ME:6130-Novel Artificial Muscles and Sensors
MATLAB/SIMULINK Tutorial

Primary Course Instructor: **Dr. Caterina Lamuta**
Tutorial Preparation & Teaching Assistant: **Thilina H Weerakkody**

March 04, 2024 (Updated)

- Open *"Apps Anywhere"* and open the latest MATLAB version *(MATLAB 2023a)*.
- Download the folder from ICON.

1 Setting up the Simulink file

1. Open the downloaded folder "ME6130-TCAM Dynamic Simulation Model" using MATLAB current directory tab.
2. Change your *current folder* to the downloaded folder.
3. Open the Simulink file "SIM_model.slx".
4. Open "MODELING" ribbon and "Model Settings". (Please follow steps 1-3 in Figure 1)

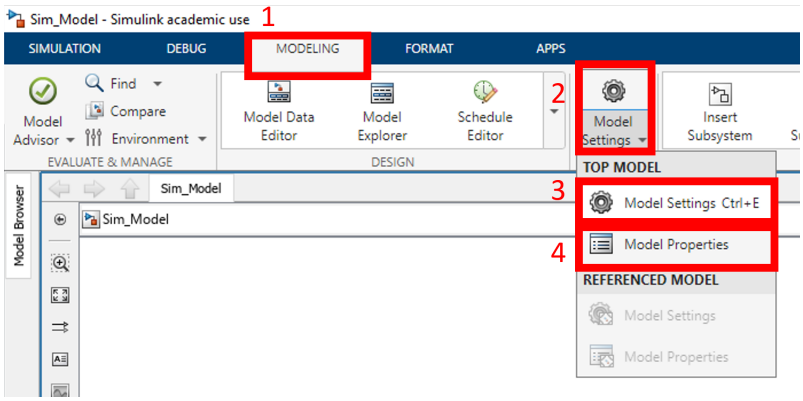

Figure 1: Open Model settings

5. Change the time steps to *Fixed-step* by drop down *Solver* → *Solver selection* → *Type* → *Fixed-step*.
6. Next, let's set the time step size. You may use the experimental data-set time step. Go to *Solver details* and set the *Fixed-step size* to T_sample (Step 1-3 in Figure 2). Apply all your changes.
7. Set the simulation stop time to 40.0 (Step 4 of Figure 2).
8. Create a *.m* MATLAB script. Name it as *SIM_parameters.m*. Input all the variables into this file. We can use SIMULINK callbacks (in the Model properties) to load the variables in the *SIM_parameters.m* script.

   During each SIMULINK running cycle, it will go through callbacks and run the scripts listed in callbacks. If we update variables in the *SIM_parameters.m* script and list in callbacks, callbacks ensure all the parameter

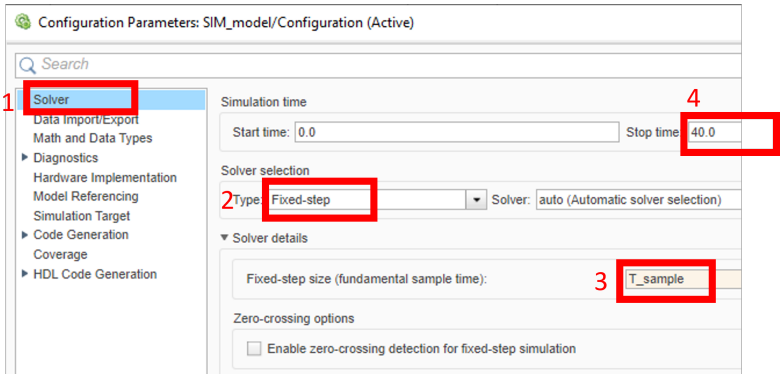

Figure 2: Open configuration parameters

updates and affect the SIMULINK model in every simulation. Also, it is important to *clear all* the variables in the MATLAB workspace at the beginning of every simulation. This can be done by adding *clear all* command to line 1 of the parameters file.

Please follow steps 1,2, and 4 in Figure 1 to open Model Properties. Then, follow steps 1 and 2 in Figure 3 and add the SIM_parameters.m file in Callbacks→InitFcn as *SIM_parameters*.

