

Week 7 – Manipulator Dynamics: MATLAB Simulink Simulation

Advanced Robotic Systems – MANU2453

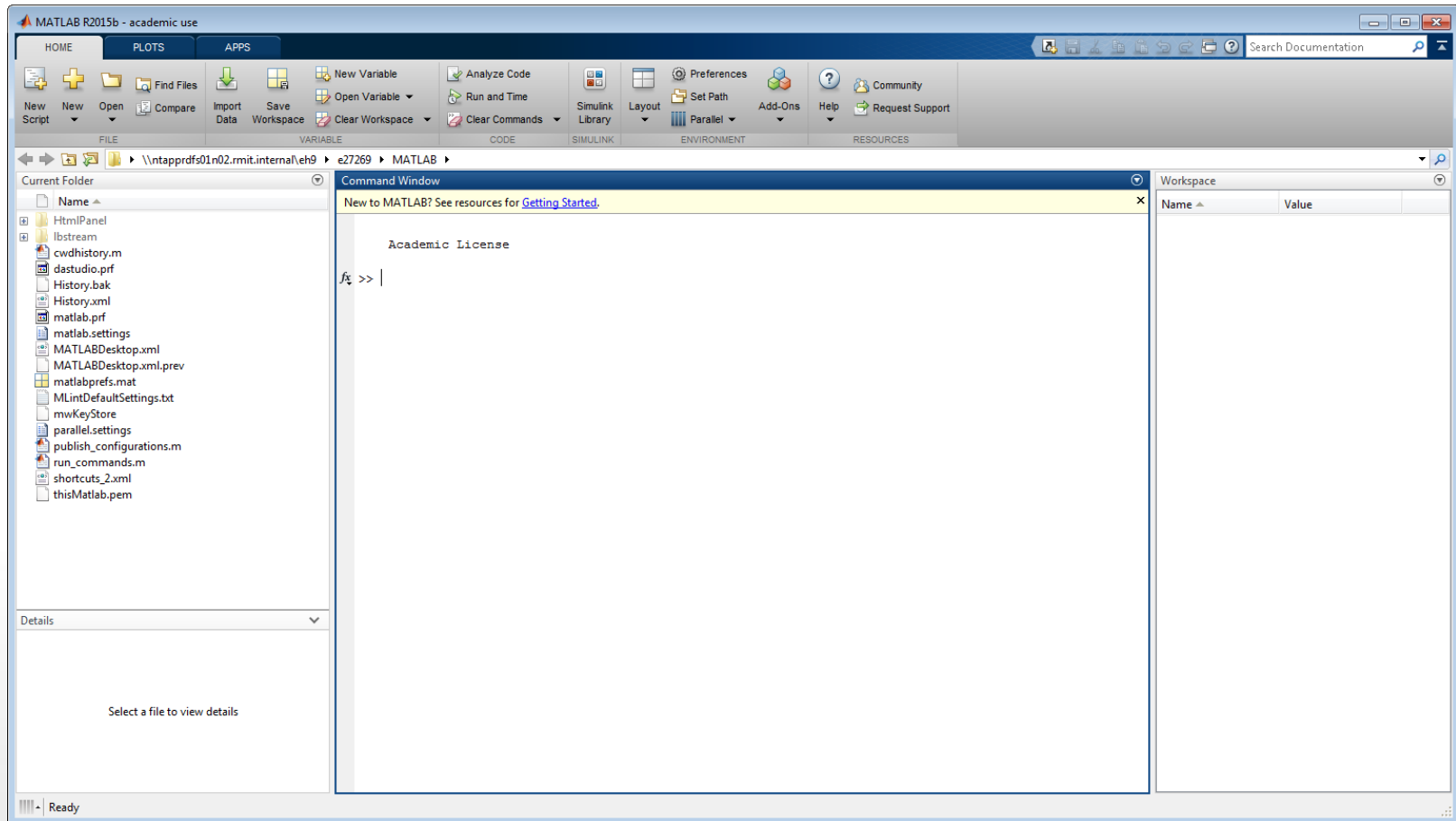
Dr Ehsan Asadi, School of Engineering
RMIT University, Victoria, Australia
Email: ehsan.asadi@rmit.edu.au

Simulation

- As mentioned in the introduction of Lecture 6, the dynamic equations / **equations of motion are useful for computer simulation**:
 - If we put in certain torques in the joints, **how will the robot moves** (how will the joints accelerate)?
 - In the next few weeks, you will also learn **how to change the torque**, such that the robot moves in a way you desire. (**Control problem**).
- We will use MATLAB Simulink for this purpose.

