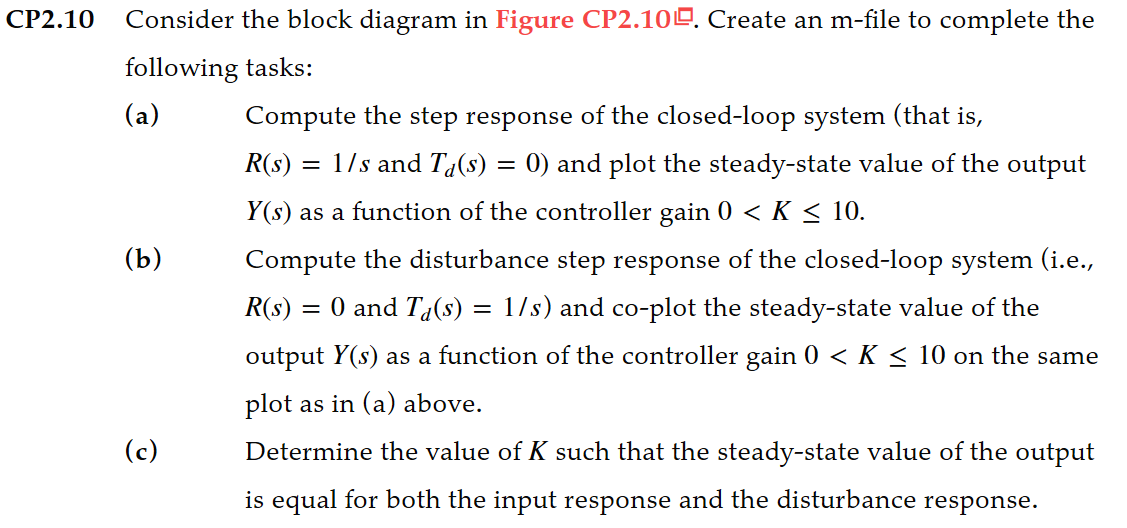
# TCES 455, Autumn 2016

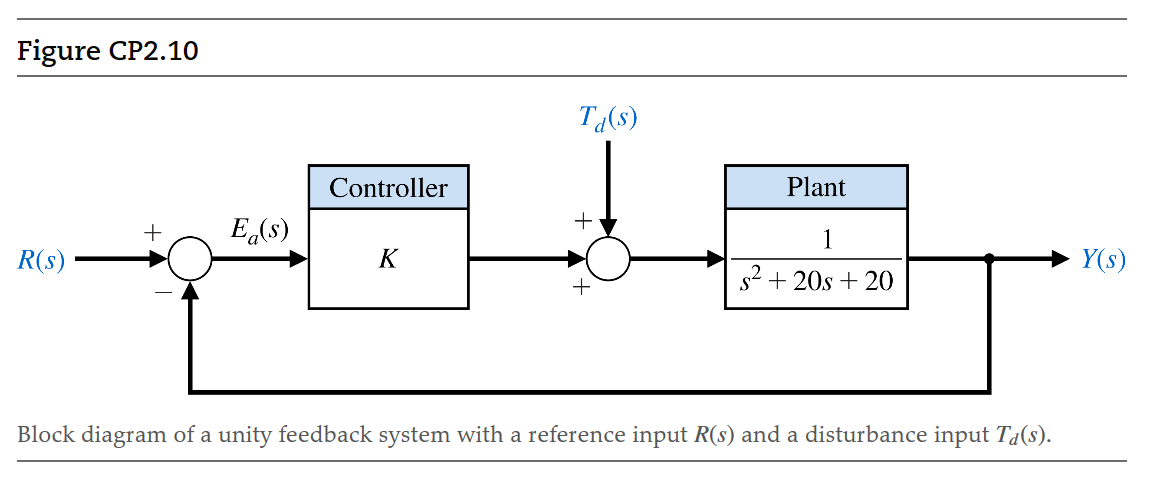
**Laboratory 2: MATLAB/Simulink Lab**

Prior to lab:

* Install MATLAB/Simulink.
* Read the Textbook Appendix A: MATLAB Basics, and Chapter 2.
* Read the basics about Simulink at <http://ctms.engin.umich.edu/CTMS/index.php?aux=Basics_Simulink>

1. **Create an m-file as required by problem CP2.10 in the Textbook, to complete the given tasks.**





Once these tasks are done, your notebook should contain

* m-file for each task, and the required plot.