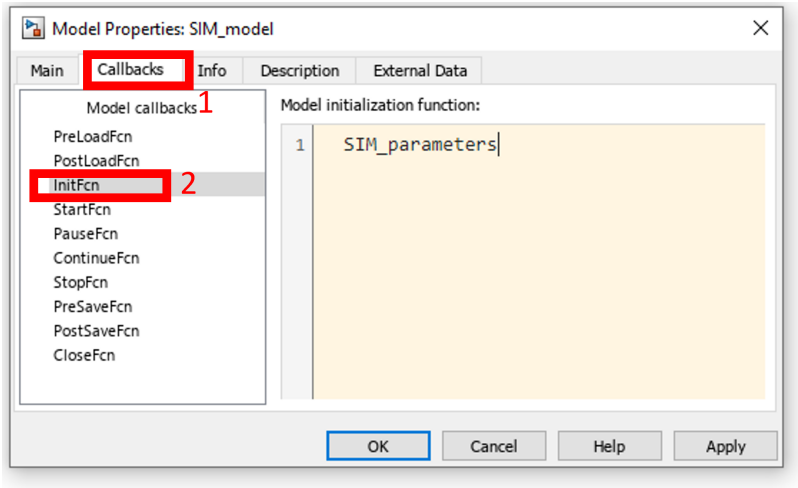

Figure 3: SIMULINK callbacks

2 Import the experimental data to SIMULINK

Let's update both *SIM_parameters.m* to load the data. (Import Carbon Fiber experimental data set.)

```
clear all                       % clear all in the workspace
load("ME6130_Dataset_CF.mat")% load the dataset
T_sample = 0.05;                % Sample time [s]
```

There are two methods to find the SIMULINK blocks.

- Using *Library Browser* in the SIMULATION→Library or,
- Left double click and search for the block.

All the block names required for this model are text-colored in blue. Please search them by name. The commands are colored in red.

1. Get From Workspace block and a Scope block.
2. Open the From workspace block and update the data parameter as follows. [Time, Disp]. (Refer Figure 4).
3. Run SIMULINK and view the displacement data by double-clicking the scope.

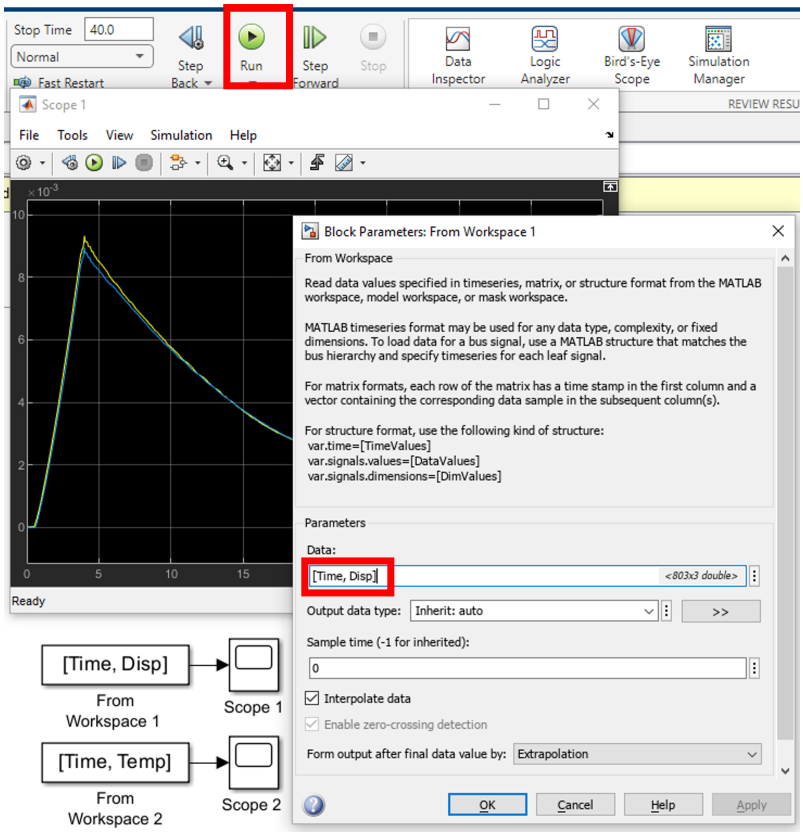

Figure 4: SIMULINK visualize experimental data

4. Load the temperature data using the same method. Read the data using the From Workspace block using the command [Time, Temp].

3 Implementation of the theoretical equations

Let's implement the theoretical model. In our experiment lab session, we Joule heat the TCAM using a constant voltage V_1 over a time t_1 followed by cooling for $(40 - t_1)$.