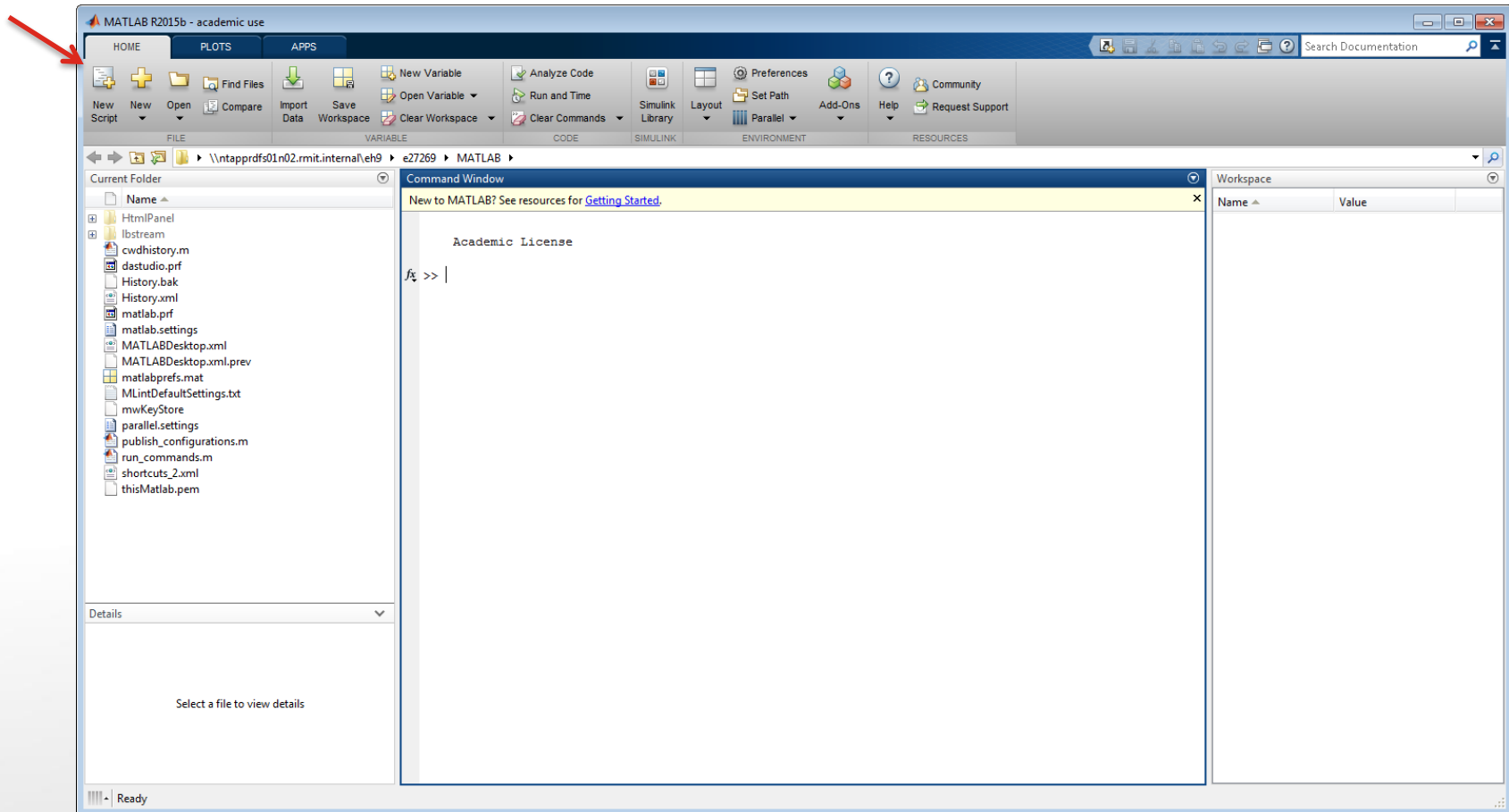
MATLAB

- First of all, start MATLAB.



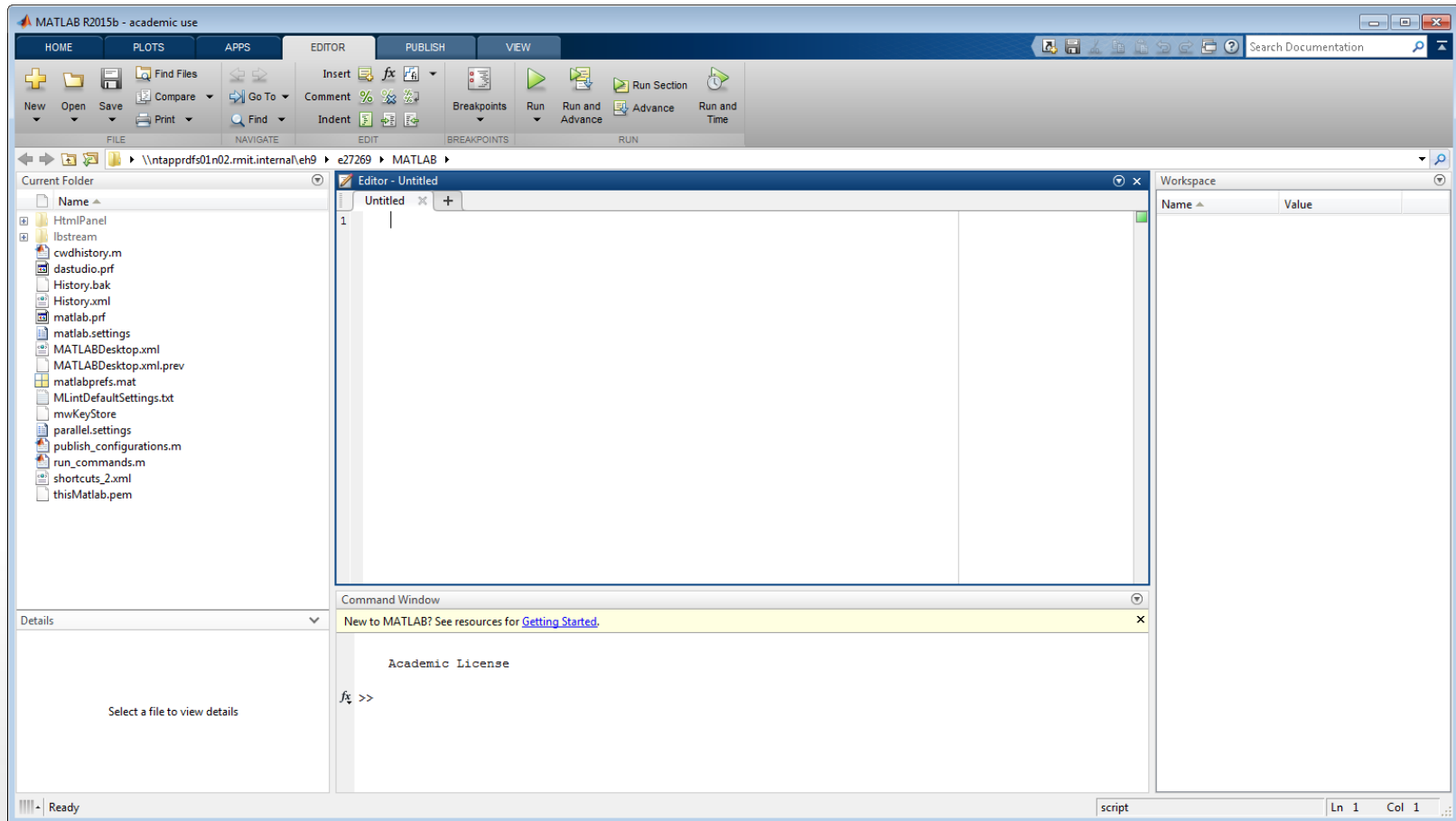
MATLAB

- We will create an m-file, where we can key in all the constant parameters.
- Click “New Script”.



