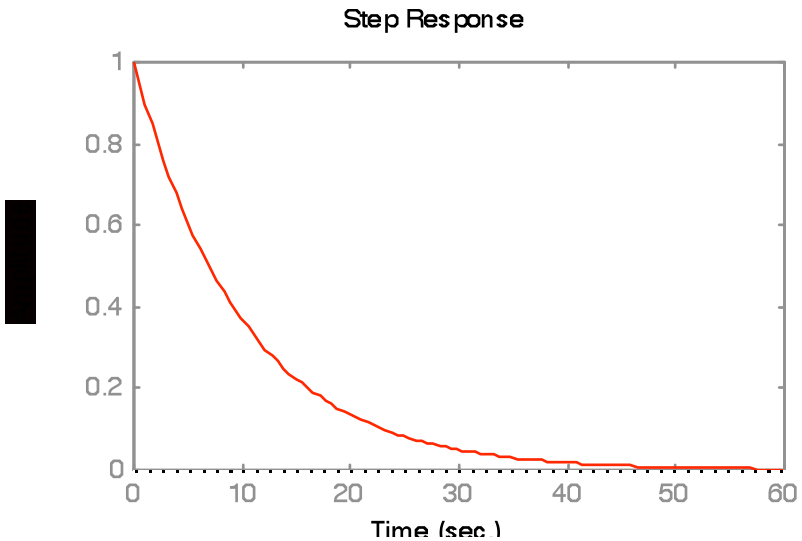
Matlab uses the transfer function to specify the circuit. The transfer function is the ratio of the response of the circuit to the input of the circuit when the input to the circuit is a complex exponential at the complex frequency s. For the Matlab step command, the transfer function must be expressed as the ratio of polynomials in the complex frequency s; Matlab characterizes the circuit by the row vector of coefficients, in descending order, of these polynomials. For example for the circuit below, the transfer function is $H(s) = sRC/(sRC+1)$ the numerator polynomial is then [RC 0] and the denominator polynomial is [RC 1].



The step response of this circuit can be plotted with the Matlab script below:

EDU>RC=10;num=[RC 0];den=[RC 1];step(num,den)

The Matlab plot of the step response is shown below.



The step response plotted by Matlab makes sense physically. If the circuit is at rest, which it is for negative times then all the voltages and currents in the circuit are zero. At time zero the input voltage jumps to unity; the voltage across the capacitor does not change instantaneously and it remains at zero. By applying Kirchoff"'s voltage law around the circuit, one finds the initial output voltage is unity. As time increases, charge accumulates on the plates of the capacitor and the voltage across the capacitor increases towards unity; Kirchoff's voltage law then dictates that the output voltage decreases toward zero.

Use this Matlab script to determine the effect of the product RC on the step response of this circuit. Discuss your results and justify you answer with a physical argument.

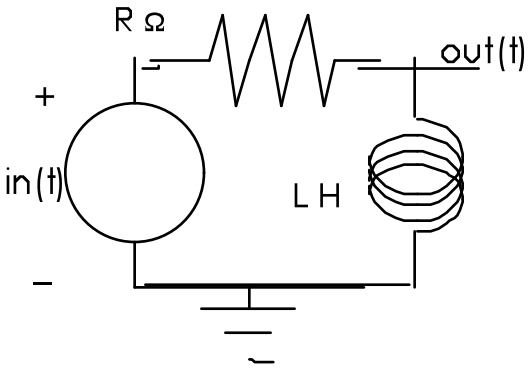
For each of the circuits below do the following:

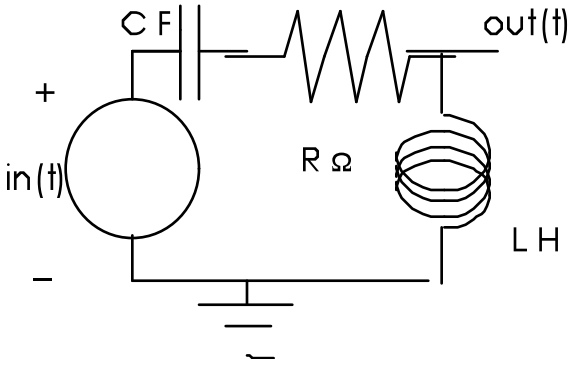
(1) Without any analysis, determine the initial and final values of the output current when the input voltage is a unit step.

(2) Use EE 211 techniques to solve for the output current when the input voltage is a unit step and all the element values are unity.

(3) Find the transfer function of the circuit; express your answer in terms of a ratio of polynomials in s; consider general values for the circuit elements.

(4) Use Matlab to plot the unit step response of each circuit below; determine the effect of the element values on the unit step response. Discuss your results.





EE 213
Spring 2008

LABORATORY #2
MATLAB SOLUTION OF LINEAR EQUATIONS

1. Introduction

In this laboratory Matlab techniques to solve sets of simultaneous linear equations are introduced; in circuit analysis these equations are usually node voltage, or mesh current, equations. There are two basic types of solutions; numeric solutions are applicable when all the coefficients are known constants; symbolic solutions are applicable when some, or all, of the coefficients do not have numeric values. Both techniques make use of the inv(.) command to compute the inverse of a square matrix. In Matlab it is better, though, to use the backslash \ command for solving linear equations.

