

Solving ODEs with MATLAB and Simulink:

Objectives:

- Learn how to solve ODEs with m-files and Simulink
- Learn how ODEs are numerically solved, and what this different terminology is
- Learn troubleshooting tips for when
 - The solution gets “stuck”
 - How to assess if the solution is actually the “real” solution
- Learn about state-space form and how it is necessary for solving ODEs
- Re-write a system of ODEs in a form that does not require you to analytically solve them

This is a simulation only lab – no hardware is needed.

Contents:

1. Solving an ODEs with Simulink
 - This is not the quickest way to get a solution, but it is the most generic tool for the job.
 - It takes a little longer to set up the problem, but it provides information about the system architecture, and can help in determining states for state-space form.
 - Is usually necessary when considering nonlinearities and control systems (or systems represented by block diagrams)
2. Solving ODEs with m-files
 - If your equations are in state space, or coefficient matrix form, this is the fastest way to get a solution.
3. How an ODE solver works
 - Explains the basic idea behind how ODEs can be numerically solved

Part 1: Solving ODEs with Simulink

Consider the ODE for a simple pendulum with specified initial conditions. The input is a torque T , and known parameters m, g, L, I .

$$I\ddot{\theta} = mgl \sin \theta + T$$

The general steps are:

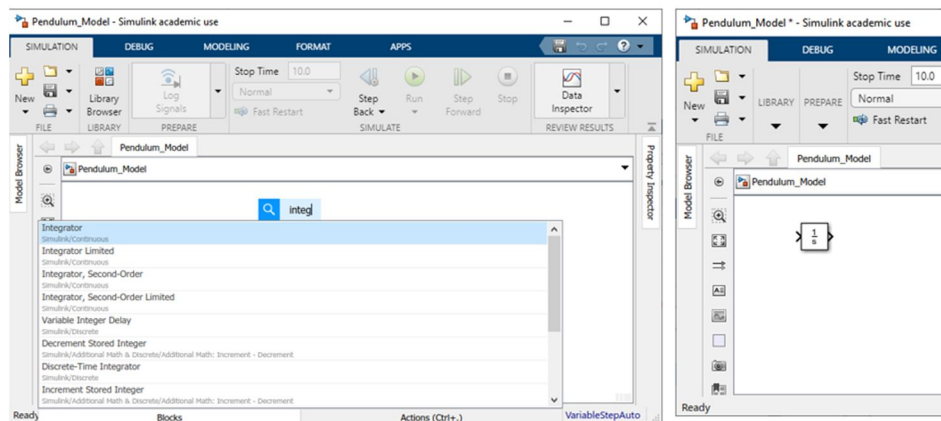
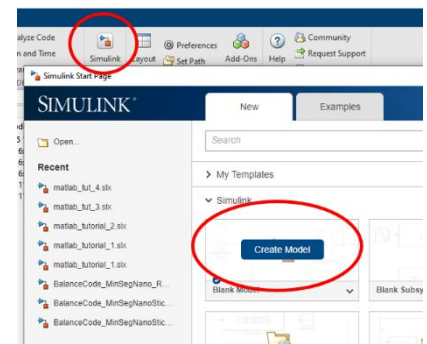
- Write these equations in Simulink
- Specify solver settings
- Define parameters, simulate (solve) the equations
- Plot the results

Writing the ODE in Simulink

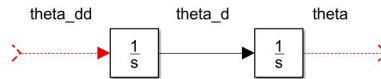
This equation will be constructed from various Simulink blocks. First is to solve the equation for the highest derivative, in this case $\ddot{\theta}$, then write this equation in Simulink.

$$\ddot{\theta} = \frac{1}{I}(T + mgl \sin \theta)$$

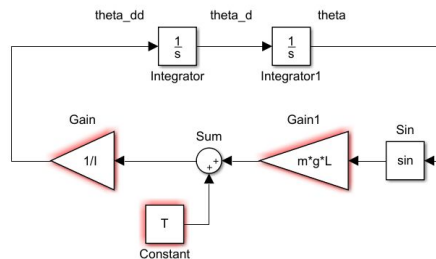
Open Simulink and create a blank new model. Save the model as “Pendulum_model”. The easiest way to add blocks is to click in the main window and just start typing the name of the block you want, for example start typing “integrator” and then select the desired choice. Alternatively search through the available blocks with the Library Browser.



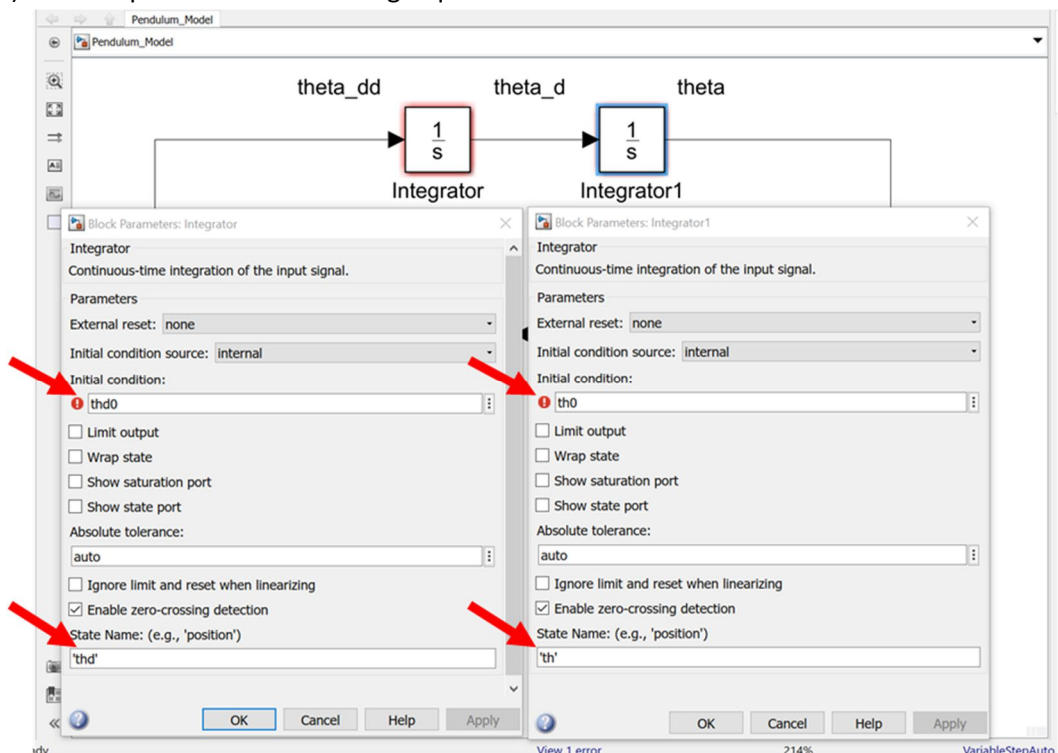
Blocks can be copied, or you can control-click and drag to create a copy. Add two integrators and connect them. Label each signal by typing the text and then pressing enter:



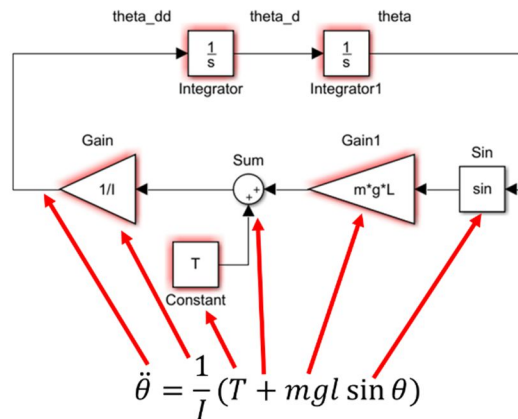
These are now the signals that are available. $\ddot{\theta}$ was integrated to obtain $\dot{\theta}$ and integrated again to obtain θ . Now the equation for $\ddot{\theta}$ can be written from the available signals:



- Show block names by selecting the blocks, right-clicking, and selecting “format”
- Red outlines indicates values for I, m, g, L, and T have not been defined. Simulink numerically integrates the equations so if it does not have numbers for these values it will can not solve the equation. These will be defined later.
- Rotate blocks by selecting the bock and clicking control+r
- Assign initial conditions by double-clicking each integrator. The first integrator on the left is the initial condition for velocity $\dot{\theta}(0)$ the second integrator on the right is the initial condition for position $\theta(0)$. The red indicates there is no value assigned yet. Label the output of this integrator block in the “State Name” (this is not necessary but naming the states will be helpful later). Be sure put the variable in single quotations:

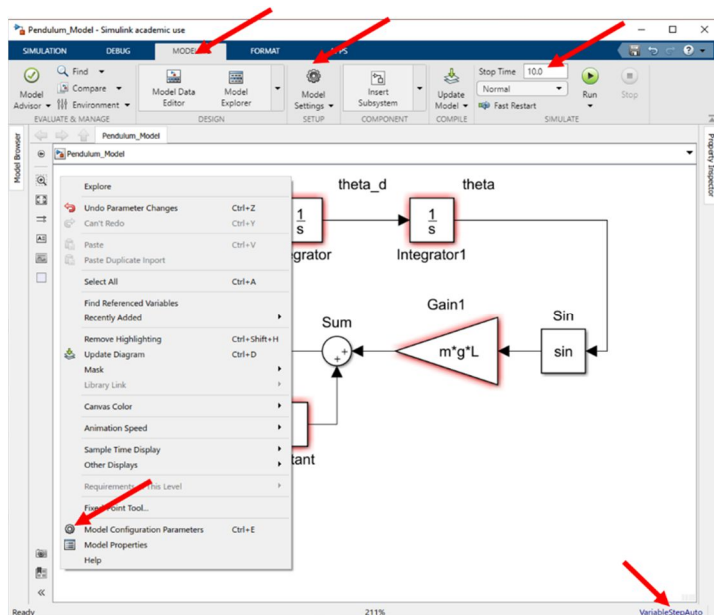


The ODE is now written in Simulink and the initial conditions have been specified. Notice that the equation can easily be “read” from the Simulink diagram:

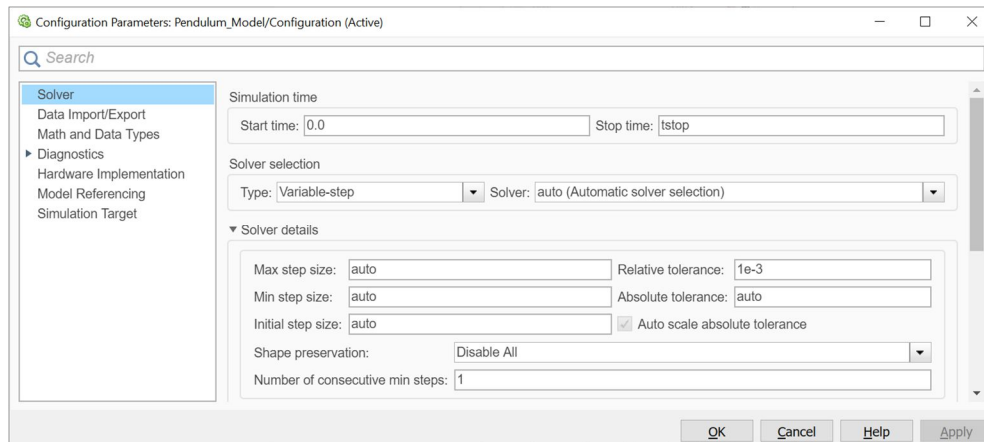


Specify Solver Settings

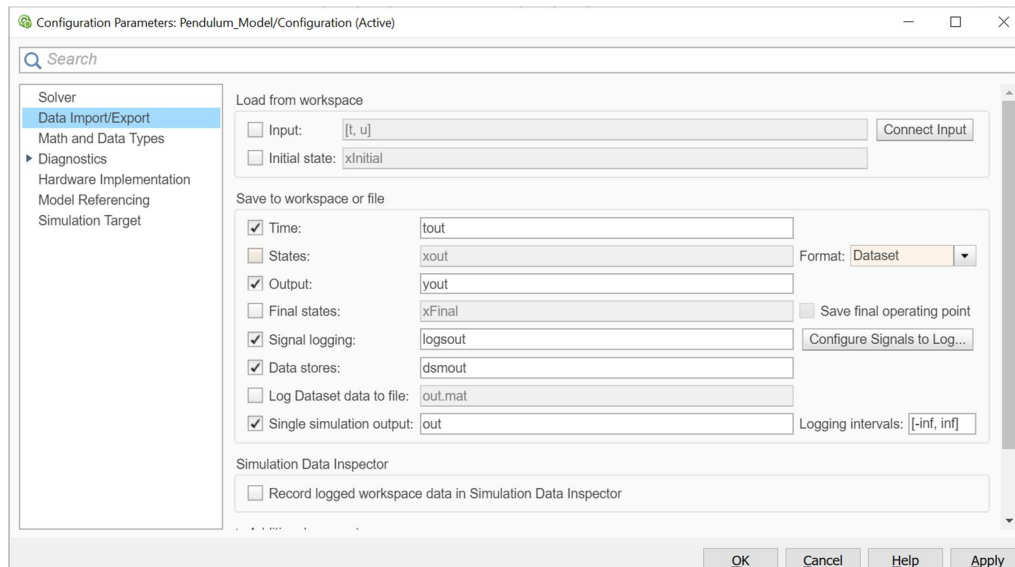
Access the solver settings by selecting the “Modelling” tab, clicking the gear icon for “Model Settings”. Or by right-clicking in the main diagram and selecting “Model Configuration Parameters” from there. The stop time and solver type are also indicated on the diagram.



Set the stop time to “tstop” and leave the other settings. The software will decide the time step size and solver to solve the ODE. These settings will be discussed later.



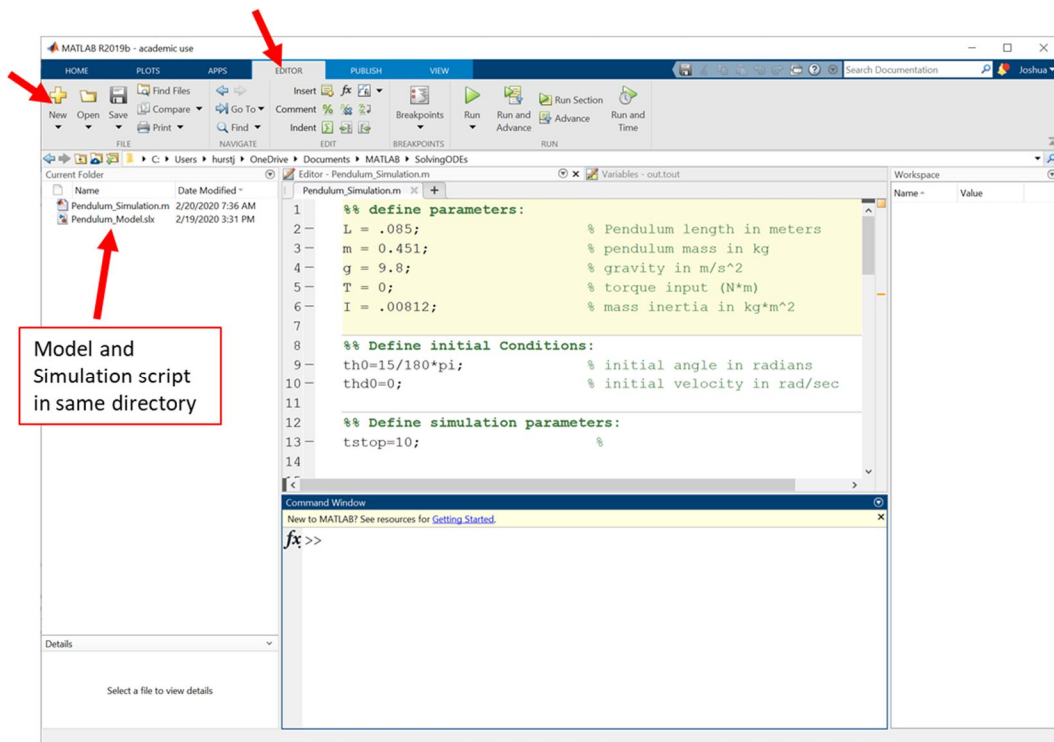
Select “Data Import/Export”:



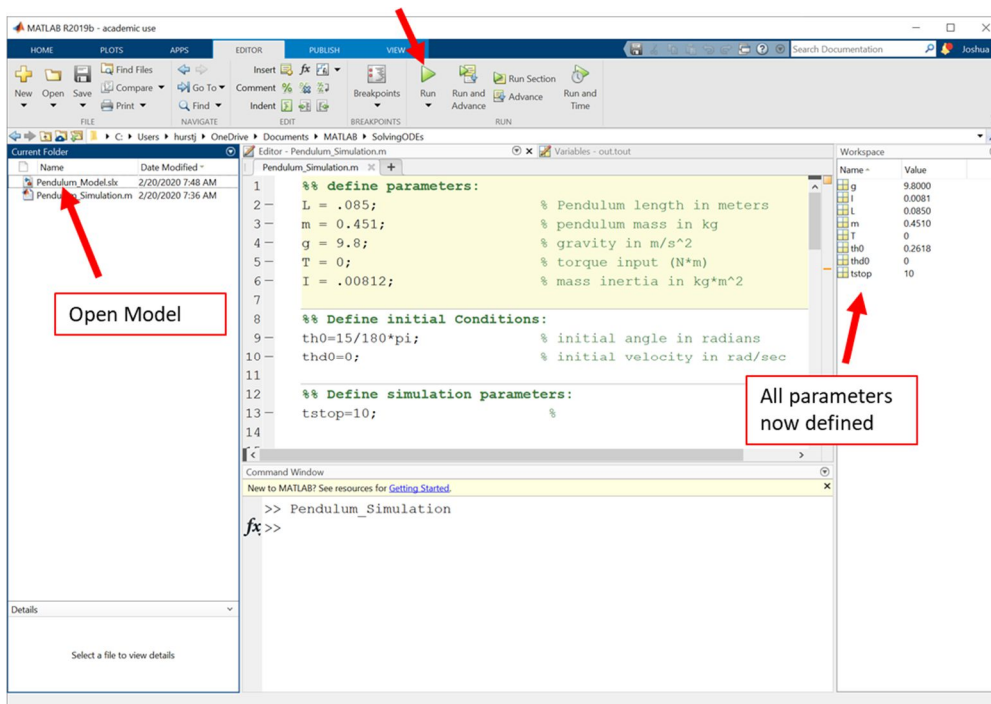
This shows what data Simulink will store. This will be modified later.

Define Parameters in a m-file and run simulation

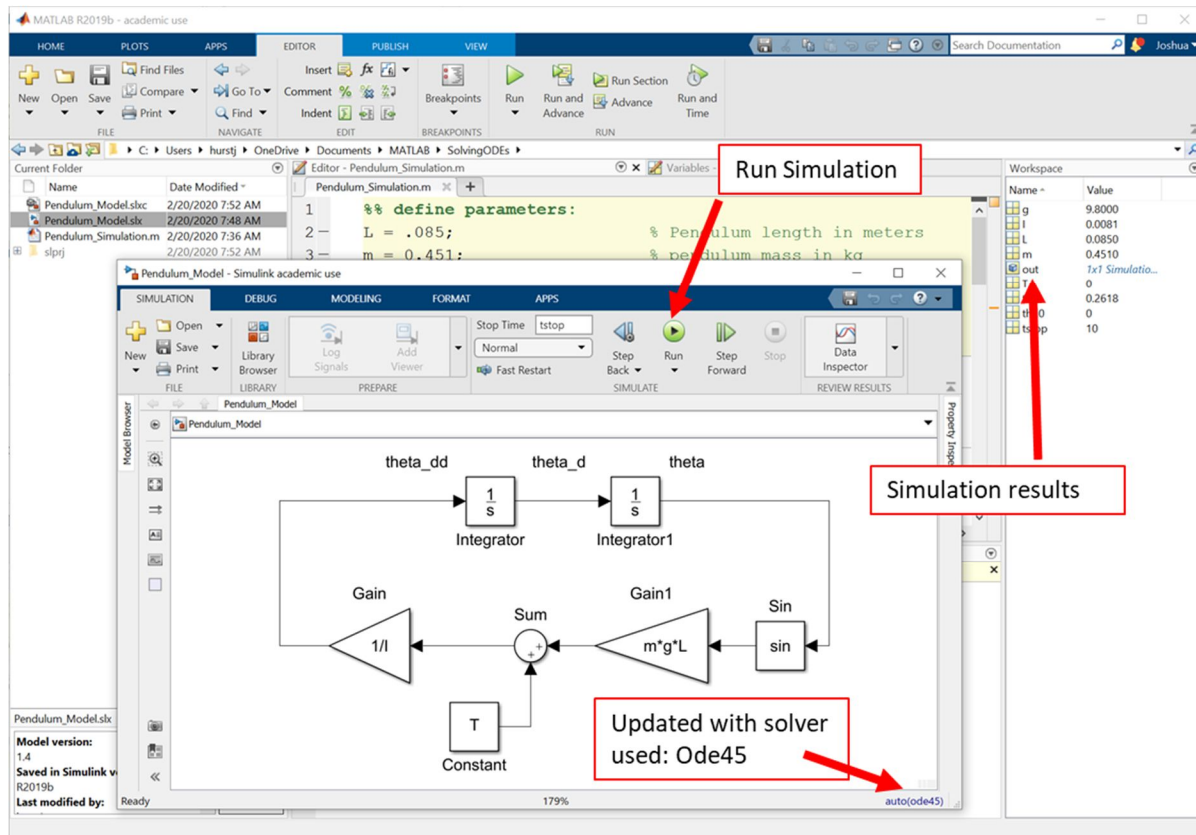
All the parameters (m, g, L, I), initial conditions ($th0, thd0$), inputs (T), and solver variables ($tstop$) need to be defined before the ODE can be solved. Matlab is a numerical tool and all variables must have number values. These values can be defined from the command line, but it is more efficient to define these in a m-file (script). From the Editor tab create a new script, save this file as “Pendulum_Simulation” and in this file enter all the parameters for the model and simulation.