1. **Continue the following Simulink excises on modeling of an armature controlled DC motor.**

Consider a permanent-magnet DC motor characterized by the following set of coupled differential equations:

Electrical Dynamics: ,

Developed Electromagnetic Torque: ,

Mechanical Dynamics: ,

where  is the applied armature voltage,  is the armature current, is the angular velocity of the rotor shaft in radians/second,  is the developed electromagnetic torque in Newton-meters, and the output load torque is given by the  term.

The constants are:

: armature resistance in ohms

: armature inductance in Henries

: the back emf of torque constant in volt-second or Newton-meter/amp

: sum of the armature inertia in kilogram-meter squared and the reflected load inertia (via the gear ratio)

: viscous friction constant in Newton-meter-second

: the load torque constant in Newton-meter-second squared

Ultimately the armature voltage will be the controllable output of a power converter and feedback system and we will be able to dynamically adjust the armature current and control the speed of the motor. For today,  will simply be a constant that we can specify.

Starting SIMULINK

To begin our investigation, open MATLAB and at the prompt type

>> simulink

Note the extensive list of libraries and functions along the left portion of the Simulink window. In order to create a simulation model, you need to select

File 🡪 New 🡪 model

Where Components and Functions are Found

This new window that opens is where we will be dropping components from the libraries and interconnecting them. If you left-click the *Continuous* Library, it will display such items as



which will be the key building blocks for modeling dynamic systems. If you double-click the Transfer Function block, you can change the polynomial coefficients (highest order to lowest order) or make them variables to be assigned in a MATLAB M-file.

The *Math Operations* Library also has a couple of key components, where the Product block is typically reserved for creating terms like  whereas the Gain block is used to make .



Finally, in the *Sources* Library, we have the ability to specify constants, create ramps, input values of the Matlab workspace, and keep track of the time variable (Clock).



A component is placed into the model window by left-click-hold then moving the part into the desired position. Once in the model window, if you need to flip or rotate the element select

Right click 🡪 Format 🡪 flip block

If you double left click the block, you open a parameter window. For example for the summing block, you can change the icon shape, the number of inputs, and their respective polarities.

Connecting Components

Components may be connected by either

1. Left select the “from” block – hold the CTRL key – left click the “to” block
2. Start from the “out” port (left click and hold) and proceed to the desired “in” port

You may always left-select a connection path and eliminate it by pressing delete. As diagrams become more complex, “From” and “GoTo” blocks in the *Signal Routing* Library may be used to manage the signal flow connections. Further for complex block diagrams, you can also subdivide your system into “subsystems” where only the inputs and outputs become visible at the top-most level. The feature is found in the *Parts & Subsystems* Library but will not be delved into today.

To Save Outputs to Matlab

First you will need to capture the time variable using the “Clock” function in the *Sources* Library. Next you will need to concatenate time along with other variables of interest using the mux block found in the *Signal Routing* Library



By double-left clicking, you can open a parameter block and specify the number of inputs. The mux output may be directed to a scope (*Sinks* Library) or to the Matlab workspace using the “To Workspace” block in the *Sinks* Library.