2. Numeric Solutions of Linear Equations

Consider the resistive circuit of Figure 1 with the indicated node voltages and loop currents.

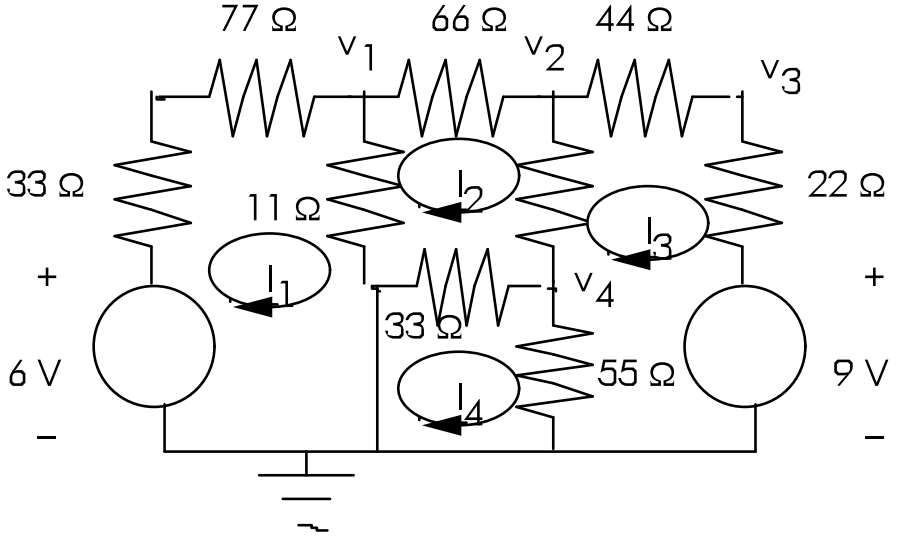


Figure 1 Numeric Restive Circuit

(a) (prelab) Use Kirchoff's current law to write four simultaneous equations for the node voltages v_1, v_2, v_3, and v_4.

(b) Express the node voltage equations in the form $\mathbf{b} = \mathbf{A}\mathbf{v}$, where $\mathbf{v}$ is a four dimensional column vector containing the unknown node voltages, $\mathbf{b}$ is a four dimensional column vector and $\mathbf{A}$ is a square, four by four, matrix. Use a Matlab symbolic script to verify that your matrix equation is correct.

(c) Use Matlab to numerically solve the equations of (b) for the node voltages: $\mathbf{v} = \mathbf{A}^{-1}\mathbf{b}$, or in Matlab notation, v=A\b.

(d) (prelab) Use Kirchoff's voltage law to write four simultaneous equations for the loop currents i_1, i_2, i_3 and i_4.

(e) Express the loop current equations in the form $\mathbf{A}\mathbf{i} = \mathbf{b}$, where $\mathbf{i}$ is a four dimensional column vector containing the unknown loop currents, $\mathbf{b}$ is a four dimensional column vector and $\mathbf{A}$ is a square, four by four, matrix. Use a Matlab symbolic script to verify that your matrix equation is correct.

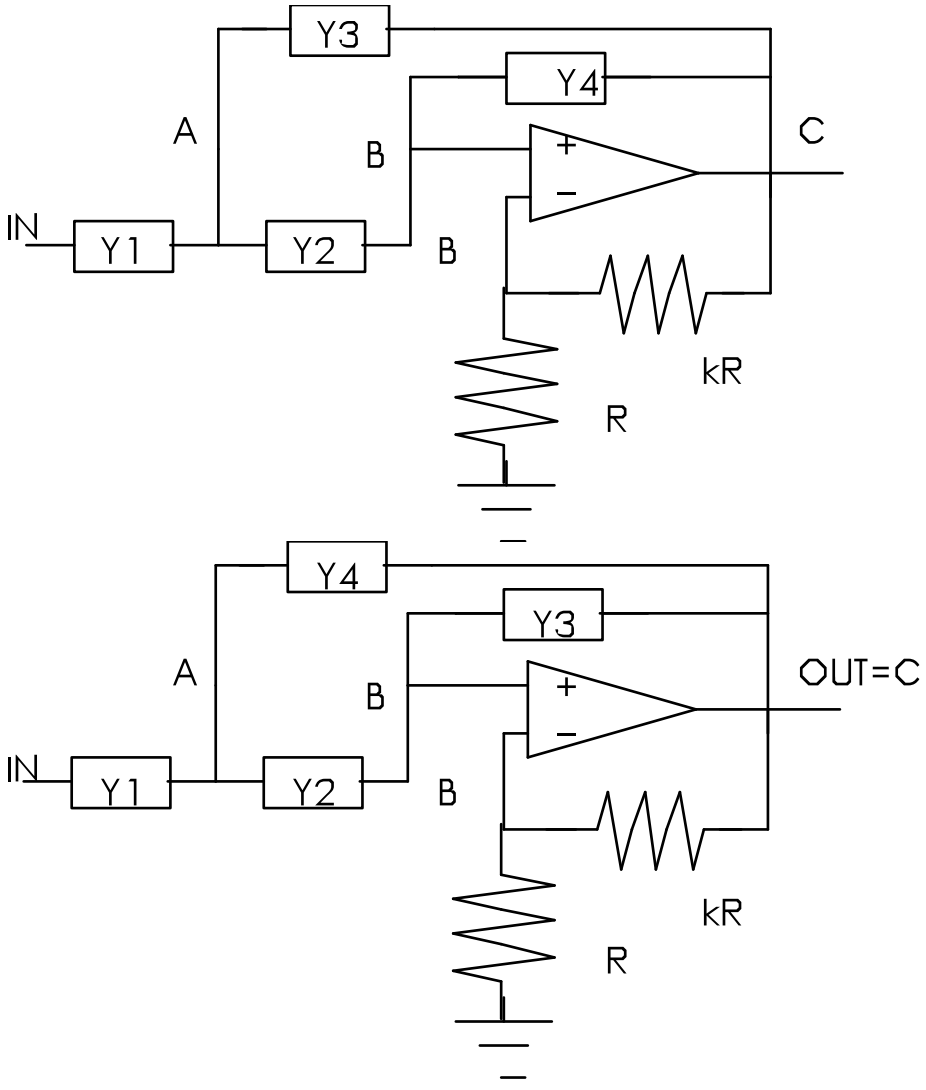
(f) Use Matlab to solve the equations of (e) for the unknown loop currents; $\mathbf{i} = \mathbf{A}^{-1}\mathbf{b}$.

