Lab 1: Modeling and Simulation in MATLAB / Simulink

"Any fool can use a computer. Many do." – Ted Nelson

1 Objectives

The goals of this lab are:

- To become familiar with the MATLAB and Simulink environments.
- To learn to construct state space, transfer function and block diagram models of dynamical systems and to simulate these models in MATLAB and Simulink.

2 Theory

MATLAB (MATrix LABoratory) is a software package that is widely used in control systems design. Simulink is a graphical front end to MATLAB that allows you to easily create models of dynamical systems in form of block diagrams. In this lab, you will learn how to construct different representations of the model of a simple RLC circuit and simulate its behavior.

This lab focuses on the use of MATLAB as a tool in control system design. If you have never been exposed to MATLAB before, please consult one of the many introductory resources available online[1]. You will mainly be using the MATLAB Control System Toolbox. A great way to get started with the toolbox is to run the demo. This is done by typing `demo('toolbox','control')` at the MATLAB prompt. One great thing about MATLAB is its comprehensive and easy-to-use documentation. Make use of it!

3 Pre-lab

3.1 Reading

Please read the following documents before the lab:

- "Lab Policies" handout
- "Useful MATLAB Commands" handout
- "Common Simulink Components" handout

3.2 Modeling a simple RLC circuit

Consider the simple RLC circuit shown in Figure 1.

1. What are the differential equations describing the dynamics of the current, $i(t)$, and the capacitor voltage, $v_C(t)$?
2. Find the transfer function $F(s) = \frac{V_C(s)}{V(s)}$ relating the capacitor voltage, $V_C(s)$, to the input voltage, $V(s)$.

[1]For an (incomplete) list, see `http://www.mathworks.com/academia/student_center/tutorials/launchpad.html`.

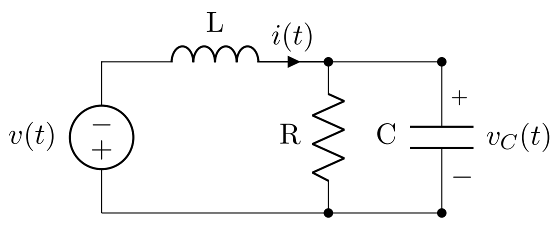

Figure 1: Simple RLC circuit

3. Choosing $x_1 = i$ and $x_2 = v_C$ as states, $u = v$ as input and $y = v_C$ as output, derive the state space model of the RLC circuit (that is, find the matrices A, B, C, D of the model $\dot{x} = Ax + Bu$, $y = Cx + Du$).

4 Lab

For the lab, use values $R = 2\,\Omega$, $L = 10^{-4}\,\text{H}$ and $C = 10^{-4}\,\text{F}$.

4.1 Simulating the RLC circuit in MATLAB "by foot"

In this section, you will use MATLAB to solve the ODEs you derived in the pre-lab. Matlab supports many different numerical schemes for solving ODEs. Here you will use the solver `ode45`, which is based on a variable step Runge-Kutta method. This is usually a good choice for most non-stiff[2] ODEs.

1. Choosing as states $x_1 = i$ and $x_2 = v_C$, write a MATLAB function `xdot = RLCdynamics(t,x)` that takes as inputs the time `t` and the vector of states `x`, and returns the vector of time-derivatives of the state, `xdot`. Assume a constant input voltage of $v(t) = 1V$. You may hard-code the values for R, L and C given above or, alternatively, use global variables. *Note:* Your derivatives will not explicitly depend on the time t. The reason the function `RLCdynamics` takes t as an argument is that MATLAB's ODE solvers expect this particular form.
2. Solve the system of ODEs numerically for the initial condition $i(0) = v_C(0) = 0$ on the time interval $t \in [0, 1 \cdot 10^{-2}]$ by calling `[t,x] = ode45(@RLCdynamics,tspan,x0)`. This uses the `ode45` solver with standard settings. Consult the MATLAB documentation for `ode45` about how to choose the values of `tspan` and initial state vector `x0`.
3. Plot the evolution of the current i and the capacitor voltage v_C as a function of time in a single plot.

4.2 Creating and using a state-space model

For the same choice of input, output and states as before, determine the state-space matrices A, B, C and D of the RLC system, using your results from the Pre-lab. Construct a state space model using the command `RLCss = ss(A,B,C,D)`. Plot the step response of the system by calling `step(RLCss)`.

[2]A *stiff* equation is a differential equation for which certain numerical methods for solving the equation are numerically unstable. A linear constant coefficient system is called stiff if all of its eigenvalues have negative real part and the ratio between largest and smalls eigenvalues is very high.