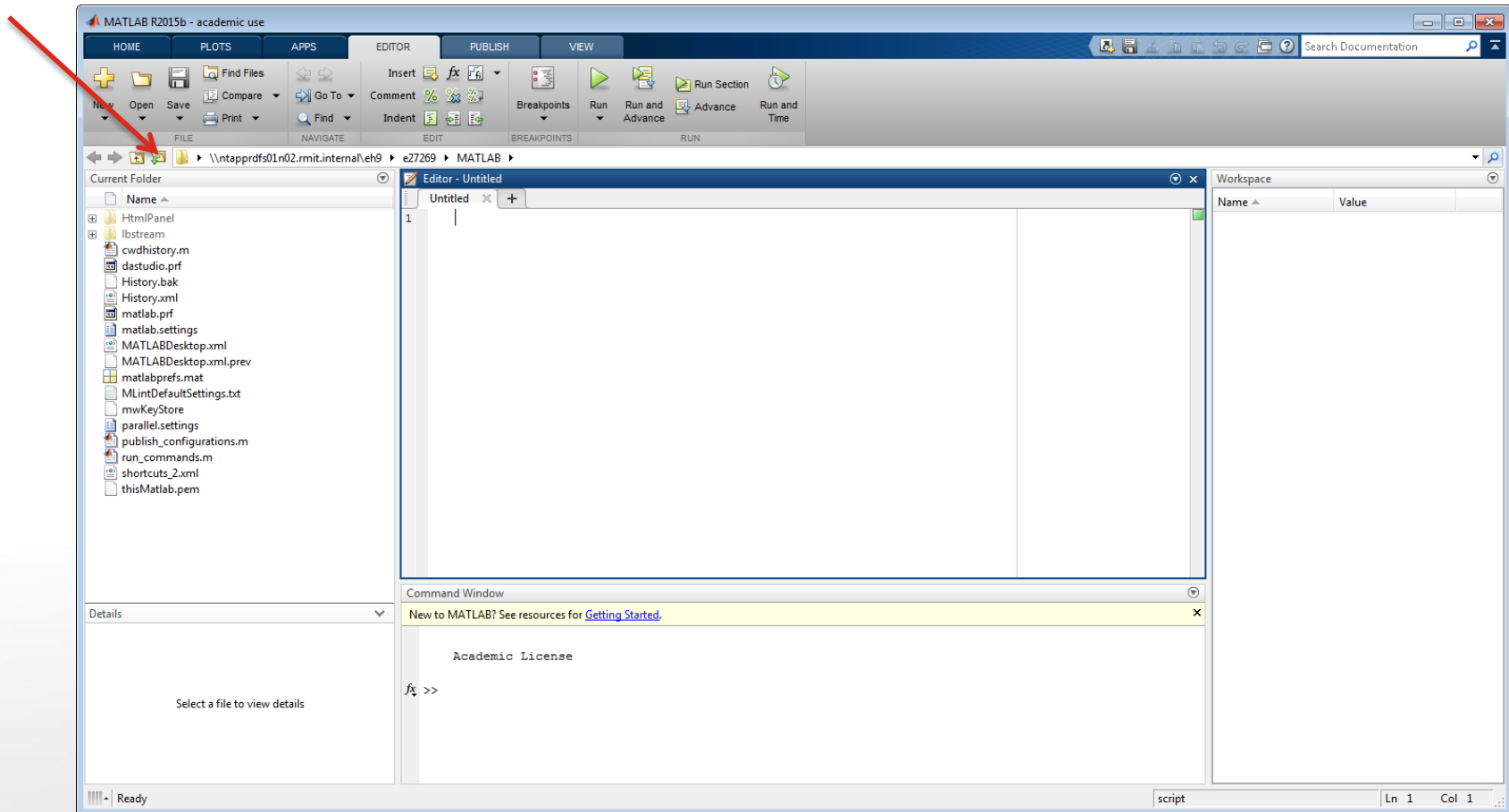
MATLAB

- An Editor will appear.
- Save it as “Parameters.m” on your desktop.



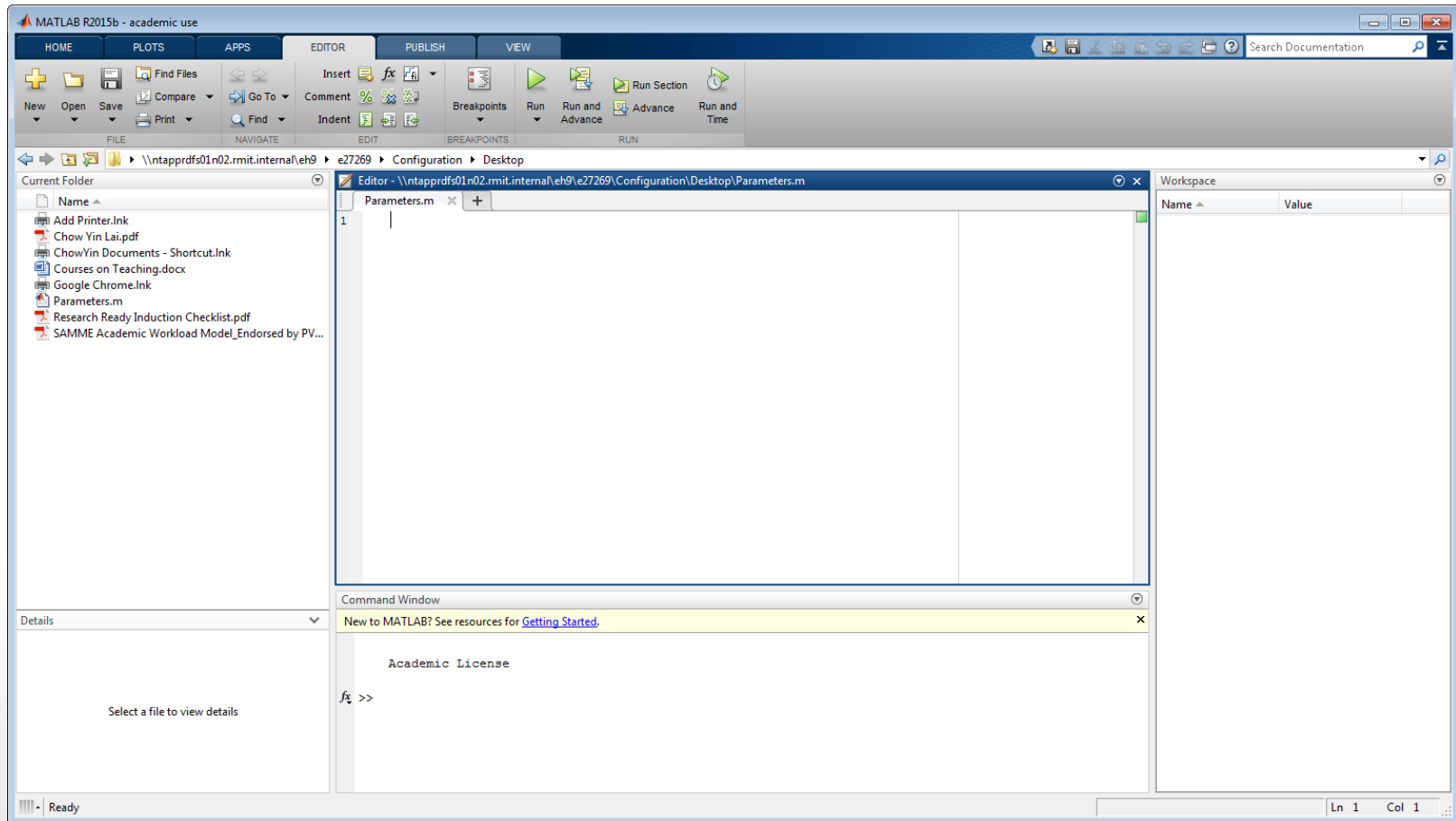
MATLAB

- At this moment, the working directory is the “MATLAB” folder.
- Change it to “Desktop” by clicking the shown button.



MATLAB

- Note that the directory is changed to Desktop.
- We will now start keying in the parameters for the robot.