c. Symbolic Matlab Solutions of Linear Equations

In Matlab, symbolic quantities can be declared with the **`syms`** command.

Consider the circuits of Figure 2 with the indicated node voltages; the branches that are not resistors are characterized by their admittances. For each circuit,

write a set of node voltage equations that will determine the node voltages; write the equations as a set of linear equations with an appropriate matrix and vector of unknowns. Verify the correctness of each matrix and vector with a symbolic Matlab script. Solve for the unknown node voltages with a symbolic Matlab script.



EE 213

Fall 2008

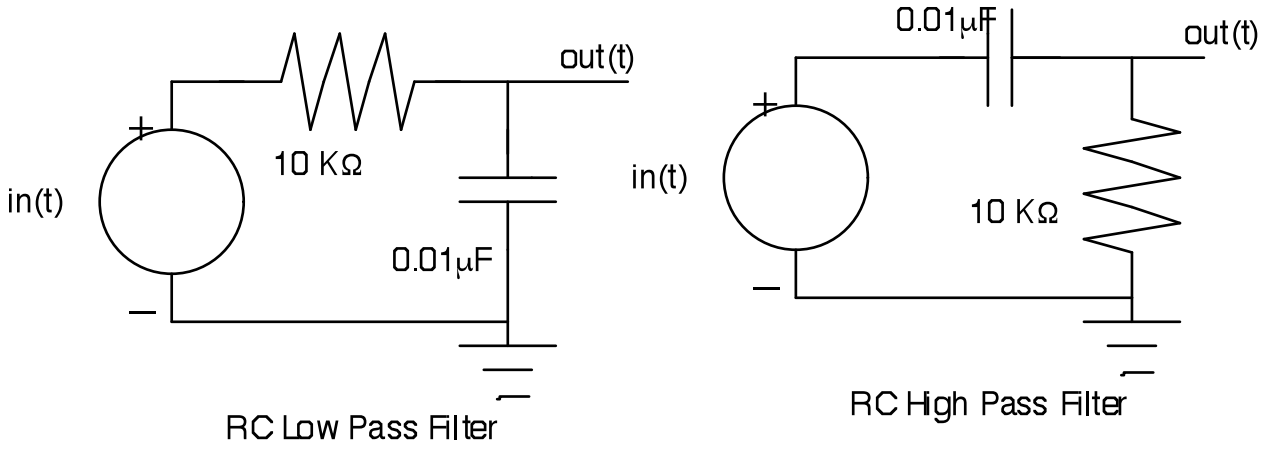
LABORATORY # 3

RC CIRCUIT RESPONSES

Objective:

The objective of this laboratory is to examine the RC low pass and high pass filters in both the time domain and the frequency domain; and simulate them in Matlab. The magnitude and phase of the frequency response and the response to a square wave are the frequency and time domain responses considered.

In this lab we consider both the RC lowpass and high pass filters shown below. The voltage input to the circuit is a sinusoid when determining the frequency domain characteristics of the circuits: the voltage input is a square wave when determining the time domain characteristics of the circuits.



(1) Determine the transfer functions of each of these circuits.

(2) Use Matlab to plot the magnitude and phase of the transfer function of each of these circuits. Discuss your results.

(3) One cycle of a square wave of amplitude a and frequency w is given by $in(t) = a\big(u(t) - u(t - \pi/\omega)\big)$. Use symbolic Matlab to verify the Laplace transform of one cycle of the square wave. Use numeric Matlab to plot the magnitude and phase of the Laplace transform when the complex frequency is purely imaginary. Discuss your results.

(4) Use symbolic Matlab to find the response of each of these filters to one cycle of a square wave; use numeric Matlab to plot the response of each of these filters to one cycle of a square wave. Discuss the effect of the amplitude and frequency of the square wave on the response of each of these filters.

EE 213

Spring 2008

LABORATORY # 4

Numeric Matlab for Laplace Transforms

Objective:

The objective of this laboratory is to introduce some numeric Matlab commands that are useful with Laplace transforms. The numeric commands are useful when all the circuit element values are specified as numbers; the only non-numeric parameter in the input and output signals is the time or the complex frequency; these commands calculate and, or plot the answers as functions at specific times or complex frequencies..