1. Get the Step source block and update all the parameters as in Figure 5.

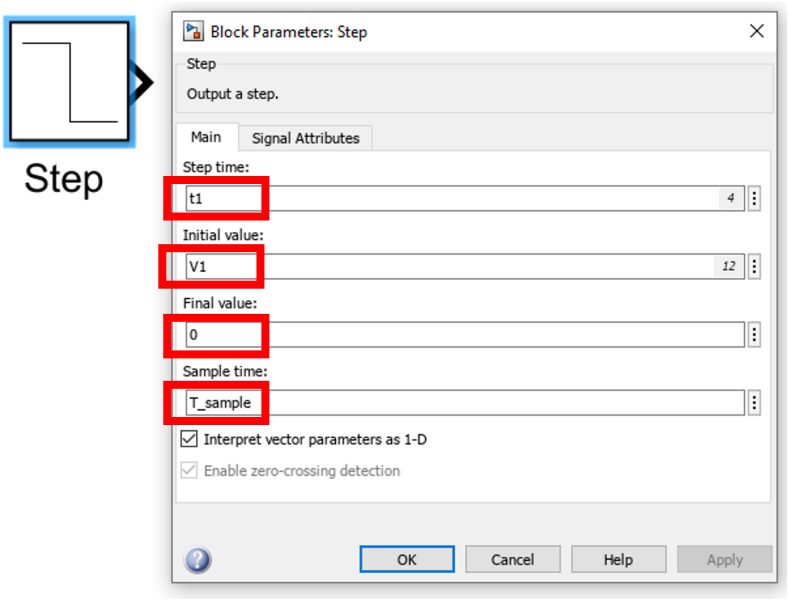

Figure 5: Step function

Update your *SIM_parameters.m* as below. (Line 4 and 5)

```
1 clear all                    % clear all in the workspace
2 load("ME6130_Dataset_CF.mat")% load the dataset
3 T_sample = 0.05;             % Sample time [s]
4 V1 = 15;                     % exp voltage [V]
5 t1 = 4;                      % V1 on time [s]
```

Alternatively, you may use if/else blocks with a clock, MATLAB function block with a clock, or From workspace block from voltage data to generate this step function. (Refer Figure 6)

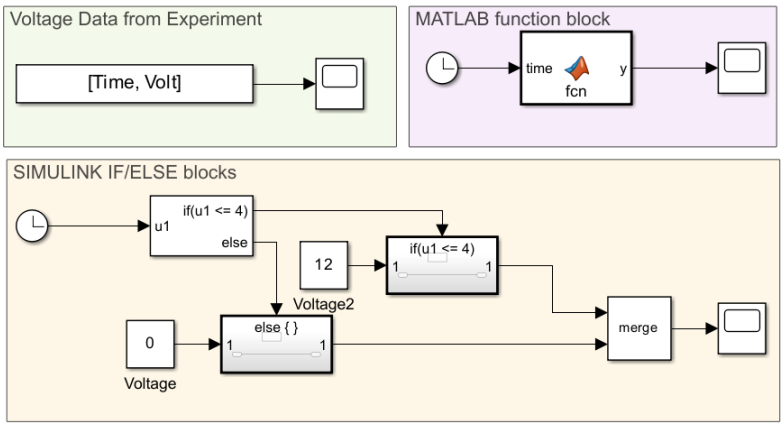

Figure 6: Other Input Methods

2. Create a Subsystem and name it as the Thermal Model. Connect to the step block. Inside the thermal model, let's implement the following equations.

$$\dot{T}(t) = \frac{V^2(t)}{C_t R} - \frac{\lambda(t)}{C_t}(T(t) - T_{amb}), \tag{1}$$

$$\lambda(t) = h(t)A_s(t). \tag{2}$$

$$h(t) = \frac{K}{2\rho(t)}\left(0.6 + \frac{0.387Ra_L^{1/6}}{\left(1 + \left(^{0.559}/_{Pr}\right)^{9/16}\right)^{8/27}}\right)^2, \tag{3}$$

Temp For this tutorial, you will only implement equations (1) and (2). Equation (3) is provided in the supplementary .m file.

Update your parameters in Lines 6 - 10 in the *SIM_parameters.m* as follows.

```
1  clear all                    % clear all in the workspace
2  load("ME6130_Dataset_CF.mat")% load the dataset
3  T_sample = 0.05;             % Sample time [s]
4  V1 = 15;                     % exp voltage [V]
5  t1 = 4;                      % V1 on time [s]
6  R = 17.0;                    % Resistance [ohm]
7  Ra = 40e6;                   % Rayleigh number for the characteristic length
8  Tamb= 25;                    % Ambient temperature [C]
9  Ct = 0.47;                   % Thermal mass [J/C]
10 xi = 0.105;                  % Length of the coiled TCAM [m]
```

Open the SIMULINK Thermal Model subsystem and implement the equations using blocks (or MATLAB function block) as shown in Figure 7. Set the initial condition of the Integrator block as T_{amb} by opening the block parameters and updating the variables. Similarly, update all the block properties with variables (upon requirement).