MATLAB

- Let's use the 2-link robotic manipulator from Week 6 Lecture as example:

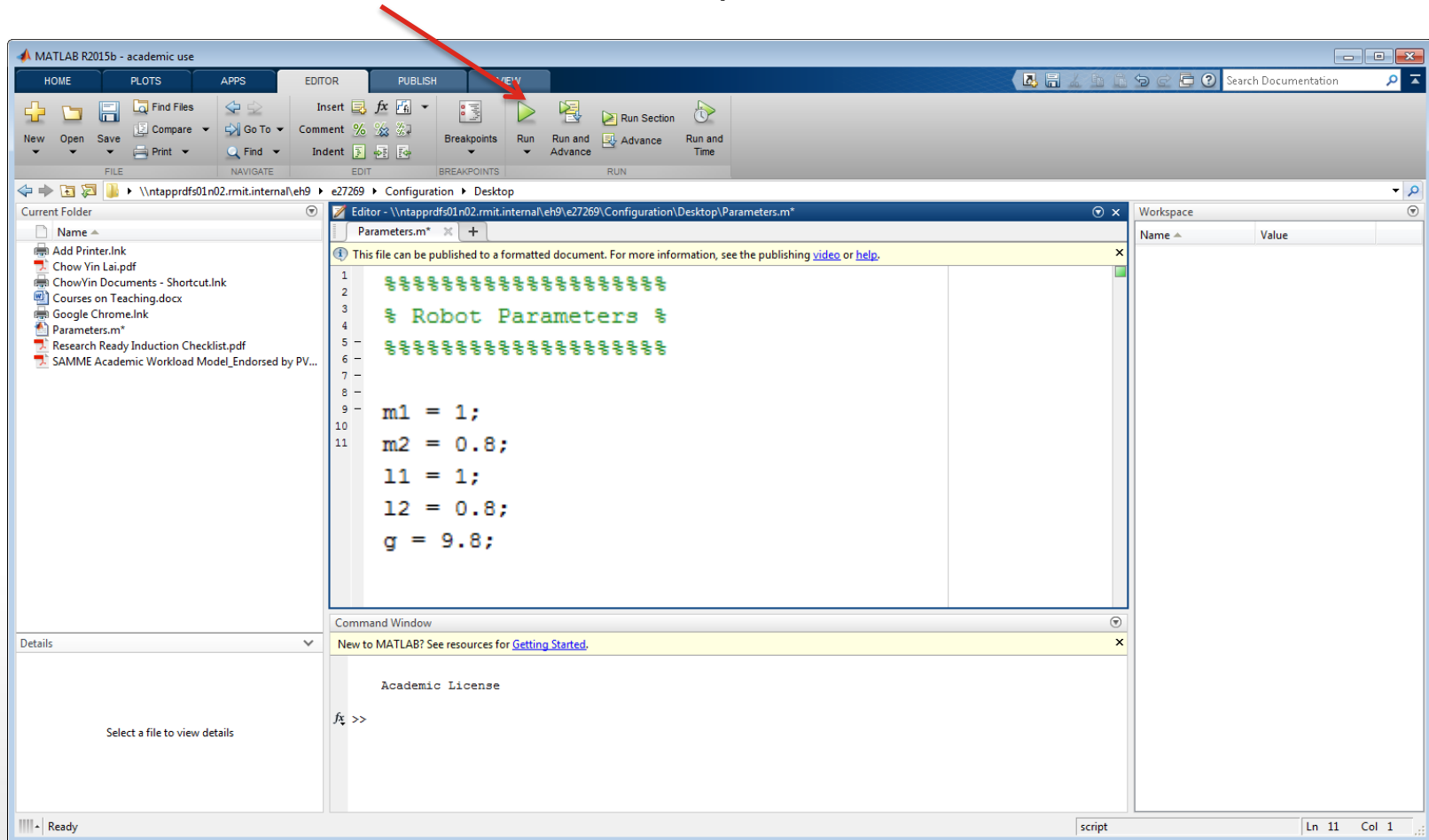
$$\underbrace{\begin{bmatrix} (m_1 + m_2)L_1^2 + m_2L_2^2 + 2m_2L_1L_2c_2 & m_2L_2^2 + m_2L_1L_2c_2 \\ m_2L_2^2 + m_2L_1L_2c_2 & m_2L_2^2 \end{bmatrix}}_{M(q)} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \underbrace{\begin{bmatrix} -m_2L_1L_2s_2\dot{\theta}_2^2 \\ m_2L_1L_2s_2\dot{\theta}_1^2 \end{bmatrix}}_{\text{Centrifugal}} + \underbrace{\begin{bmatrix} -2m_2L_1L_2s_1\dot{\theta}_1\dot{\theta}_2 \\ 0 \end{bmatrix}}_{\text{Coriolis}} + \underbrace{\begin{bmatrix} m_2gL_2c_{12} + (m_1 + m_2)gL_1c_1 \\ m_2gL_2c_{12} \end{bmatrix}}_{G(q)} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