Run the script and by pressing the green play button or with the keyboard shortcut F5 all the values will be defined for the simulation:



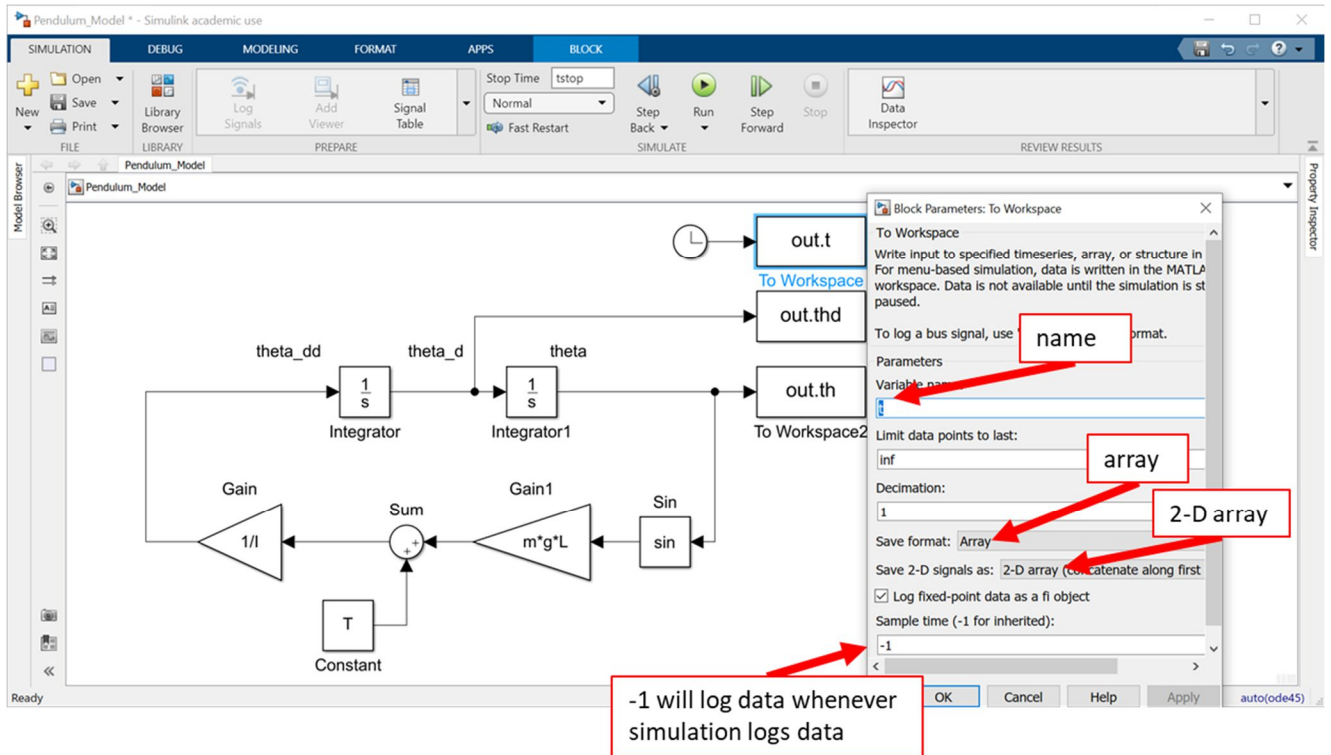
Open the model and now all the red highlighting is removed indicating all the parameters have been defined. Now the ODE can be solved by pressing the green Run button.



If an error message appears read all the information posted about the error. Usually it will provide you enough information to understand what the specific problem is. If it solves the ODE successfully it will store some information in the “out” variable and you can access this information by double clicking this variable. By default the usual outputs of interest θ and $\dot{\theta}$ are not logged.

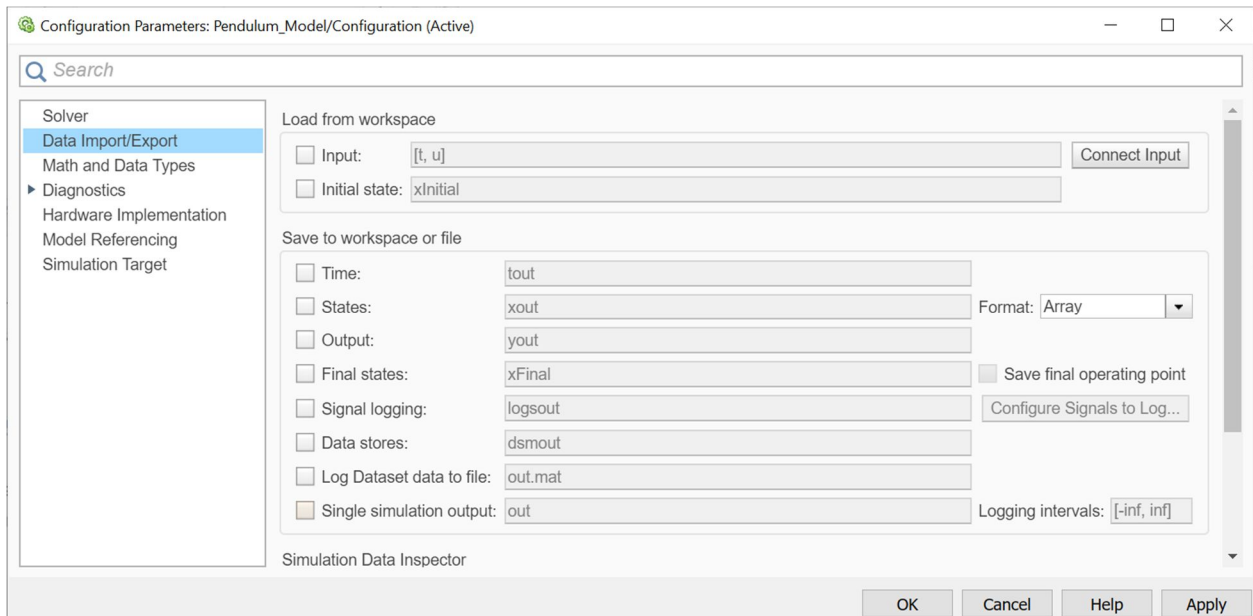
Define outputs, run simulation and plot the results

The simplest way to get access to the signals is to use “to workspace” blocks and store each signal in an array. Add a clock for time, and “to workspace” blocks for each of the integrator outputs (states) to log $t, \theta, \dot{\theta}$.

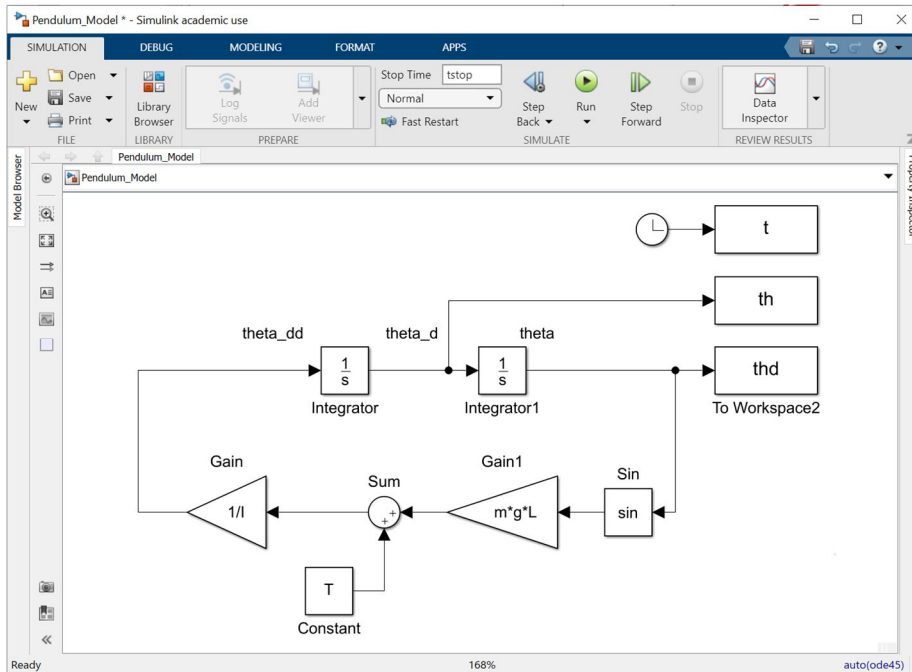


Update the script file to run the simulation diagram with the command “sim” and then plot the results:

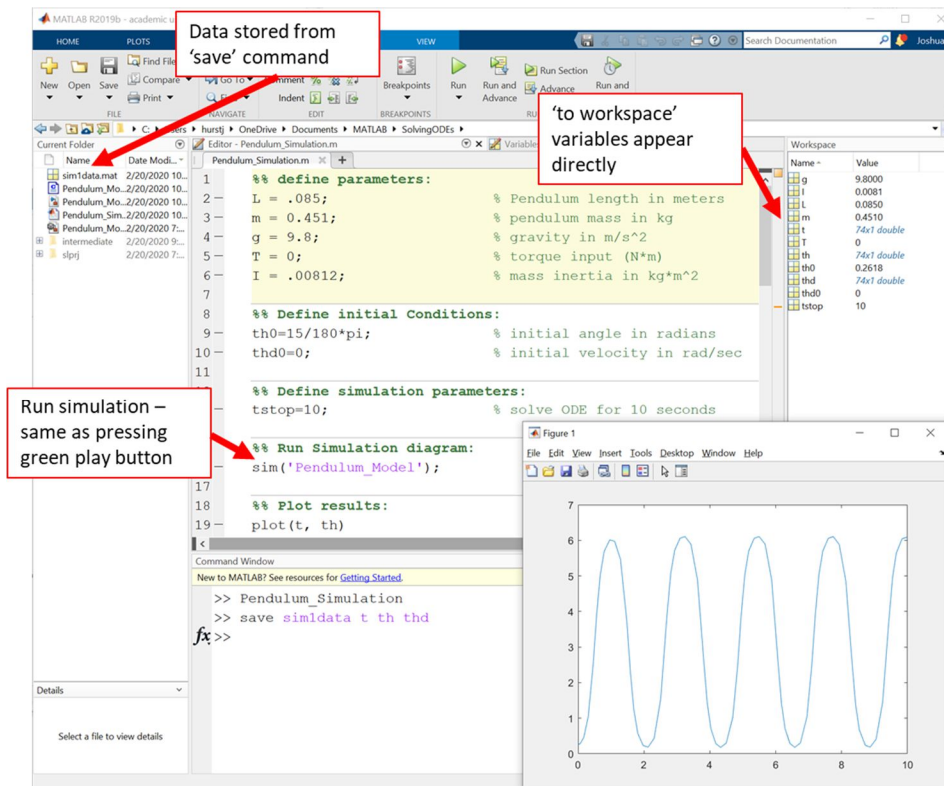
In the main simulation diagram press “Control+E” to open the model configuration parameters (or right-click and select “model configuration parameters”). Got to Data Import/Export and unselect all of the “Save to workspace commands” and click apply.



Now the data will be written directly to array variables specified in the to workspace blocks instead of the single simulation output “out”:



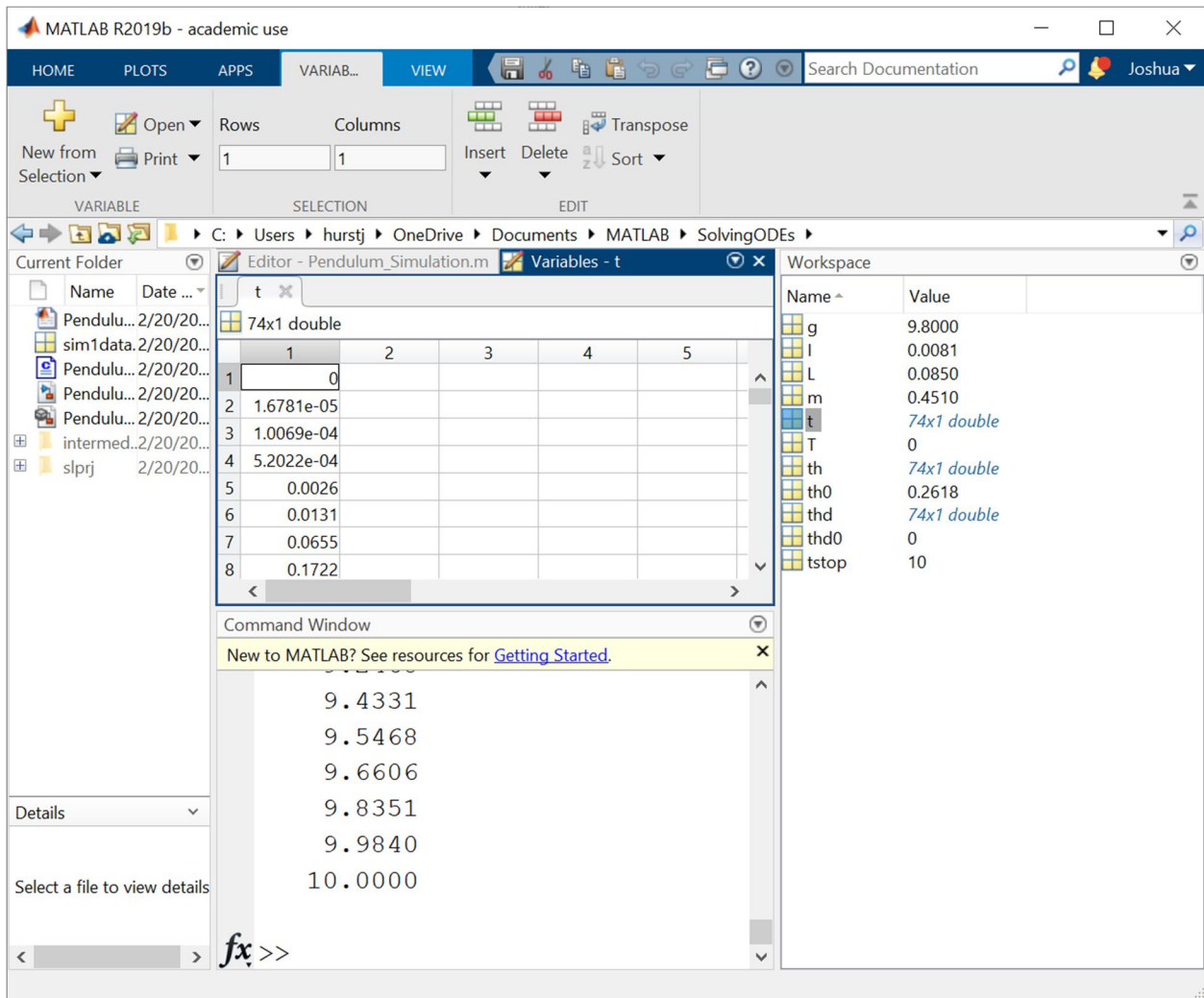
Edit the script file to run the simulation diagram, then plot the results:



Notice the “to workspace” variable appear directly in the workspace no and can then be easily plotted. Use the ‘save’ command to store the data (type ‘help save’ in the command line for more information)

Specifying the output interval – get more data

The plotted response is not as “smooth” as you might expect. The data points are accurate, but there may not be enough data points to plot a smooth curve. In the command line type “t” so see the vector for the variable “t” or double-click the variable in the Workspace.