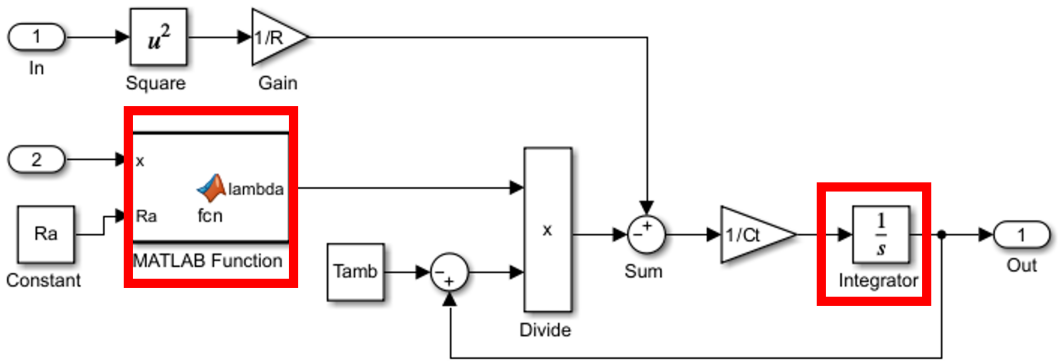

Figure 7: Thermal Model subsystem (Update the Block properties of the Red-colored blocks)

Please copy and paste equation (3) from the supplementary_2 to the MATLAB Function block.

```
function lambda= fcn(x, Ra)
Pr = 0.71;           % Prandtl number
k_heat = 28e-3;      % Thermal conductivity of the TCAM
rho = 0.0082;        % Initial radius of the coil
Nu = ((((Ra^(1/6))*0.387)/(((0.559/Pr)^(9/16)+1)^(8/27)))+0.6)^2;
h  = k_heat*Nu/(x);% Convection heat transfer coefficient
As = 2*pi*rho*x;    % Effective cross-section area of the TCAM
lambda = h*As;
```

The model should be as in Figure 8. Add Mux block to view the theoretical and experimental data using the same scope. Before *RUN* the model, add a Constant block with *xi* parameter. This is a temporary parameter to complete the model and run to identify any errors. Later we will delete this Constant block and connect the feedback from the Mechanical Model subsystem.

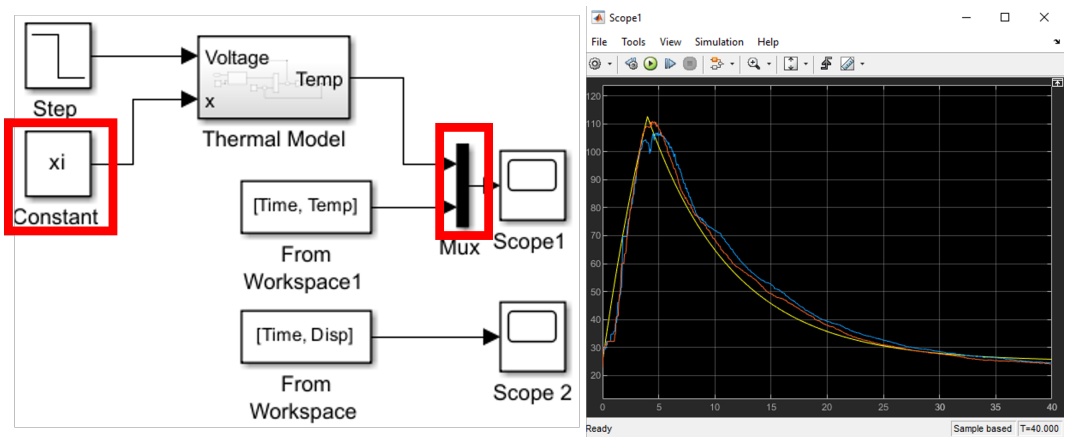

Figure 8: Updated model with Thermal Model subsystem and the temperature scope view. (Please add the Red-colored blocks)

When you *RUN* the model, you may see both simulation and experimental plots on the same scope.

3. Create another Subsystem block and name it as Mechanical Model. Connect the Thermal Model output as the input to the Mechanical model input (which is the temperature or *Temp*). Using another Mux block and connect the Mechanical model output to the scope. These updates are shown in Figure using red boxes.