$V(q, \dot{q})$

- The constant parameters are:
 - m_1
 - m_2
 - L_1
 - L_2
 - g

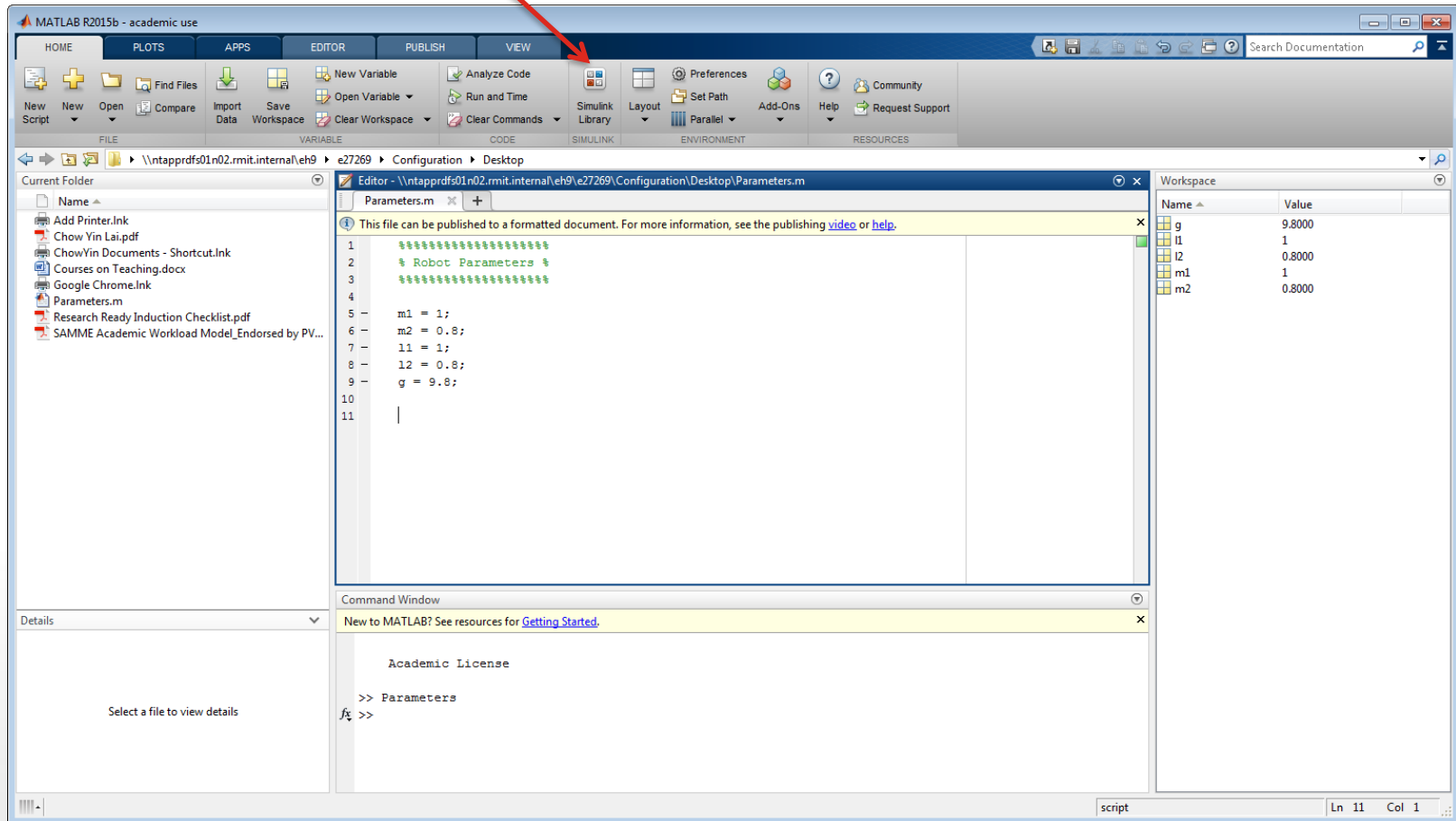
MATLAB

- Write these in (together with some assumed values) in the Editor.
- Then click the “Run” button on the top of window.



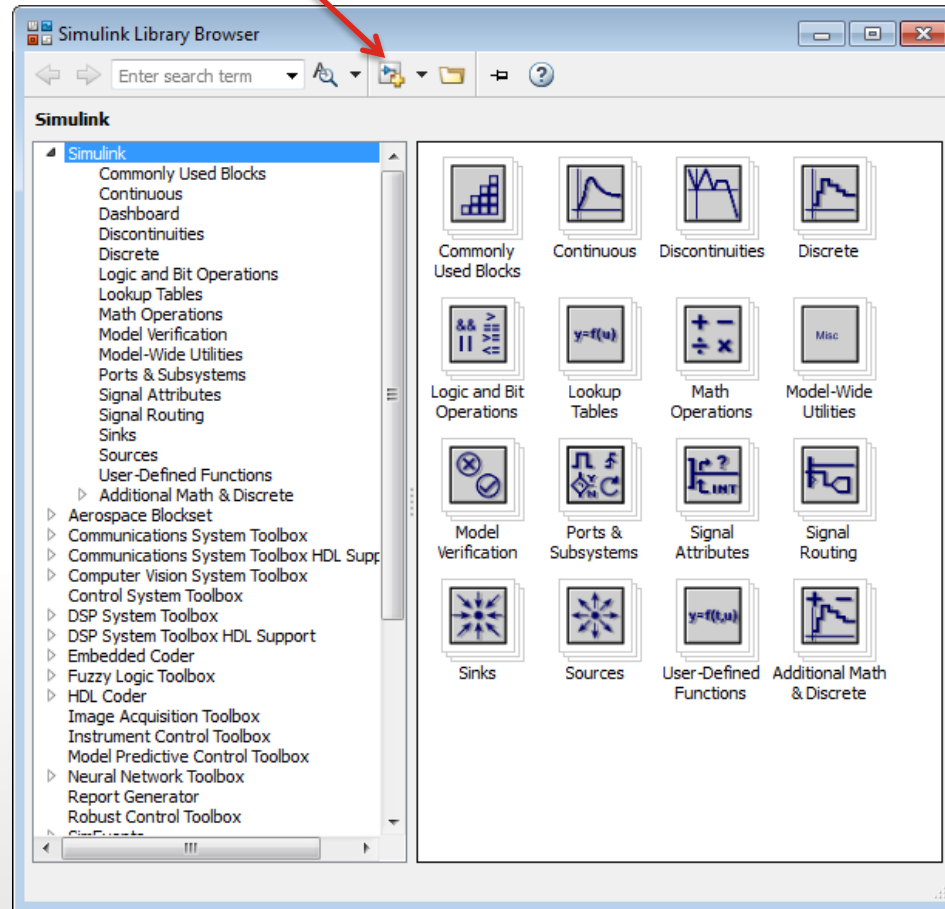
MATLAB

- Go to the “Home” tab of MATLAB, and click the “Simulink Library” button.



