ECE4390 Lab 2

To Begin, download all lab 2 files into the same folder from the class website. Using the MATLAB files provided as a starting point, write a program that will solve for circuit node voltages with Modified Nodal Analysis (MNA). The program must be able to find solutions in both frequency and time domains. In the time domain, solutions must be computable using the Backward Euler (BE) and Trapezoidal Rule (TR) schemes. When solving the matrix equation, the "\" operator or `linsolve()` may be used but bonus marks will be awarded if LU-decomposition is implemented and is faster than the backslash operator alone (note that a very efficient algorithm is used internally by MATLAB when the matrix to the left of the "\" operator is upper or lower triangular).

Any circuit to be solved by your implementation must be described as a netlist similar to what was shown in class. This netlist will be read by the `ReadCirc()` function provided. To be read successfully, the netlist must be given in a text file with the following formatting:

Type Name Node1 Node2 Node3 Node4 Parameter

In the above, the fields are space-delimited. *Type* is a 1 to 3 character string describing the branch element. A naming scheme is already given in *main.m* but you can create your own if you wish. *Name* is a 1 to 20 character label that is given to the branch element. The *Node* fields describe the location of the branch within the circuit network. For one-port elements requiring only two nodes, a "0" must still be placed in the two remaining *Node* fields. *Parameter* is an entry of float type and is the value assigned to the branch element. For example, the *Parameter* field for a 100Ω resistor would contain "100". For elements that do not require a value, enter "0". Any line starting with a "%" will be commented and ignored by the netlist reader.

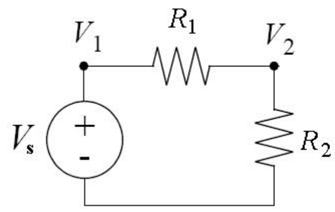
A report is to be handed-in by the start of the next lab. All the requested plots are to be submitted. Be sure to include any explanations describing the plots as needed. Also, submit a print-out of your code but if you wish to save paper, the files can be emailed to umfauch2@cc.umanitoba.ca. Be sure to include your name and student number. There will also be a 15 to 20 minute quiz to be written near the end of the lab period.

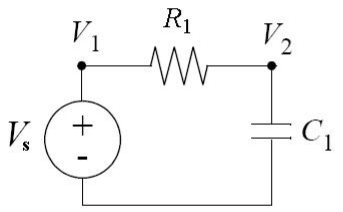
1a. The MATLAB code provided can solve a circuit containing voltage sources and conductances only. The input file "`circ.txt`" contains an example netlist for a voltage divider with conductances. Run the code and see what happens. Become familiar with the program structure.

b. Make the necessary code modifications to simulate the same circuit with resistances instead of conductances. To test your code, find the voltage at V_2 in the resistor circuit below for different values of R_1, R_2 and V_s.

c. To test your finite difference schemes, plot the time-domain response of the

RC circuit to a step input. Try different parameters for R_1 and C_1. Observe the variation in the solution for different time-step sizes and its relation to the circuit time constant. Note: the `heaviside()` function emulates the step function but use it wisely since it is not defined at $t = 0$ in our version of MATLAB.

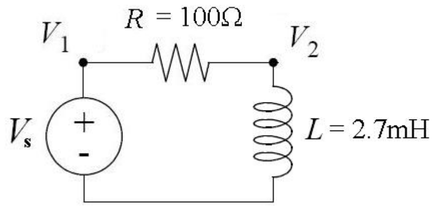




2a. For the RL high-pass filter given below, plot its frequency and phase response to a $1V$ input for the frequencies ranging from $0Hz$ to $30kHz$. What is its $3dB$ cut-off frequency? Note: the `abs()` and `angle()` MATLAB commands return the magnitude and phase of a complex number, respectively.

b. Plot the voltage across the inductor as a function of time. Use a step function at the input and compare it to the analytic solution ($V_L = e^{-t\frac{R}{L}}$).

c. What is the time-step size limit for stability of the solution for the filter below? Illustrate your findings with a plot of the solution for different values of Δt for the BE and TR finite difference schemes shown in class. Note the difference between instability and inaccuracy of the computation.



3a. Create a netlist for the circuit shown on page 175 of the course text book (also available for download from the course website in CH4 Vlach and Singhal) and pass it to your MNA program for simulation. Plot the frequency response of the filter for frequencies ranging from $0Hz$ to $50kHz$. Note: the waveform should look like figure 4.10.4 on page 176. Also, to convert voltages to decibels, use the expression $20 \log \frac{V_{out}}{V_{in}}$ and in MATLAB use `log10()` to find the base 10 logarithm of a number.

b. The file "noisy_sample.wav" is sound clip with added high frequency noise. Remove this noise using your MNA program and the filter from part (a). This is done by setting the audio clip as the "source" in the program and recording the output voltage at node 5. The audio sample can be imported into MATLAB with the following command: `[source, fs] = wavread('noisy_sample.wav')`.

The vector *source* will hold a “voltage” representation of the audio clip and *fs* is its sampling frequency. The sampling frequency is used to determine the time-step size for the system. A “sound vector” can be played on the computer speakers with the `sound(source,fs)` command but MAKE SURE THE SPEAKER VOLUME IS TURNED DOWN before you do this. To demonstrate that the clip was properly filtered, plot the output waveform.