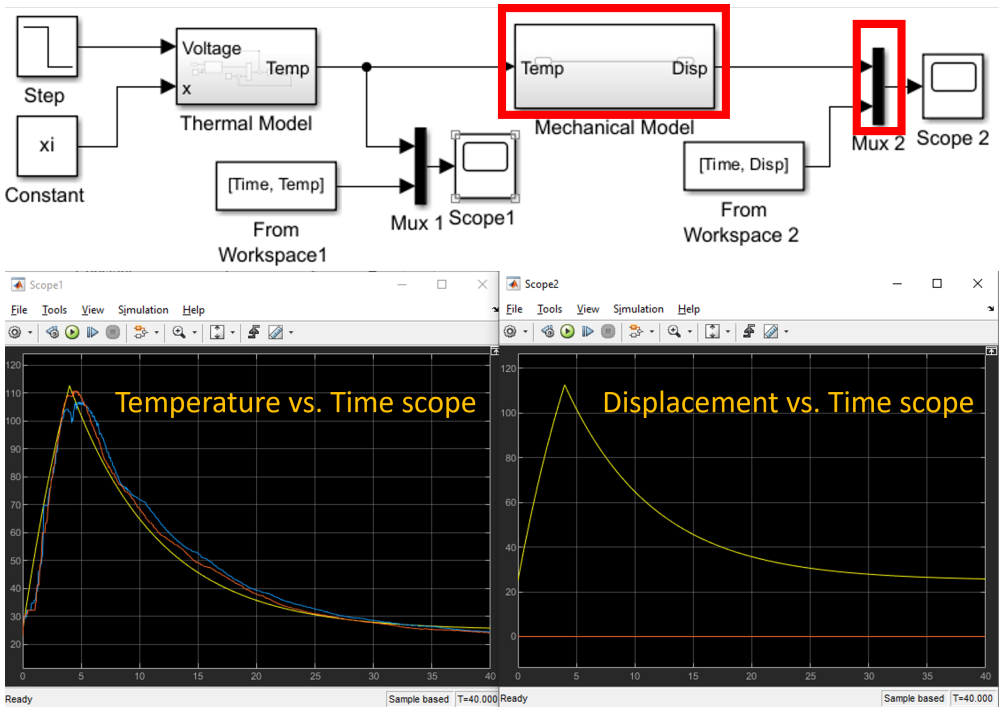

Figure 9: Updated model with Mechanical Model subsystem

At this stage, the two-scope data visualization is as in Figure 9. While there is a good fitting for temperature data, displacement scope doesn't show even the experimental results. Once we complete the Mechanical model equations, this will be solved.

4. Let's Implement the mechanical model. We can start with a static model. Open the Mechanical Model subsystem and implement the below equations using SIMULINK blocks or a MATLAB Function block.

$$\frac{r(t) - r_0}{r_0} = CTE\ (T(t) - T_{amb}). \tag{4}$$

The supplementary code is attached to the detailed calculation for obtaining Young's Modulus E_x and Shear Modulus G_{yx}. Please use the code to obtain the both E_x and G_{yz} to calculate dynamic $\cos\beta(t)$

$$B(t) = E_x \frac{\pi}{4} r^4(t), \tag{5}$$

$$C(t) = G_{yz} \frac{\pi}{2} r^4(t), \tag{6}$$

$$q = \frac{2\pi n}{S}, \tag{7}$$

$$\cos\beta(t) = 1 - \frac{1}{B(t) - C(t)} \left(\frac{qB(t)C(t)}{\sqrt{B(t)mg}} - 2C(t) \right), \tag{8}$$

$$\bar{\ell} = S\cos\beta(t). \tag{9}$$

Let's update the *SIM_parameters.m* file from lines 12-26. These parameters are essential for modeling the next set of equations.

```
Material = 1;                % 1 for Carbon fiber and 2 for Ny
CTE = 6.1*10^(-5); %         % Coefficient of Thermal Expansion [1/C]
r0 = 0.29e-3;                % Initial fiber radius [m]
S  = 0.25;                   % Initial fiber length [m]
n = 120;                     % number of coiled turns
m = 50e-3;                   % Fabrication mass [kg]
g = 9.81;                    % gravity [m/s^2]
Ex = 6.75e9;                 % Longitudinal Young s modulus [N/m^2]
Gyz= 8.8e7;                  % Shear modulus [N/m^2]

q = 2*pi*n/S;
B_ = Ex*pi*r0^4/4;           % Bending stiffness [Nm^2]
C_ = Gyz*pi*r0^4/2;          % Torsional stiffness [N/m^2]
cosbeta_ = 1-(1/(B_-C_))*(q*B_*C_/sqrt(B_*m*g)-2*C_);
rho = sin(acos(cosbeta_))/(m*g/B_)^0.5; % Initial coil radius of the coil
```

Then implement equation (4) using the SIMULINK blocks. Then create a MATLAB Function block and copy-paste the provided code to calculate E_x and G_{yz} from equation (5) – (9). Figure 10 shows the completed view.