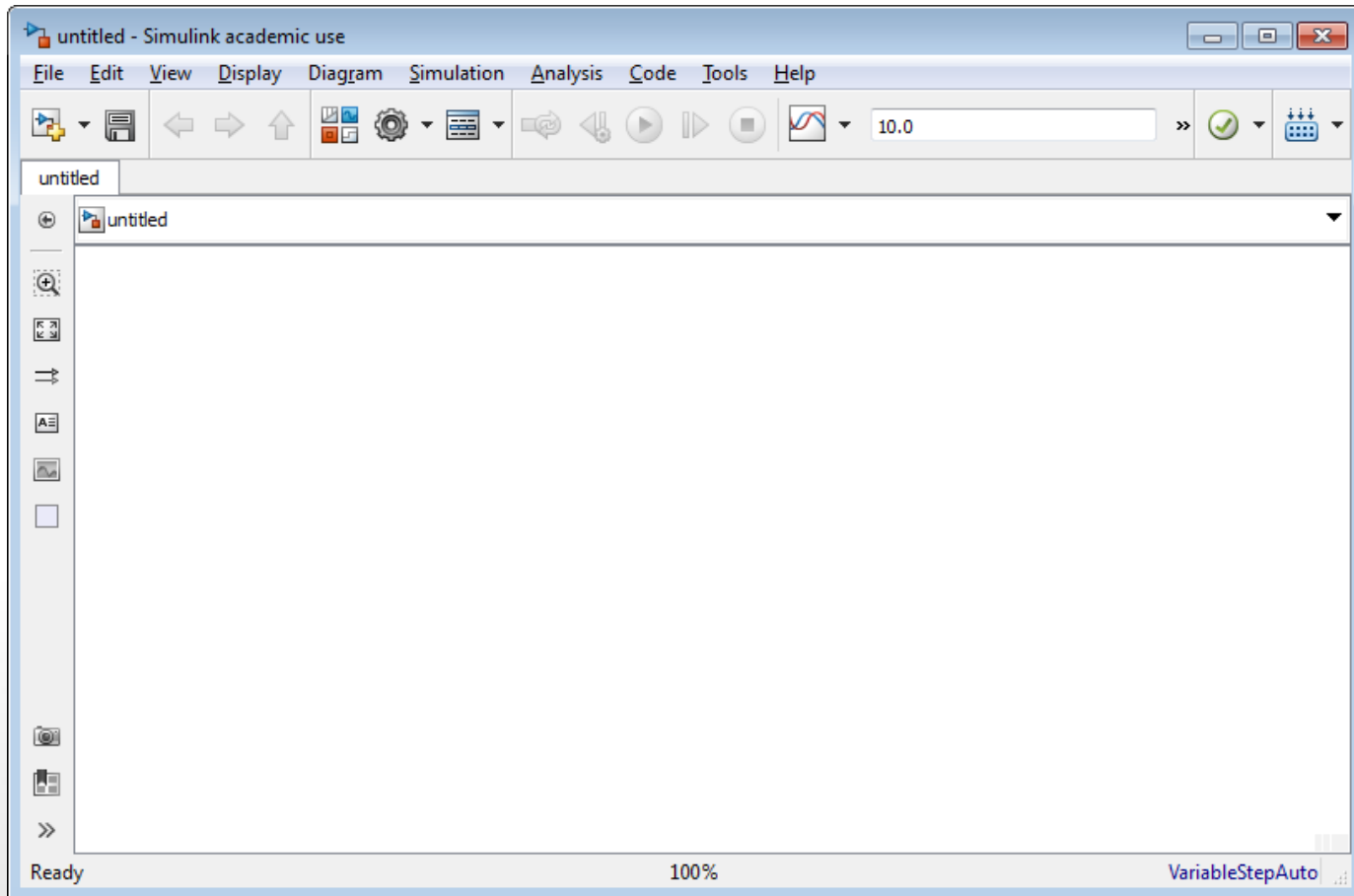
MATLAB / Simulink

- The Simulink Library will appear.
- Click the “+” button to open a new simulation window.



MATLAB / Simulink

- For the new simulation window (as shown), save it as “Simulation.slx”.

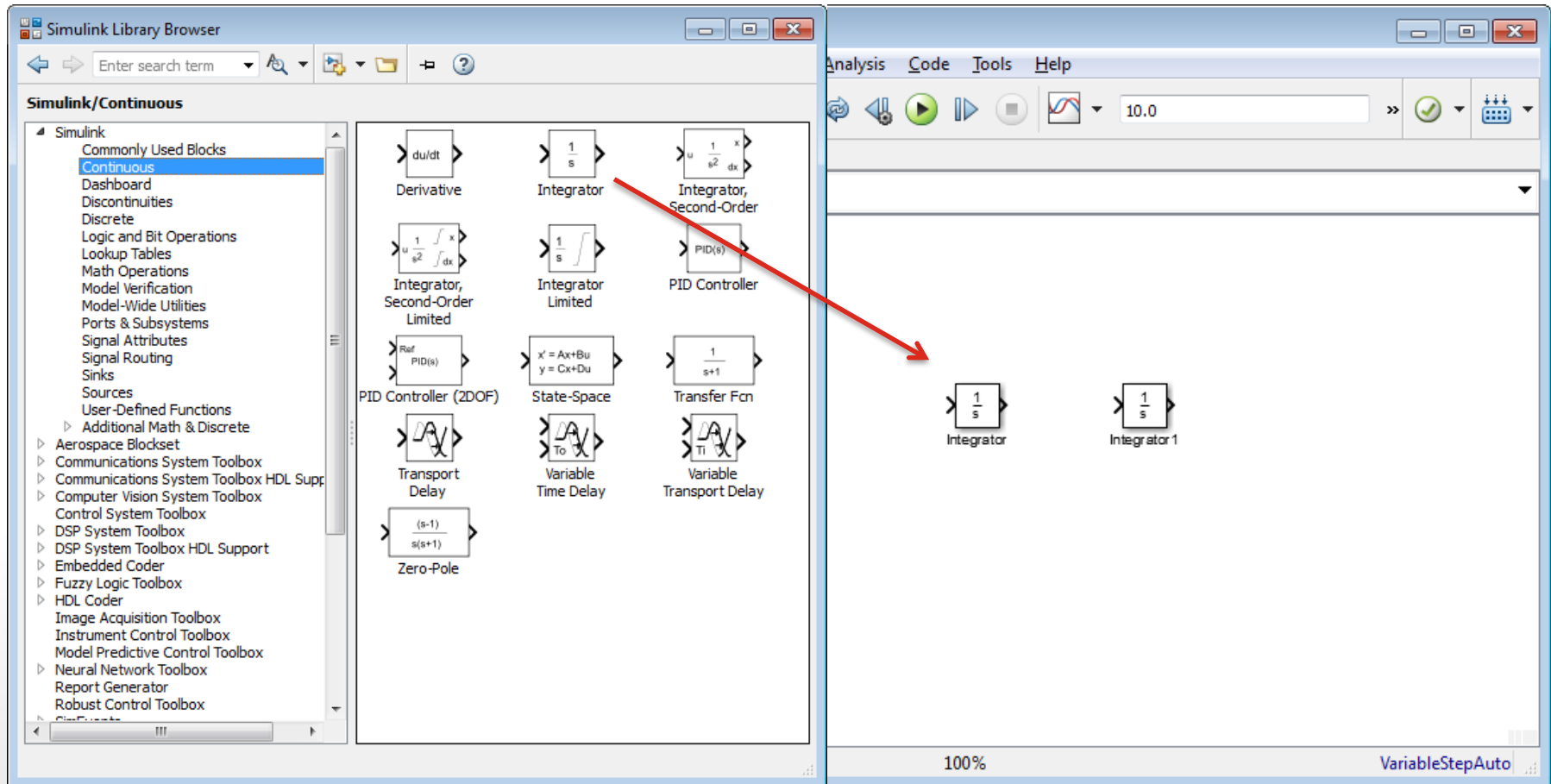


MATLAB / Simulink

- The first thing to do in Simulink is to build the signal chain, from the **highest derivative to lowest derivative**.
- In our case, it would be $\ddot{\theta}$ down to θ .
- How is this done?
- Imagine you have a signal $\ddot{\theta}$.
- If we **integrate** this signal, it would become $\dot{\theta}$.
- If we **integrate** this once more, it would become θ .