Numerical Commands:

TRANSFER FUNCTION REPRESENTATION:

A general form for many Laplace transforms is as the ratio of polynomials in the Laplace variable s:

$$X(s) = \frac{n_k s^k + n_{k-1} s^{k-1} + n_{k-2} s^{k-2} + ... + n_1 s^1 + n_0 s^0}{d_l s^l + d_{l-1} s^{l-1} + d_{l-2} s^{l-2} + ... + d_1 s^1 + d_0 s^0};$$

such a Laplace transform is referred to as a rational Laplace transform..

The variable k is the degree of the numerator; l is the degree of the denominator. In many problems, termed proper, or bottom heavy, rational Laplace transforms, the degree of the numerator is strictly larger than the degree of the denominator: l> k; otherwise, the rational Laplace transform is termed improper, or not bottom heavy.

For Matlab numerical commands, the rational polynomial form of the Laplace transform is represented by two row vectors of coefficients of the polynomials in

descending powers of the complex frequency: $num = [n_k\ n_{k-1}\ n_{k-2} \cdots n_1\ n_0]$ and $den = [d_l\ d_{l-1}\ d_{l-2} \cdots d_1\ d_0]$. In many Matlab commands, such as zp2tf or tf2zp the letters tf are used as a memory aid for this representation.

POLE AND ZERO REPRESENTATION:

The pole and zero representation of a Laplace transform is: $X(s) = g\frac{(s-z_k)(s-z_{k-1})(s-z_{k-2})\cdots(s-z_2)(s-z_1)}{(s-p_l)(s-p_{l-1})(s-p_{l-2})\cdots(s-p_2)(s-p_1)}$. The roots of the numerator, $z_1, z_2, \ldots, z_{k-2}, z_{k-1}, z_k$, are referred to as the zeros of the Laplace transform; the roots of the denominator, $p_1, p_2, \ldots, p_{l-2}, p_{l-1}, p_l$, are referred to as the poles of the Laplace transform, and the ratio of the two highest power coefficients is the gain $g = n_k / d_l$.

For Matlab numerical commands, this pole zero form of the Laplace transform is represented by two column vectors of roots of the numerator and denominator and the gain: $z = [z_k; z_{k-1}; z_{k-2}; \ldots; z_1]$, $den = [p_l; p_{l-1}; p_{l-2}; \ldots; p_1]$ and g=[g]. In many Matlab commands, such as zp2tf, tf2zp , or pzmap the letters zp are used as a memory aid for this representation. Matlab determines the appropriate representation pz or tf by whether the input vectors are two rows or two columns and a scalar for this and many other commands.

CONVERSION BETWEEN POLE ZERO AND TRANSFER FUNCTION REPRESENTATIONS:

To convert from the rational polynomial, or transfer function, representation of a Laplace transform to the pole zero representation of the same transform one uses the Matlab command [z,p,g]=tf2zp(num,den), with the numerator polynomial num and the denominator polynomial den as inputs. To plot the poles and zeros on the complex plane

the appropriate Matlab command is pzmap(num,den); the zeros are represented by 0's and the poles are represented by X's.

Use Matlab to determine, and plot, the poles, zeros, and gain of a Laplace transforms with the numerator and denominator polynomials generated with each of the numeric Matlab commands below:

```
>> [n1,d1]=butter(10,5,'s')
>> [n2,d2]=ellip(8,1,25,[1 3],'stop','s');pzmap(n,d)
```

.

Plot the poles and zeros of each of these Laplace transforms. Use the Matlab command zp2tf to verify that Matlab calculates the proper numerator and denominator polynomials for a given set of poles zeros and gain.

PARTIAL FRACTION REPRESENTATION

The partial fraction representation of a rational Laplace transform is: $X(s) = \sum_{j=1}^{l} \frac{r_j}{s - p_j} + \sum_{j=0}^{k-l} d_j s^j$. The roots of the denominator $p_1, p_2, \ldots, p_{l-2}, p_{l-1}, p_l$ are still referred to as the poles of the Laplace transform, the numerators of the fractions $r_1, r_2, \ldots, r_{l-2}, r_{l-1}, r_l$, are referred to as the residues and the polynomial $\sum_{j=0}^{k-l} d_j s^j$ is referred to as the direct term. When a rational Laplace transform is bottom heavy, the direct term is zero.

For Matlab numerical commands, the residue form of the Laplace transform is represented by two column vectors of the same dimension containing the residues and the poles and a row vector representing the direct term polynomial. Corresponding terms in the column vectors are the residue and pole of one term in the expansion; the row vector that represents the direct term polynomial has the highest power coefficient first. The direct term is an empty vector when the Laplace transform is proper.

To convert from the rational polynomial, or transfer function, representation of a Laplace transform to the residue representation of the same transform one uses the Matlab command [r,p,d]=residue(num,den), with the numerator polynomial num and the denominator polynomial den as the arguments of the command.

Use the Matlab to determine the residue representation of each of the rational Laplace transforms with the numerator and denominator polynomials generated with each of the numeric Matlab commands below:

```
>> [n1,d1]=butter(10,5,'s')
>> [n2,d2]=ellip(8,1,25,[1 3],'stop','s');pzmap(n,d).
```

The Matlab command residue also converts from the residue representation to the transfer function representation if the input to the command consists of the column vector of residues, the column vector of poles and the row vector representing the coefficients of the direct term; the vectors returned are the row vectors of coefficients of the numerator and denominator of the transfer function. The correct form of the command is [num,den]=residue(r,p,d). Use this command to verify that the residue calculations made above are correct.