The following code is available in your supplementary file 3, which should be pasted in the MATLAB Function block created inside the Mechanical Model.

5. Afterwards, let's implement the dynamic model. The highly nonlinear TCAM dynamics are modeled using the second-order dynamic equation.

$$m\ddot{\ell}(t) + b(t)\dot{\ell}(t) + k(t)\ell(t) = F_e. \tag{10}$$

The experimental setup configuration can be schematically shown in Figure 11.

Let's find all the constants for implement (10).

$$k(t) = \frac{mg}{S\cos\beta(t)}. \tag{11}$$

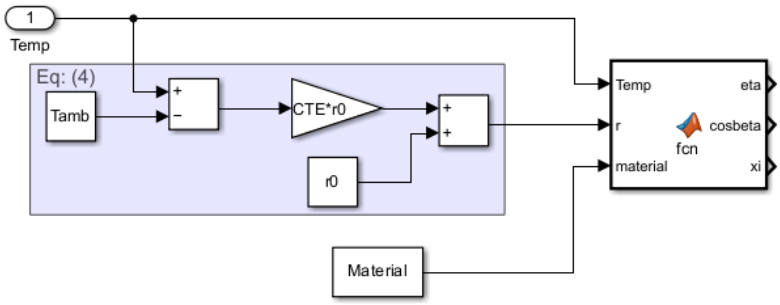

Figure 10: Thermal Expansion calculation

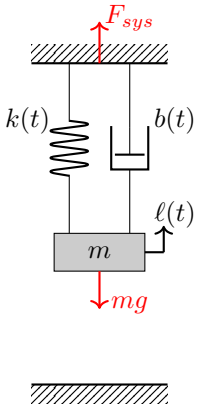

Figure 11: Schematic representation of the experimental setup for constant stress TCAM configuration

$$\eta(t)\frac{d\epsilon(t)}{dt} + E\epsilon(t) = \sigma(t), \tag{12}$$

$$b(t)\dot{\ell}(t) + k(T_\Delta)\ell(t) = \bar{m}g. \tag{13}$$

$$\frac{d\epsilon(t)}{dt} = \frac{\dot{\ell}(t)}{\bar{\ell}}. \tag{14}$$

$$b(t) = \frac{\eta(t)A_s}{\bar{\ell}}. \tag{15}$$

By substituting all the parameters obtained in equations (11) and (15) we can obtain the following equation.

$$\underbrace{\underbrace{m\ddot{\ell}(t) + \bar{b}\dot{\ell}(t) + \bar{k}\ell(t)}_{F_{init}} + \underbrace{b_\Delta\dot{\ell}(t) + k_\Delta\ell(t)}_{F_{TCAM}}}_{F_{sys}} = mg, \tag{16}$$