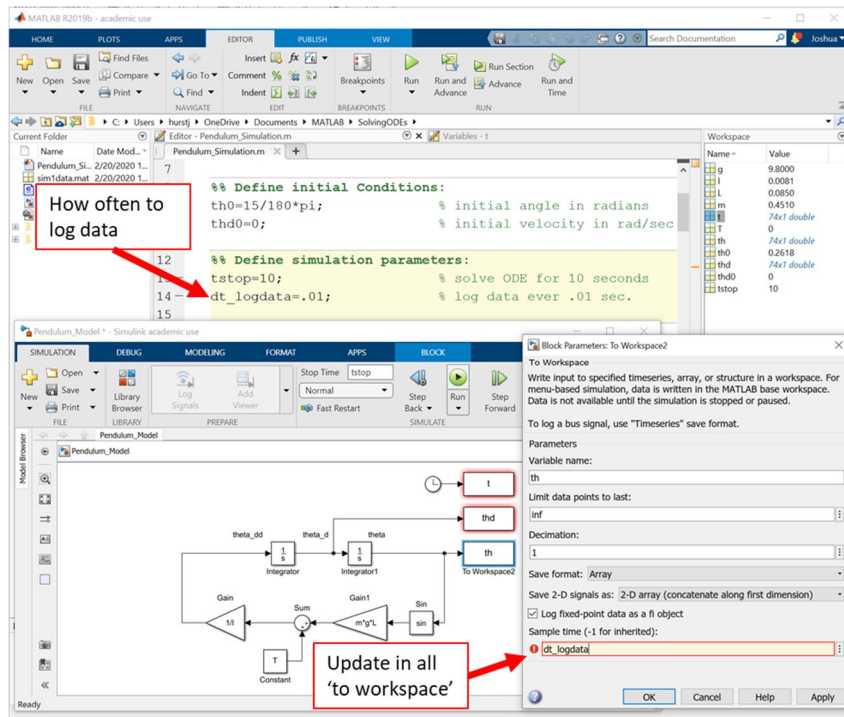


The screenshot shows the MATLAB R2019b interface. The Workspace pane on the right lists variables: g (9.8000), l (0.0081), L (0.0850), m (0.4510), t (74x1 double), T (0), th (74x1 double), th0 (0.2618), thd (74x1 double), thd0 (0), and tstop (10). The Command Window at the bottom shows the output of the 't' variable, which is a 74x1 double vector with values: 9.4331, 9.5468, 9.6606, 9.8351, 9.9840, 10.0000. The Editor pane shows the 'Pendulum_Simulation.m' file with the following code:

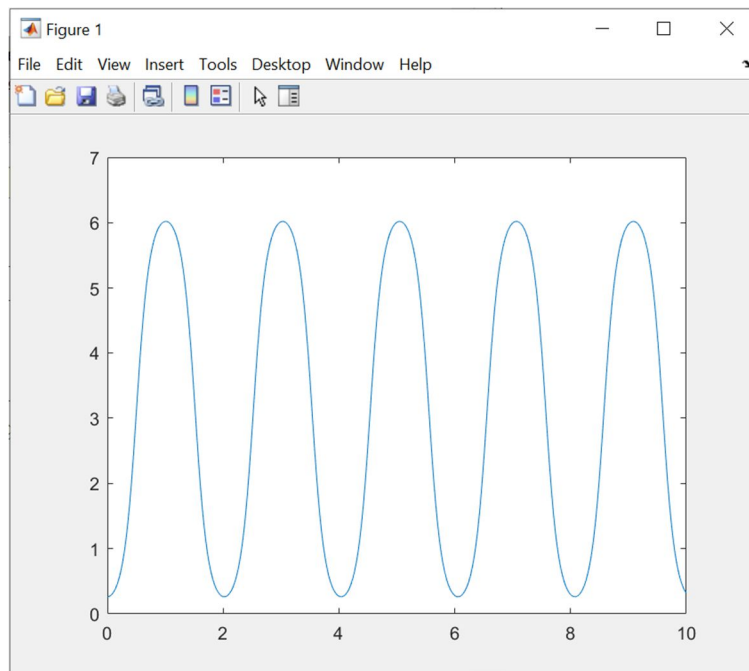
```
1 0
2 1.6781e-05
3 1.0069e-04
4 5.2022e-04
5 0.0026
6 0.0131
7 0.0655
8 0.1722
```

Notice that the returned points for t are not evenly spaced. The Solver returns data whenever it is convenient when it is calculating the solution. In the “to workspace” blocks the sample time was set to “-1” which means data will get logged whenever the solver decides to log data.

To obtain data at a specified interval define that interval in the script and update the sample time in the “to workspace” block with this value:



Now run the script and notice the smooth response:



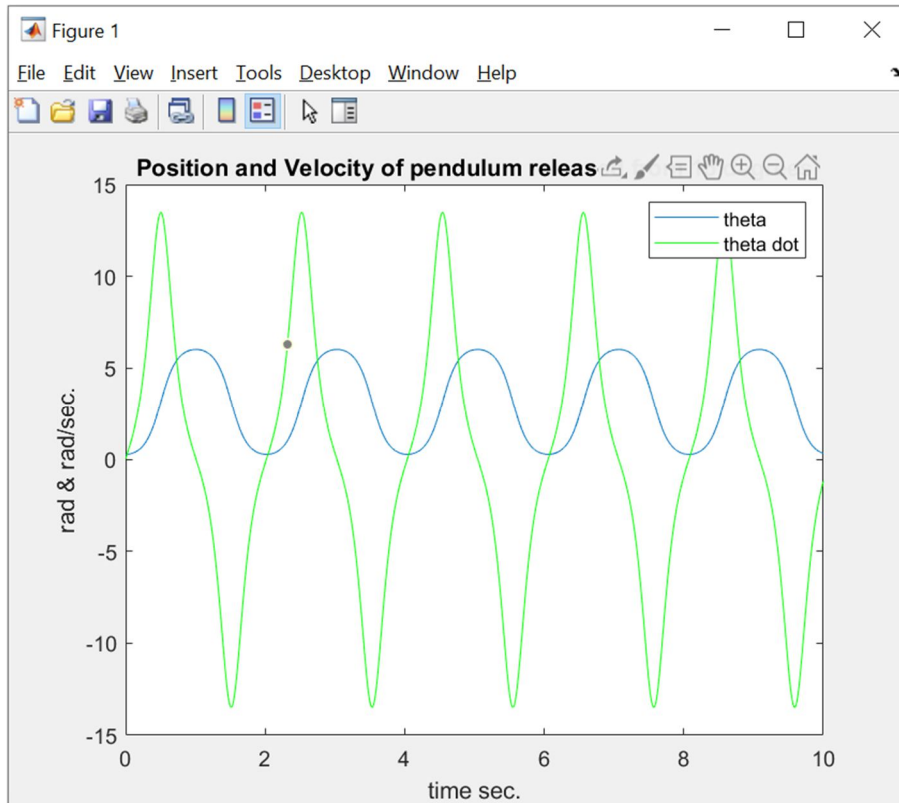
Basic Plotting commands

Modify the script to add legend, and labels to the response:

```

%% Plot results:
plot(t, th)
hold on
plot(t, thd, 'g')
legend('theta','theta dot')
ylabel('rad & rad/sec.')
xlabel('time sec.')
title('Position and Velocity of pendulum released from 15 degrees')

```



Part 2: Solving ODEs with a m-file

The general steps are:

- Convert the ODE to state space form
- Write the state space equations in a function m-file
- Write a script m-file that will
 - Define parameters, simulate (solve) the equations
 - Plot the results

Convert the ODE to state space form

The general steps to obtain a state space model are:

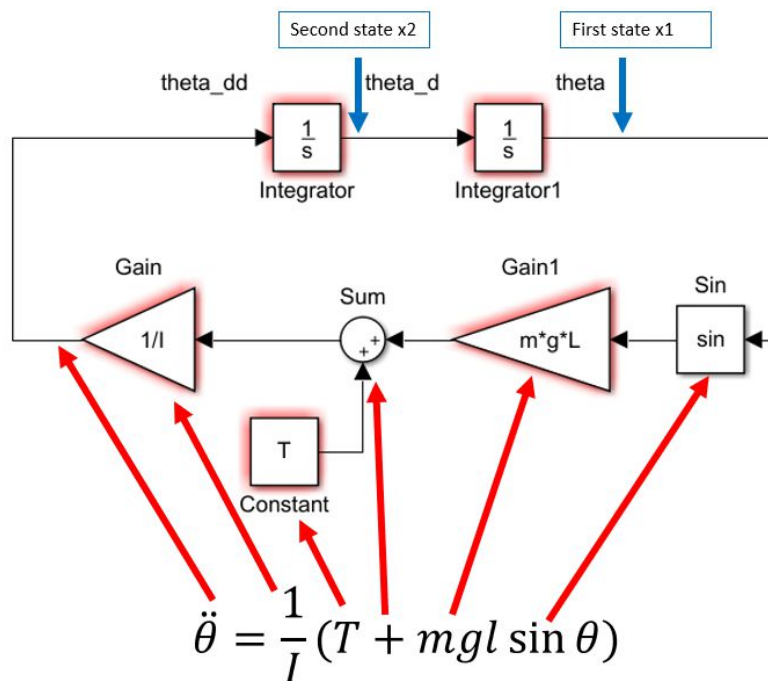
1. Define the states
2. Take the derivative of each state, and then express this only in terms of the states and the input
3. Put these equations in matrix form $\dot{x}=f(x,u)$ where u is the input

There are different methods to determine an appropriate state space form, here the previous Simulink diagram is used to help find the state space model.

In the previous section the original ODE,

$$I\ddot{\theta} = mgL \sin \theta + T$$

was “written” in Simulink by solving for the highest derivative and writing this in Simulink:



Following the general steps:

1. The states can be defined as the output of integrators in the Simulink diagram, in which we would have two states

$$\begin{aligned}x_1 &= \theta \\x_2 &= \dot{\theta}\end{aligned}$$

2. Taking the derivative of each state

$$\begin{aligned}\dot{x}_1 &= \dot{\theta} \\ \dot{x}_2 &= \ddot{\theta}\end{aligned}$$

Note this NOT in the form $\dot{x}=f(x,u)$. Meaning the only variable appearing should be states x and inputs u . To simplify this use the expressions in step 1, then the first equation becomes

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= \ddot{\theta}\end{aligned}$$

Now there needs to be a way to simplify $\ddot{\theta}$. There are no obvious substitutions, but the equation of motion provides an expression for $\ddot{\theta}$:

$$\ddot{\theta} = \frac{1}{I}(T + mgl \sin \theta)$$

But this can ONLY be expressed in terms of states and input so use the appropriate definitions in step 1 and realize the input $u = T$, then this equation becomes

$$\ddot{\theta} = \frac{1}{I}(T + mgl \sin \theta) = \frac{1}{I}(u + mgl \sin x_1)$$

The right hand side is now ONLY written in terms of states x and inputs u (and parameters which are just numbers).

3. Now the state space equations can be written in the correct vector form

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{I}(u + mgl \sin x_1)\end{aligned}$$

Which is in the correct vector form $\dot{x}=f(x,u)$, where the vector.

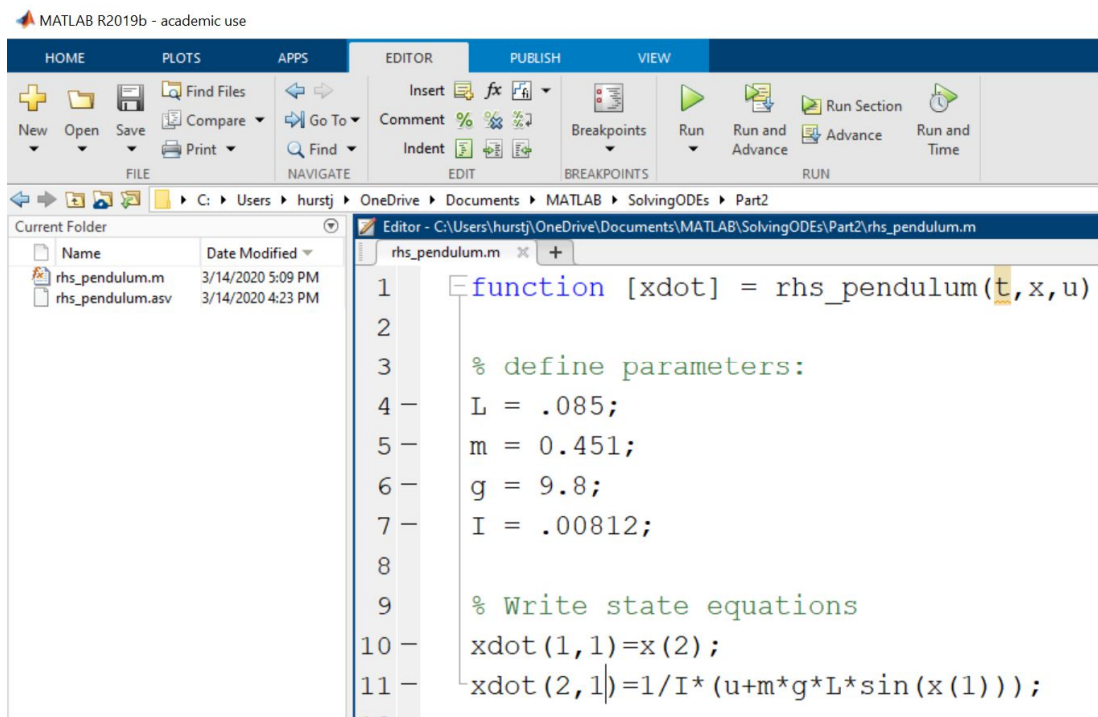
Write the State Space Equations in a function m-file

The original 2nd order ODE has been converted to two first order state-space equations:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{1}{I}(u + mgl \sin x_1) \end{aligned} \quad \longleftrightarrow \quad I\ddot{\theta} = mgl \sin \theta + T$$

These equations are now in the form $\dot{x} = f(x, u)$. Usually time t will not appear explicitly in these equations but it could so the more general form is $\dot{x} = f(t, x, u)$. MATLAB can be used to solve equations of this form.