TIME DOMAIN SIGNALS

If a rational Laplace transform is proper and represented with the transfer function representation, the inverse Laplace transform is easily plotted with the Matlab command impulse; this command will also return the vectors used to plot the inverse transform. If the Laplace transform is not proper, impulse plots, or computes, only the inverse Laplace transform with a null direct term.

For Matlab numerical commands when the numerator and denominator have been defined and correspond to a proper Laplace transform, the inverse Laplace transform is plotted by the command impulse(num,den); the command [h,t]=impulse(num,den) will return the vectors used to plot the impulse response. The argument of the impulse command can also be the column vectors of poles and zeros and the gain of the Laplace

transform rather than the row vectors representing the numerator and denominator of the Laplace transform.

Another operation that one frequently encounters in inverting Laplace transforms is the need to multiply two rational Laplace transfer functions. If both Laplace transforms are in the transfer function representation, the numerator of the product is the product of the two numerators and the denominator is the product of the two denominators. The coefficients of the product of two polynomials can be computed with the command prod=conv(poly1,poly2). If the polynomials a(s) and b(s)are represented by the column vectors of roots pA and pB, respectively, and the cofficients of the highest power cA and cB, respectively, and the polynomial c(s)=a(s)b(s) is represented by the column vector pC and coefficient cC. The column vector pC is easily calculated with the Matlab command pC=[pA;pB] and the coefficient cC=cA+cB.

Use the Matlab to plot the inverse Laplace transforms of each of the Laplace transforms with the numerator and denominator polynomials generated with each of the numeric Matlab commands below:

```
>> [n1,d1]=butter(10,5,'s')
>> [n2,d2]=ellip(8,1,25,[1 3],'stop','s');pzmap(n,d)
```

.

Also plot the inverse transform of the product of these two functions.

.

EE 213
Spring 2007

LABORATORY # 5
CONVOLUTION LABORATORY

OBJECTIVE:

The objective of this laboratory is to first investigate Matlab techniques to evaluate convolutions both numerically, and when appropriate, symbolically. Properties of convolution are then investigated using Matlab: these properties are linearity, time invariance, and association. Finally the results of convolution with special signals such as impulses, unit steps, and complex exponentials are found.

PRE LABORATORY:

Use convolution to directly evaluate the following convolutions:

$$a(t) = e^{-3t}u(t), \quad b(t) = e^{4t}u(-t), \quad c(t) = \Pi(t/4), \quad d(t) = t\Pi(t/2)$$

$$w(t) = \text{conv}\big(a(t), b(t)\big), \quad x(t) = \text{conv}\big(a(t), c(t)\big),$$

$$y(t) = \text{conv}\big(b(t), d(t)\big), \quad z(t) = \text{conv}\big(b(t), c(t)\big),$$

After plotting your solutions, discuss your results; pay particular attention to the smoothness of signals, the regions where signals are non-zero, and the widths of your solutions.

LABORATORY:

NUMERICAL CONVOLUTION:

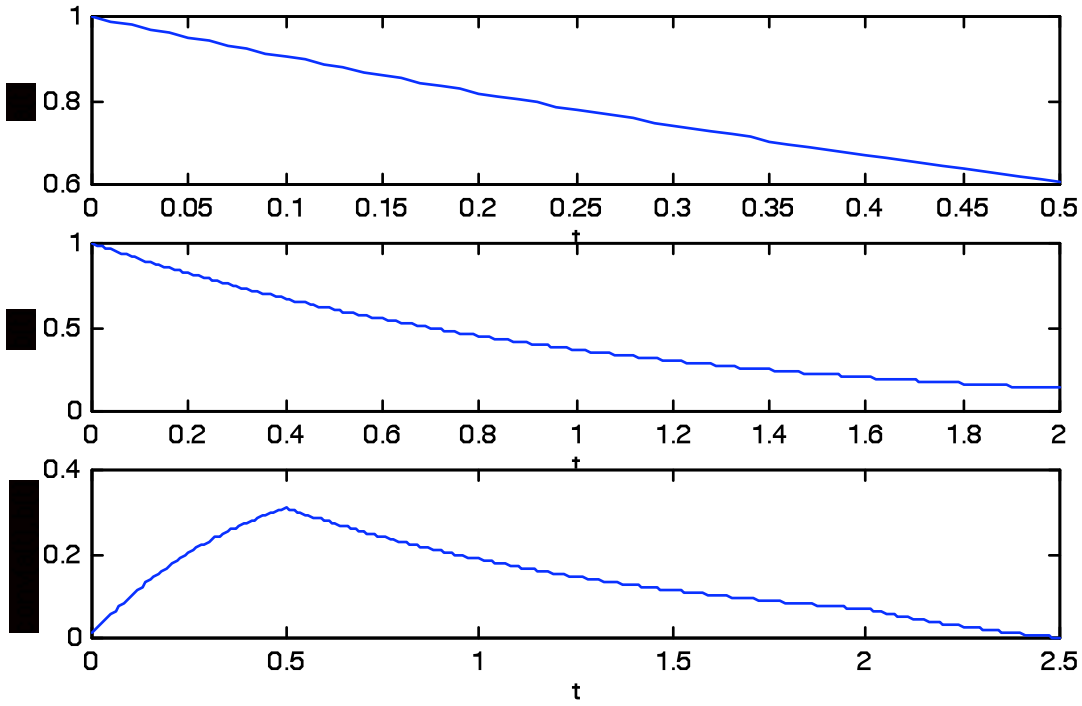
The Matlab script below will approximate and plot the signal c(t) that is the convolution of a signal a(t) with the signal b(t). The signal a(t) is non zero only for times in the interval from goa to stopa; the values of a(t) in this interval every dt time units are first stored in the vector a. Similarly, the signal b(t) is non zero only for times in the interval from gob to stopb; the values of a(t) in this interval every dt time units are first stored in the vector b. The convolution is non zero only for times between goa + gob and stopa+stopb; The values every dt time units are stored in the vector c.