4.3 Creating and using a transfer function model

For a SISO transfer function of the form

$$G(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + ... + b_0}{a_n s^n + a_{n-1} s^{n-1} + ... + a_0}$$

you can construct a transfer function model using the command `sys = tf(num,den)`, where `num =` $[b_m,\ b_{m-1},\ \dots,\ b_0]$ and `den =` $[a_n,\ a_{n-1},\ \dots,\ a_0]$. Construct a transfer function model `RLCtf` of the transfer function $F(s)$ you derived in the pre-lab. Plot its step response using `step(RLCtf)` and make sure it matches the step response of the state space representation. Another way to check your results is using the command `ss2tf` to convert your state space model to a transfer function model.

4.4 Creating and using a Simulink model

You can start Simulink by typing `simulink` in the MATLAB command prompt. The Simulink Library Browser should open, from which you can select the blocks in your model. To create a new model, choose "File → New → Model".

Drag-and-dropping the necessary blocks from the Library Browser, create the model of the RLC circuit shown in Figure 2. Here $K1$, $K2$ and $K3$ are constants depending on R, L and C that you need to determine. The simulation of the model will run properly as long as $K1$-$K3$ are defined as variables in the MATLAB workspace. Alternatively, you can directly enter expressions depending on R, L and C as the values of the gain blocks. If needed, you can perform additional changes to the model configuration by opening the dialogue "Simulation → Model Configuration Parameters". Save the model under the file `RLC.mdl`.

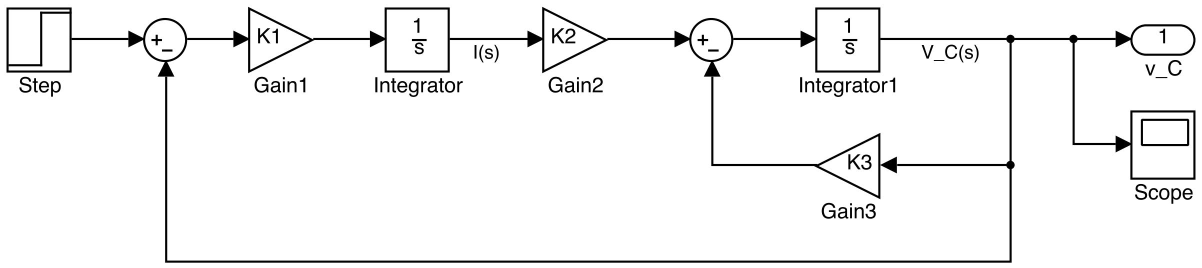

Figure 2: Simulink model of the RLC circuit

Note that the model in Figure 2 does not use differentiation blocks. Why do you think that is?

There are two main ways you can simulate your model. One is to use the Simulink GUI, the other one is to use the MATLAB command `sim`. The latter way is very useful if you want to automate different simulations using a MATLAB script.

To simulate the model from the GUI, simply enter a simulation duration in the field at the top of the model window and hit "Run". To simulate your model from the command prompt or a script, use the command `sim('RLC')`. Consult the MATLAB documentation of `sim` on how to pass additional simulation and configuration parameters to the model. You can use the Scope block to show the value of any signal in your model. Additionally, each output block will be accessible as a variable in the workspace after simulation (with default name `yout`). You can change the name of the variables that are saved to the workspace under the "Data Import/Export" section in the model configuration parameters window.

1. Change the "Max step size" in the "Solver" settings of the configuration parameters window to $1 \cdot 10^{-6}$. Set the "Step time" value of the Step block to zero and simulate the model over the time interval $t \in [0, 1 \cdot 10^{-2}]$. Plot the resulting capacitor voltage $v_C(t)$ over time.
2. The default initial condition for integrator blocks in Simulink is zero. Change the initial condition of the block corresponding to the capacitor voltage to $1V$. Simulate the model and plot the output.
3. Replace the step input in your diagram by a square wave with amplitude $1V$ and frequencies $f = 25, 250\,\text{Hz}$. Provide separate graphs of the response at each frequency, and explain their differences.

   Hint: You can use the "Signal Generator" block under the "Sources" section of the Library Browser (set the "Time(t)" field in the block parameter settings to "Use simulation time"). Again simulate the model and plot the resulting signals $v_C(t)$.

4.5 Comparison of the different methods

1. Which of the different model representations discussed above can be used to model nonlinear dynamical systems?
2. Which of them is well-suited to account for different initial states?
3. Can you think of additional advantages and disadvantages of the different model representations?

In the upcoming labs, we will model our systems mostly in Simulink. One of the main reasons for this is that the software that we use for our hardware-in-the-loop experiments integrates very well with Simulink. However, you should take away from this lab that there are other ways to represent, simulate and analyze dynamical systems in MATLAB.

4.6 Individual lab report

For this lab only, you should submit an individual report.