Write the following function m-file:



```

MATLAB R2019b - academic use

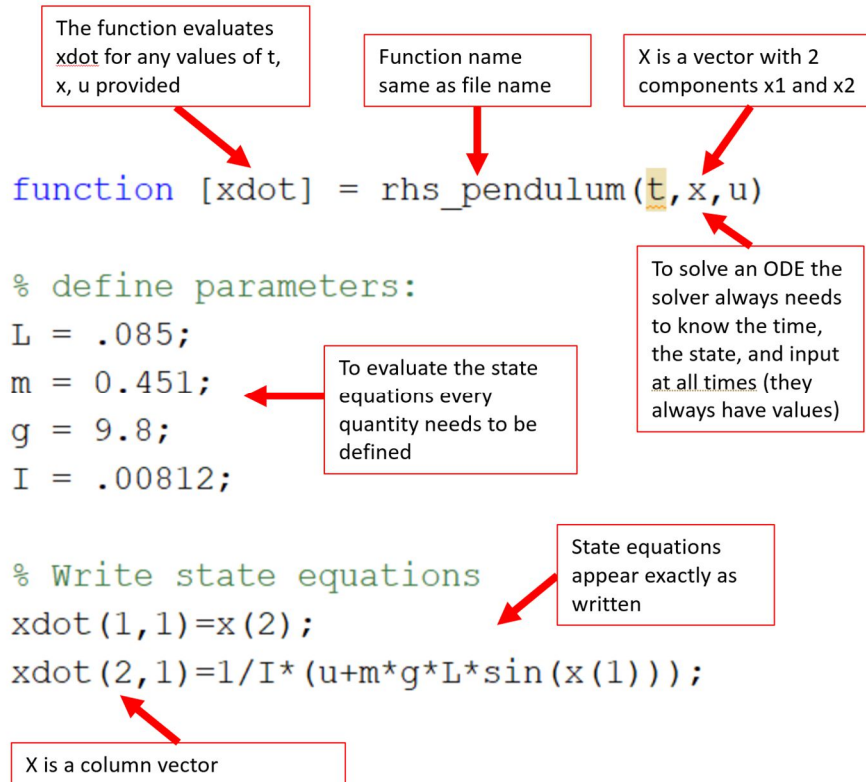
HOME PLOTS APPS EDITOR PUBLISH VIEW
New Open Save Find Files Compare Go To Comment % Indent Breakpoints Run Run and Run and
Print Find NAVIGATE EDIT BREAKPOINTS RUN ADVANCE Run and Time

C:\Users\hurstj\OneDrive\Documents\MATLAB\SolvingODEs\Part2
Current Folder
Name Date Modified
rhs_pendulum.m 3/14/2020 5:09 PM
rhs_pendulum.asv 3/14/2020 4:23 PM

Editor - C:\Users\hurstj\OneDrive\Documents\MATLAB\SolvingODEs\Part2\rhs_pendulum.m
rhs_pendulum.m
1 function [xdot] = rhs_pendulum(t,x,u)
2
3 % define parameters:
4 L = .085;
5 m = 0.451;
6 g = 9.8;
7 I = .00812;
8
9 % Write state equations
10 xdot(1,1)=x(2);
11 xdot(2,1)=1/I*(u+m*g*L*sin(x(1)));

```

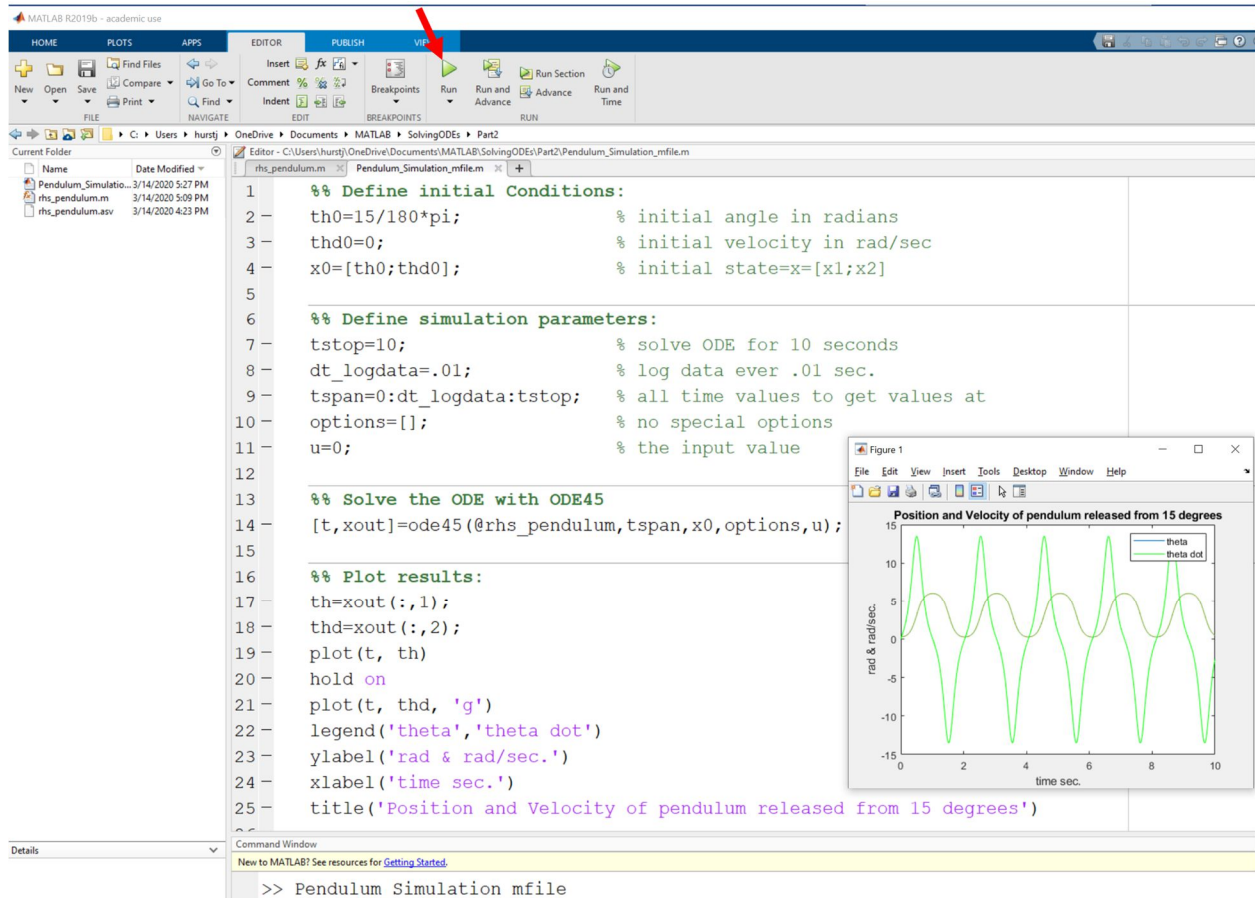
To solve an ODE MATLAB needs to be able to evaluate the state equations for any values of t , x , and u :



The 't' in the input is highlighted because it is not used anywhere. In this case it not, but MATLAB still needs the first argument to be time, the second the state vector, and the third is any other parameters or inputs you pass from the solver (see below).

Write a script m-file to solve and plot the ODE

Similar to the Simulink approach we need to specify the type of solver, initial conditions, the total time for the solution and the frequency at which data is returned. This is all defined in a mfile then press the green play button to run the m-file to solve and plot the ODE:



The difference from the previous Simulink script are

- Define the initial state from the initial conditions
- Use an ODE45 to solve the ODE
 - Note the '@' symbol placed before the name of the function that evaluates the state equations
- Extract the angle and position from the state solution $x(t)$

```

%% Define initial Conditions:
th0=15/180*pi;
thd0=0;
x0=[th0;thd0];

%% Define simulation parameters:
tstop=10; % solve ODE for 10 seconds
dt_logdata=.01; % how often data is returned (time vector)
tspan=0:dt_logdata:tstop; % at what time intervals to get values at
options=[]; % special options
u=0; % input value

%% Solve the ODE with ode45
[t,xout]=ode45(@rhs_pendulum,tspan,x0,options,u);

%% Plot results:
th=xout(:,1);
thd=xout(:,2);
plot(t, th)
hold on
plot(t, thd, 'g')
legend('theta', 'theta dot')
ylabel('rad & rad/sec.')
xlabel('time sec.')
title('Position and Velocity of pendulum released from 15 degrees')

```

Define the initial state from the initial values

How often data is returned (time vector)

Function that evaluates the state equations

special options

input value

Inputs or other parameters at the end

No options

Initial condition/ Initial state

Extract angle and velocity from the state vector

Part 1 & 2 Exercise

Solve the following ODE with Simulink and then with MATLAB using m-files

$$A\ddot{z} + B\dot{z} + Cz = D$$

Where $D = 1$ is the constant input, $A = 1, B = 2, C = 3$ are parameters and the initial conditions are $z(0) = 0, \dot{z}(0) = 4$. Plot $z(t)$, and $\dot{z}(t)$ for 5 seconds both on the same graph. Obtain data every 0.01 seconds.

Include the following files for submission:

- 1.) Simulink file and the m-file to run it
- 2.) M-file function and script to solve the ODE