MATLAB SCRIPT:

```
EDU>dt=.01;goa=0;stopa=1/2;gob=0;stopb=2;
        ta=goa:dt:stopa;a=exp(-ta);tb=gob:dt:stopb;b=exp(-tb);
EDU>tc=goa+gob:dt:stopa+stopb;c=dt*conv(a,b);
EDU>subplot(311),plot(ta,a),xlabel('t'),ylabel('a(t)'),
EDU>subplot(312),plot(tb,b),xlabel('t'),ylabel('b(t)'),
EDU>subplot(313),plot(tc,c),xlabel('t'),ylabel('conv(a(t),b(t)'),
```

MATLAB PLOTS:

The Matlab plots that result from the above script are shown below.



Use a variation of this script to plot the solutions to each of the pre-lab problems. Investigate and then discuss the effects of the choice of the sampling interval, dt, as well as the start and stop times for signals that are not zero outside a finite interval.

Use a variation of this script to plot each of the outputs of an RC low-pass filter with unity time constant to an input equal to each of the pre-lab signals.

SYMBOLIC CONVOLUTION:

The Matlab symbolic toolbox does not contain a symbolic convolution command; however the laplace(.) and ilaplace(.) commands can be used to evaluate a convolution when each of the signals to be convolved are one-sided. A one-sided signal is zero for all

negative times. The Matlab symbolic script below convolves the two one sided exponentials of the numeric example when the stop times are infinite.

Matlab Script:

```
EDU>syms t s
EDU>a=exp(-t)*'Heaviside(t)';
        A=laplace(a,t,s);b=ilaplace(A*A,s,t)*'Heaviside(t)'
```

The Matlab response to this script is:

```
        b =t*exp(-t)*Heaviside(t).
```

Use a variation of the script above to verify the solutions of each of the pre-lab problems. Use a variation of the script to find the response of the RC high-pass filter with unity time constant to an input equal to each of the pre-lab signals. For those signals that can not be evaluated with Laplace transforms, use the Matlab int(.,.,.) command to evaluate the convolutions.

CONVOLUTION PROPERTIES:

The three Matlab properties to be considered are association, linearity and time invariance. The formal statements are:

ASSOCIATION:

$$conv\big(a(t),b(t)\big) = conv\big(b(t),a(t)\big)$$

LINEARITY:

$$conv\big(a(t) + \alpha b(t),c(t)\big) = conv\big(a(t),c(t)\big) + \alpha conv\big(b(t),c(t)\big)$$

TIME INVARIANCE:

If $c(t) = conv\big(a(t),b(t)\big)$, then $c(t-\alpha) = conv\big(a(t-\alpha),b(t)\big)$.

Verify each of these properties numerically for at least three cases; then verify each of these properties symbolically for at least three cases.

Discuss how time invariance can be used to compute symbolic convolutions for signals that are not one sided but are zero for large negative times. Illustrate your procedure with at least four examples.

Three special signals: the impulse, the unit step, and the complex exponential, are easy to convolve. The convolutions are:

$$conv\big(any(t),\delta(t)\big) = any(t)$$

$$conv\big(any(t),u(t)\big) = \int_{-\infty}^{t} any(x)dx$$

$$conv\left(any(t), Xe^{st}\right) = Xe^{st} \int_{-\infty}^{\infty} e^{-sx} any(x)dx.$$

If possible verify, numerically and symbolically, each of the convolutions with the three special signals for each of the special signals of the prelab.

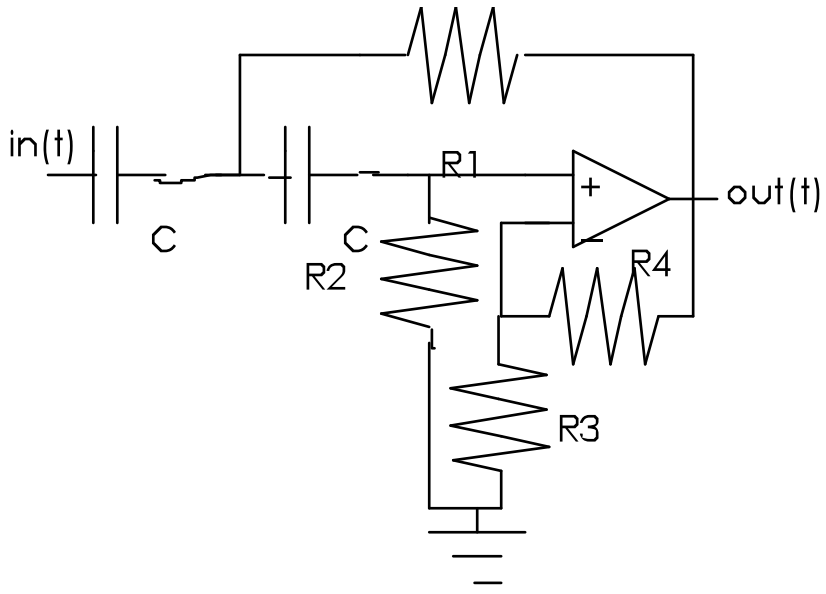
EE 213
Spring 2008

LABORATORY#6
FIRST BIQUAD LABORATORY

OBJECTIVE:

The objective of this laboratory is to experimentally determine the step response, the natural response, and the frequency response of the biquad circuit below. The transfer function, the step response and the natural response of the circuit are obtained with Matlab; the conditions for over damped, under damped, and unstable cases are investigated in the pre-lab. The experimental results are then compared with the Matlab results obtained in the pre-lab.

The biquad circuit to be investigated is shown below.



PRELABORATORY:

(1) Use physical arguments to determine the very high, and very low, frequency behavior of the circuit above; determine the type of filter realized by this biquad. Use physical arguments to determine the initial value and final value for the step response of this circuit.

(2) Use Matlab to determine the transfer function of the biquad circuit. Check that your answers are consistent with your results to (1). Determine the effect of doubling and halving all the impedances in this biquad. Determine the effect of halving all the capacitors.

(3) Determine how to pick the resistance values so that the transfer function of the biquad is proportional to the transfer function of a series RLC circuit with an output across one of the circuit elements. Use Matlab to determine a condition for an over damped circuit, and for an under damped circuit. Simplify your results when R3=R4. Use the Matlab command pzmap(n,d) to verify your result. Determine a condition to insure the circuit has no poles in the right half plane.

(4) Plot the step response of the biquad for an under damped, critically damped, and over damped case. Check that your answers are consistent with your results to (1).

(5) Plot the magnitude and phase of the transfer function of the biquad for an under damped, critically damped, and over damped case.

(6) Find the changes to your solutions to (4) and (5) when all the capacitor values are doubled; and again when they are halved. Discuss your result.

(7) Use Matlab to find the response of this circuit to the input u(-t); this is referred to as the natural response of the circuit. Use a physical argument to find the initial and final value of the natural response; use this to check your Matlab solution.

EXPERIMENTAL

Construct the biquad for an under damped, critically damped, and over damped case; use reasonable values for the circuit elements. Measure the square wave response of the biquad in each case; compare your experimental results with the step response and natural response found with Matlab. Measure the frequency response of the biquad in each case. Carefully compare your results with the results obtained from Matlab; discuss your results.

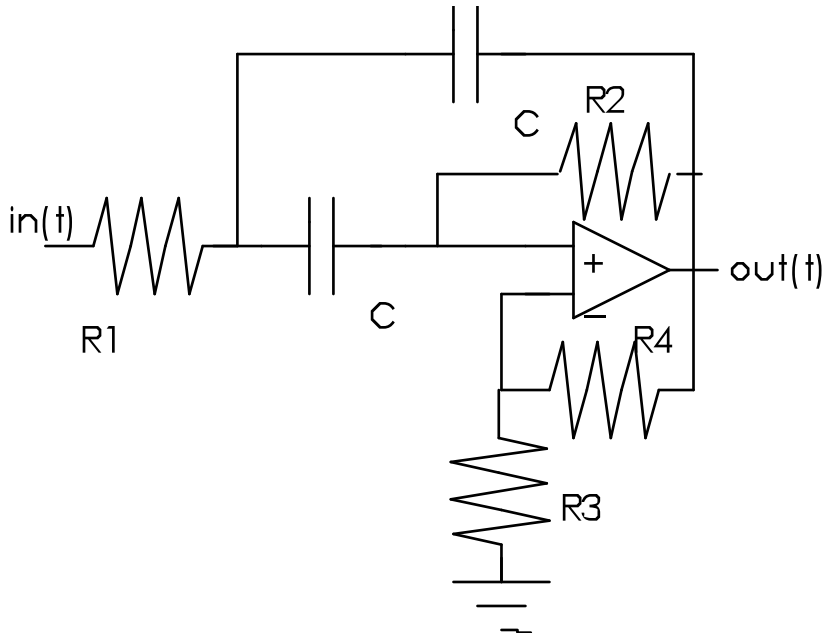
EE 213
Spring 2008

LABORATORY#7
SECOND BIQUAD LABORATORY

OBJECTIVE:

The objective of this laboratory is to experimentally determine the step response, the natural response, and the frequency response of the biquad circuit below. The transfer function, the step response and the natural response of the circuit are obtained with Matlab; the conditions for over damped, under damped, and unstable cases are investigated in the pre-lab. The experimental results are then compared with the Matlab results obtained in the pre-lab.

The biquad circuit to be investigated is shown below.



PRELABORATORY:

(1) Use physical arguments to determine the very high, and very low, frequency behavior of the circuit above; determine the type of filter realized by this biquad. Use

physical arguments to determine the initial value and final value for the step response of this circuit.