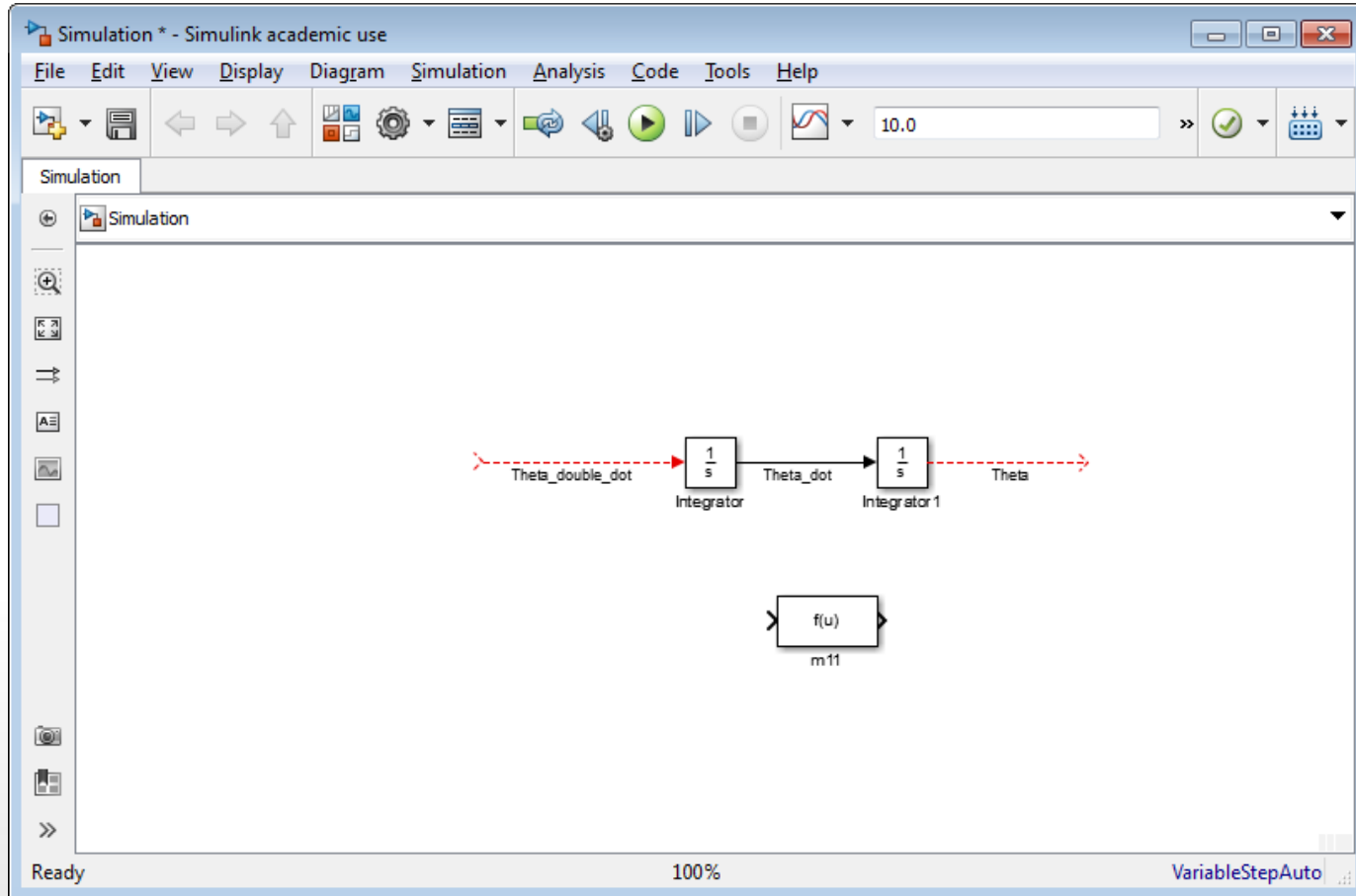
MATLAB / Simulink

- From Simulink Library, go to “Continuous” and then drag and drop “Integrator” twice into the Simulink window.



MATLAB / Simulink

- From Simulink Library, go to “**User-defined Functions**” and then drag-and-drop “**Fcn**” into Simulink Window.
- Rename “Fcn” to “m11”.

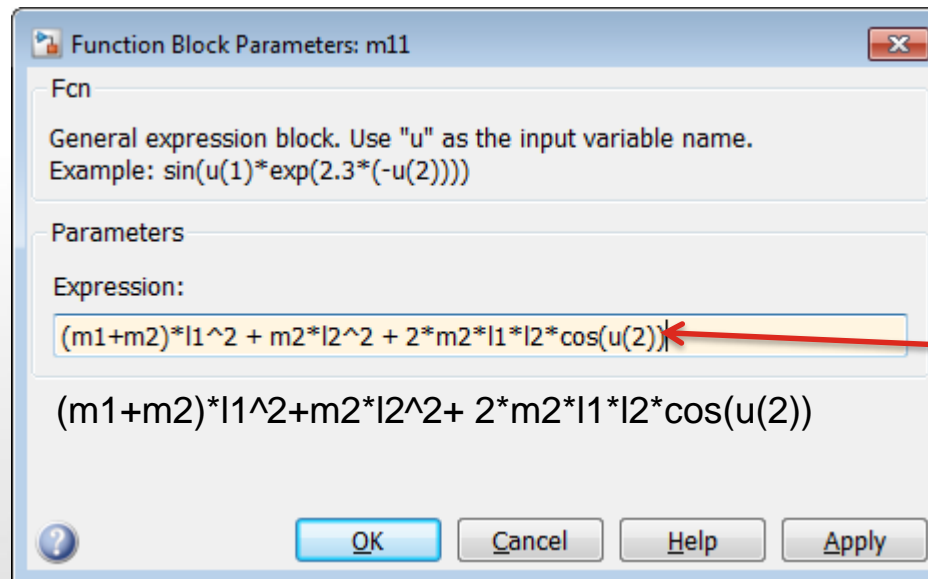


MATLAB / Simulink

- Double click the block “m11” and key in the expression for m11.

$$\underbrace{\begin{bmatrix} (m_1 + m_2)L_1^2 + m_2L_2^2 + 2m_2L_1L_2c_2 & m_2L_2^2 + m_2L_1L_2c_2 \\ m_2L_2^2 + m_2L_1L_2c_2 & m_2L_2^2 \end{bmatrix}}_{M(q)} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \underbrace{\begin{bmatrix} -m_2L_1L_2s_2\dot{\theta}_2^2 \\ m_2L_1L_2s_2\dot{\theta}_1^2 \end{bmatrix}}_{\text{Centrifugal}} + \underbrace{\begin{bmatrix} -2m_2L_1L_2s_1\dot{\theta}_1\dot{\theta}_2 \\ 0 \end{bmatrix}}_{\text{Coriolis}} + \underbrace{\begin{bmatrix} m_2gL_2c_{12} + (m_1 + m_2)gL_1c_1 \\ m_2gL_2c_{12} \end{bmatrix}}_{G(q)} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