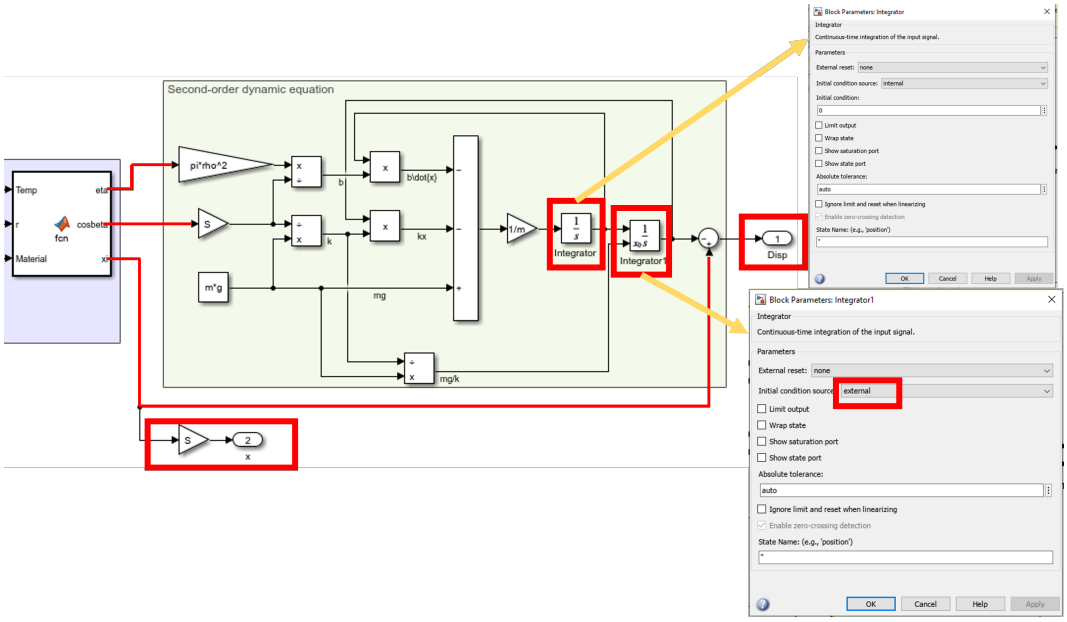

Figure 12: Second-order dynamic equation in SIMULINK

6. Figure 12 shows the implemented model as a screenshot. Finally, remove the constant xi and connect the feedback x from the Mechanical Model block back to the Thermal Model block, as in Figure 13. This concludes

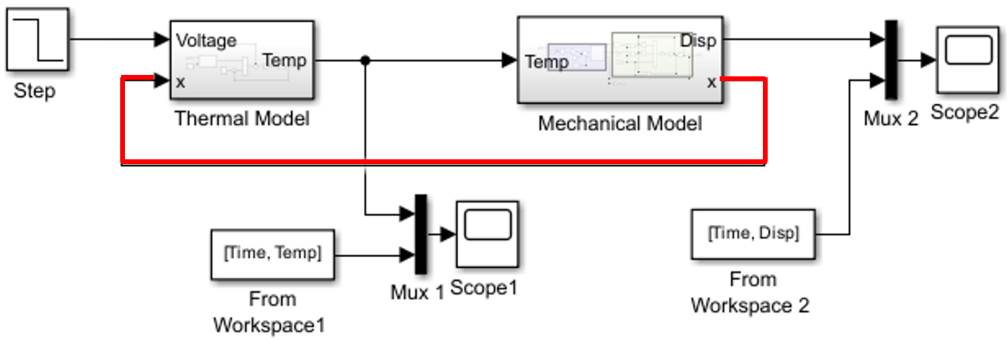

Figure 13: Completed Model

the implementation of the Carbon fiber theoretical model.

7. Only the variables need to be changed to model the simulation for Nylon TCAMs. This can easily be done by modifying the *SIM_parameters.m* script.

```
clear all                       % clear all in the workspace
load("ME6130_Dataset_Ny.mat")% load the dataset
T_sample = 0.05;                % Sample time [s]
V1 = 30;                        % exp voltage [V]
t1 = 4;                         % V1 on time [s]
R  = 95.0;                      % Resistance [ohm]
Ra = 20e6;                      % Rayleigh number for the characteristic length
Tamb= 25;                       % Ambient temperature [C]
Ct = 0.65;                      % Thermal mass [J/C]
xi = 0.1218;                    % Length of the coiled TCAM [m]

Material = 2;                   % 1 for Carbon fiber and 2 for Ny
CTE = 2.4*10^(-4); %            % Coefficient of Thermal Expansion [1/C]
r0 = 0.4e-3;                    % Initial fiber radius [m]
S  = 0.50;                      % Initial fiber length [m]
n = 240;                        % number of coiled turns
m = 400e-3;                     % Fabrication mass [kg]
g = 9.81;                       % gravity [m/s^2]
Ex = 1.408e10;                  % Longitudinal Young s modulus [N/m^2]
Gyz= 2.18e8;                    % Shear modulus [N/m^2]

q = 2*pi*n/S;
B_ = Ex*pi*r0^4/4;              % Bending stiffness [Nm^2]
C_ = Gyz*pi*r0^4/2;             % Torsional stiffness [N/m^2]
cosbeta_ = 1-(1/(B_-C_))*(q*B_*C_/sqrt(B_*m*g)-2*C_);
rho = sin(acos(cosbeta_))/(m*g/B_)^0.5; % Initial coil radius of the coil
```

The result plots should be as in Figure 14 .