Make sure to double-click the “To Workspace” block and change the “Save Format” setting from *Structure* to *Array.* You may also change the outputted variable name from simout to something more descriptive. The variable simout will house column-wise the variables from the mux. So for instance, if time is the first element of the mux, we could then access it in Matlab and reassigning the name by typing

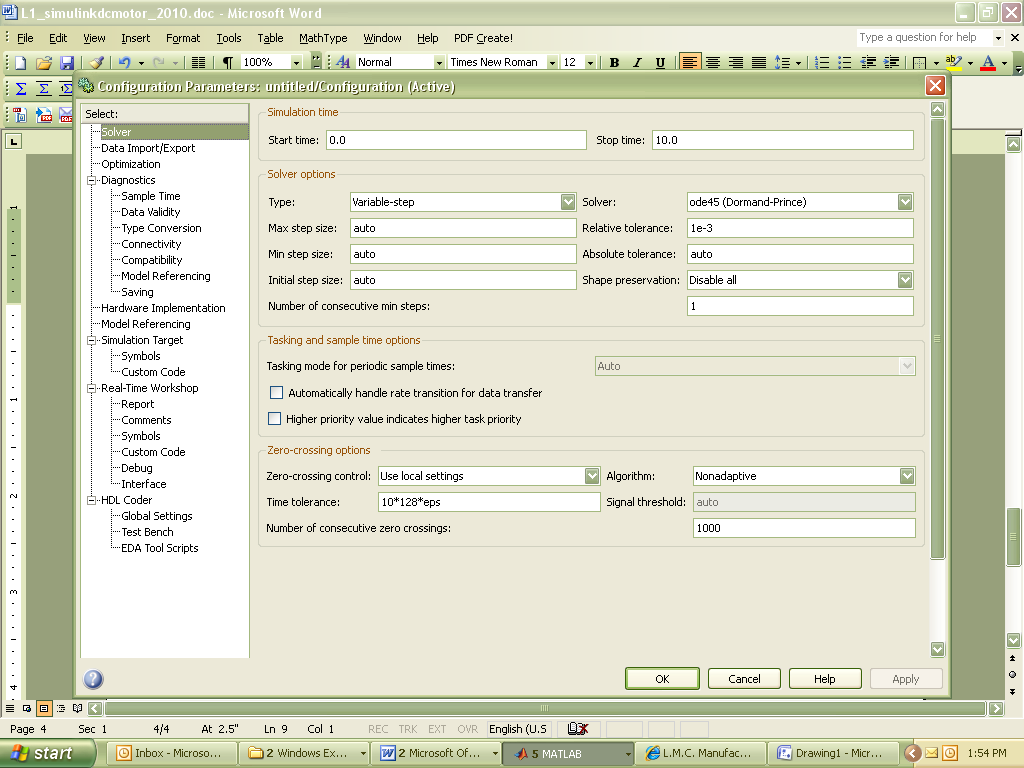
>> t = simout( : , 1 );

If we repeat this for the other variables, then we are set up to conveniently generate plots or perform data analysis.