(2) Use Matlab to determine the transfer function of the biquad circuit. Check that your answers are consistent with your results to (1). Determine the effect of doubling and halving all the impedances in this biquad.

(3) Determine how to pick the resistance values so that the transfer function of the biquad is proportional to the transfer function of a series RLC circuit with an output across one of the circuit elements. Use Matlab to determine a condition for an over damped circuit, and for an under damped circuit. Simplify your results when $R_1 = R_2 = R_4$. Use the Matlab command pzmap(n,d) to verify your result. Determine a condition to insure the circuit has no poles in the right half plane.

(4) Plot the step response of the biquad for an under damped, critically damped, and over damped case. Check that your answers are consistent with your results to (1).

(5) Plot the magnitude and phase of the transfer function of the biquad for an under damped, critically damped, and over damped case.

(6) Find the changes to your solutions to (4) and (5) when all the capacitor values are doubled; and again when they are halved. Discuss your result.

(7) Use Matlab to find the response of this circuit to the input u(-t); this is referred to as the natural response of the circuit. Use a physical argument to find the initial and final value of the natural response; use this to check your Matlab solution.

EXPERIMENTAL

Construct the biquad for an under damped, critically damped, and over damped case; use reasonable values for the circuit elements. Measure the square wave response of the biquad in each case; compare your experimental results with the step response and natural response found with Matlab. Measure the frequency response of the biquad in each case. Carefully compare your results with the results obtained from Matlab; discuss your results.

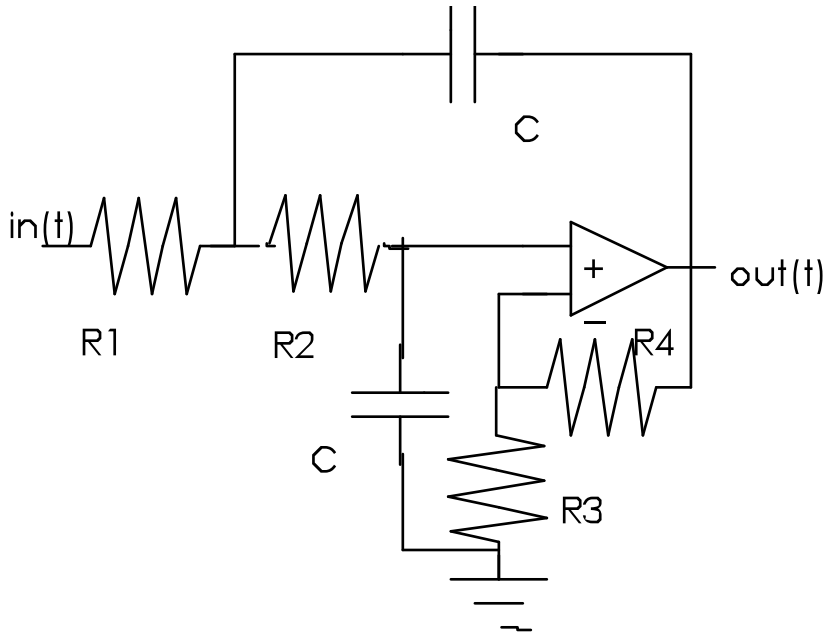
EE 213
Spring 2008

LABORATORY#8
THIRD BIQUAD LABORATORY

OBJECTIVE:

The objective of this laboratory is to experimentally determine the step response, the natural response, and the frequency response of the biquad circuit below. The transfer function, the step response and the natural response of the circuit are obtained with Matlab; the conditions for over-damped, under-damped, and unstable cases are investigated in the pre-lab. The experimental results are then compared with the Matlab results obtained in the pre-lab.

The biquad circuit to be investigated is shown below.



PRELABORATORY:

(1) Use physical arguments to determine the very high, and very low, frequency behavior of the circuit above; determine the type of filter realized by this biquad. Use

physical arguments to determine the initial value and final value for the step response of this circuit.

(2) Use Matlab to determine the transfer function of the biquad circuit. Check that your answers are consistent with your results to (1). Determine the effect of doubling and halving all the impedances in this biquad.

(3) Determine how to pick the resistance values so that the transfer function of the biquad is proportional to the transfer function of a series RLC circuit with an output across one of the circuit elements. Use Matlab to determine a condition for an over-damped circuit, and for an under-damped circuit. Simplify your results when $R3 = \infty$, and $R4 = 0$. Use the Matlab command pzmap(n,d) to verify your result. Determine a condition to insure the circuit has no poles in the right half plane.

(4) Plot the step response of the biquad for an under-damped, critically-damped, and over-damped case. Check that your answers are consistent with your results to (1).

(5) Plot the magnitude and phase of the transfer function of the biquad for an under-damped, critically-damped, and over-damped case.

(6) Find the changes to your solutions to (4) and (5) when all the capacitor values are doubled; and again when they are halved. Discuss your result.

(7) Use Matlab to find the response of this circuit to the input u(-t); this is referred to as the natural response of the circuit. Use a physical argument to find the initial and final value of the natural response; use this to check your Matlab solution.

EXPERIMENTAL

Construct the biquad for an under-damped, critically-damped, and over-damped case; use reasonable values for the circuit elements. Measure the square wave response of the biquad in each case; compare your experimental results with the step response and natural response found with Matlab. Measure the frequency response of the biquad in each case. Carefully compare your results with the results obtained from Matlab; discuss your results.