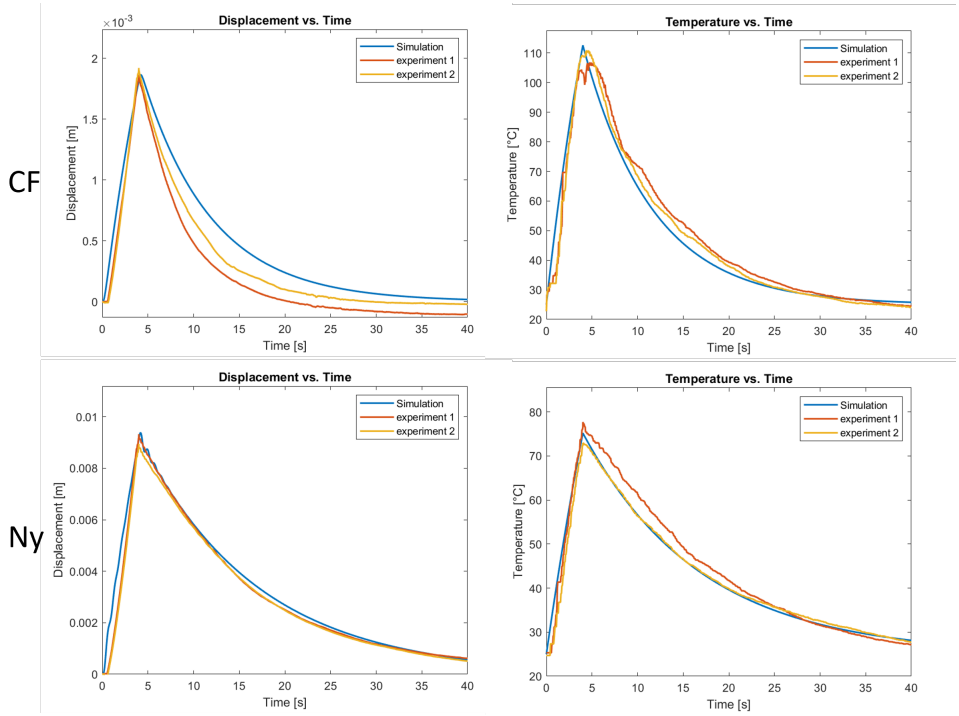

Figure 14: Sample Plots

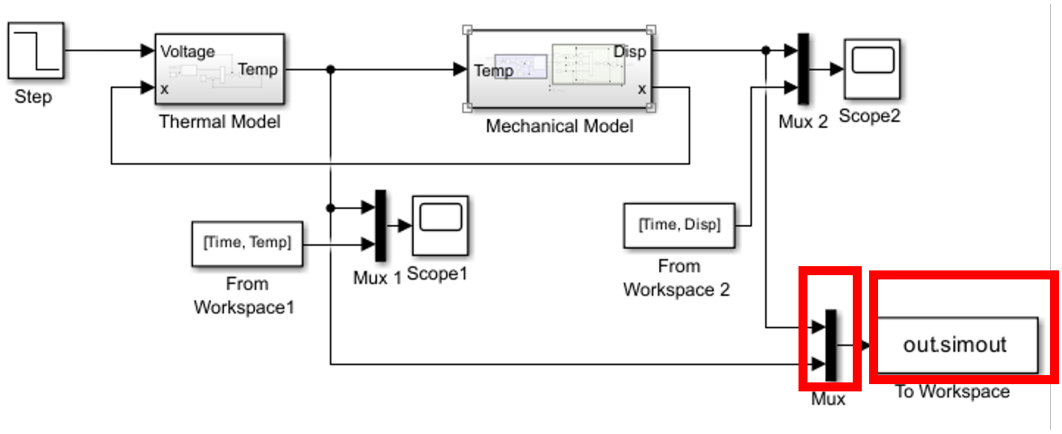

Figure 15: Addition of "to Workspace" block for MATLAB data export

8. If you want to plot your results using MATLAB, you may need to send the simulation results back to the workspace. this can be done by following modifications to the SIMULINK code, as in Figure 15. Then run the *plotting.m* file.

```
fig1 = figure(1);
plot(out.simout.Time, out.simout.Data(:,1),LineWidth=1.5)
hold on
plot(Time,Disp,'-',LineWidth=1.5)
hold off
legend('Simulation','experiment 1','experiment 2')
xlabel('Time [s]')
ylabel('Displacement [m]')
ylim([min(Disp(:,1))*1.2 max(Disp(:,1))*1.2])
title('Displacement vs. Time')

fig2 = figure(2);
plot(out.simout.Time, out.simout.Data(:,2),LineWidth=1.5)
hold on
plot(Time,Temp,'-',LineWidth=1.5)
hold off
legend('Simulation','experiment 1','experiment 2')
xlabel('Time [s]')
ylabel('Temperature [C]')
ylim([20 max(Temp(:,1))*1.1])
title('Temperature vs. Time')

if Material ==1
    print(fig1,'disp_CF','-dpng')
    print(fig2,'temp_CF','-dpng')
else
    print(fig1,'disp_Ny','-dpng')
    print(fig2,'temp_Ny','-dpng')
end
```