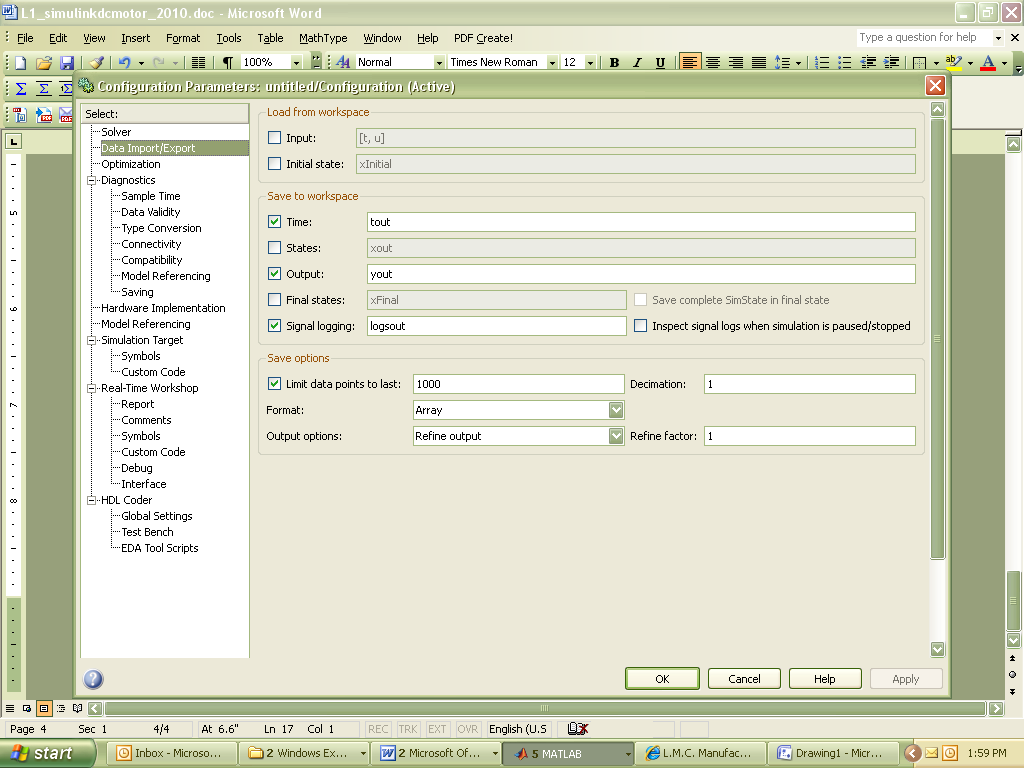
To Set Up a Simulation Run

After saving a completed model (making sure that no inputs are left unassigned), you next need to let SIMULINK know how long you want the simulation to execute, which numerical integration routine to use, and what time step to employ. In the model window select,

Simulation 🡪 Configuration Parameters



As shown, under the “Solver” tag (on the left menu), we can adjust the start and stop times and the solver routine. The variable-step solvers continuously adjust the algorithm step size to maximize the computational efficiency while maintaining a specified accuracy. The ode45 routine is excellent and is well suited for systems with discontinuities as we might find in power converters. On the “Data Import/Export” tag,



we see that we can automatically save the final values of the state variables to the Matlab workspace. We can then use these saved variables to initialize a future study (using the Initial state check box in the window). Essentially, state variables are established for each integrator block used and for and for each transfer function denominator power of ‘s’. We will explain more about state variables as we encounter them in practice.

To Start and Run the Simulation

Once you have built your model, established the desired outputs to Matlab, and configured your solver parameters, the simulation is executed by selecting

Simulation 🡪 Start

Or clicking the  menu button. A “beep” will signify when the run is completed. OK, let’s apply what we now know to modeling and simulating the PM DC motor.

Step 1: Substitute  into the mechanical dynamics equation and solve for  and  (do this with the parameter names maintained)





The variables  and are our *state variables*. These are the quantities that will change with time.

Step 2: Next, build these equations using integrators, gain blocks, summing junctions, and a multiplier. Fill in the gain blocks with the appropriate terms and label the summing blocks with ‘+’ or ‘-‘ as required.



The variables iaic and  are the initial conditions on the integrators. These may be assigned to these variable names by double-left-clicking the integrator block and typing in the name.

Step 3: Open Simulink (if it is not open) and create the block diagram developed in Step 2. Use a “constant” block for the armature voltage. Include a time clock in the diagram. Use a mux to concatenate time, motor speed, and motor current. Then use a “To Workspace” block to send the matrix to Matlab (make sure to make the switch from structure to array as discussed earlier). Save the Simulink model.

Step 4: Create a Matlab M-file that will assign the following parameters (make sure the names are the same as you used in the Simulink diagram!). Recall, the total inertia is the sum of the motor and reflected load inertias. Run the file so these parameters are then available to your Simulink model.

Step 5: Set-up a SIMULINK run that captures the startup transient (do this by trial and error as to when the speed settles out…you may choose to place a scope on the motor speed to do this more efficiently). In MATLAB, plot  and  versus time.

Once these tasks are done, your notebook should contain

* Information you discovered in Steps 1-4, and how you discovered it
* Equations, Simulink model, M-files, and required plots. They should either be written down or printed out and included in your notebook.