$V(q, \dot{q})$



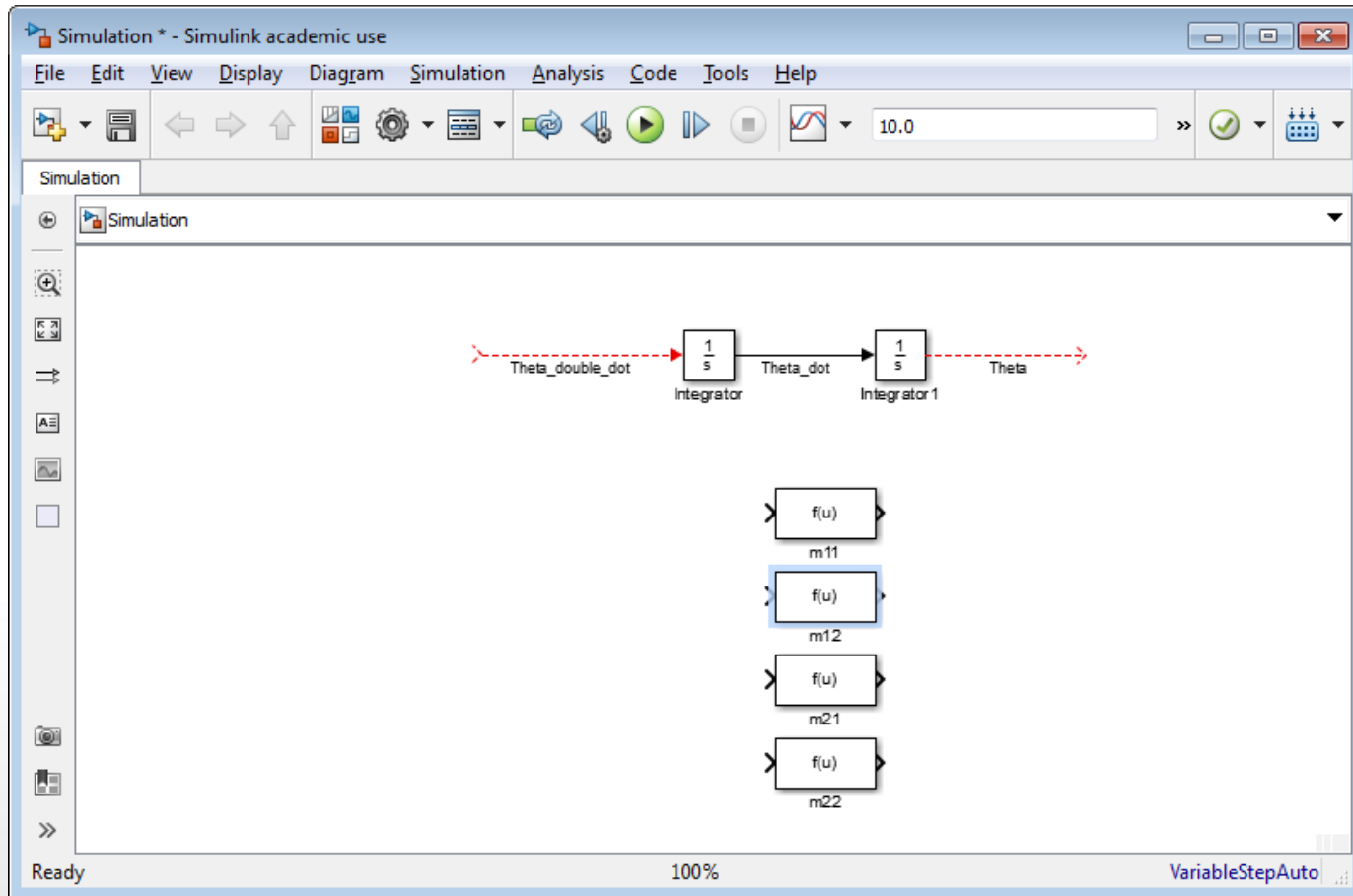
The signal is a vector of theta1 and theta2.

Theta1 = u(1)

Theta2 = u(2)

MATLAB / Simulink

- Repeat the last two steps for m12, m21 and m22.

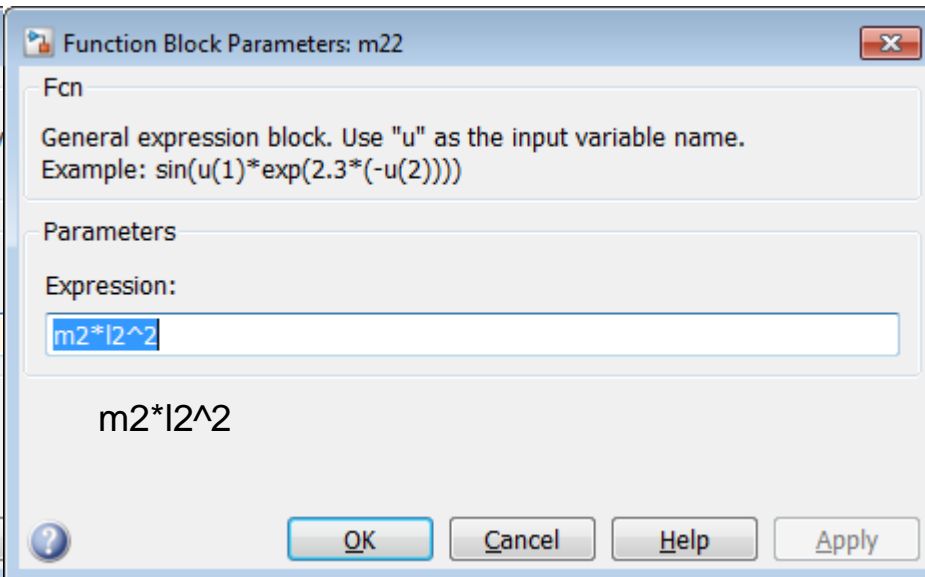
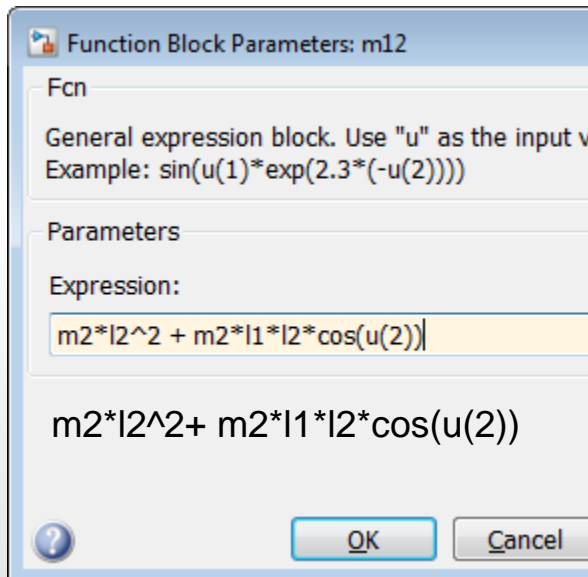


MATLAB / Simulink

- The expressions for m12, m21 (=m12), m22 should be as shown:

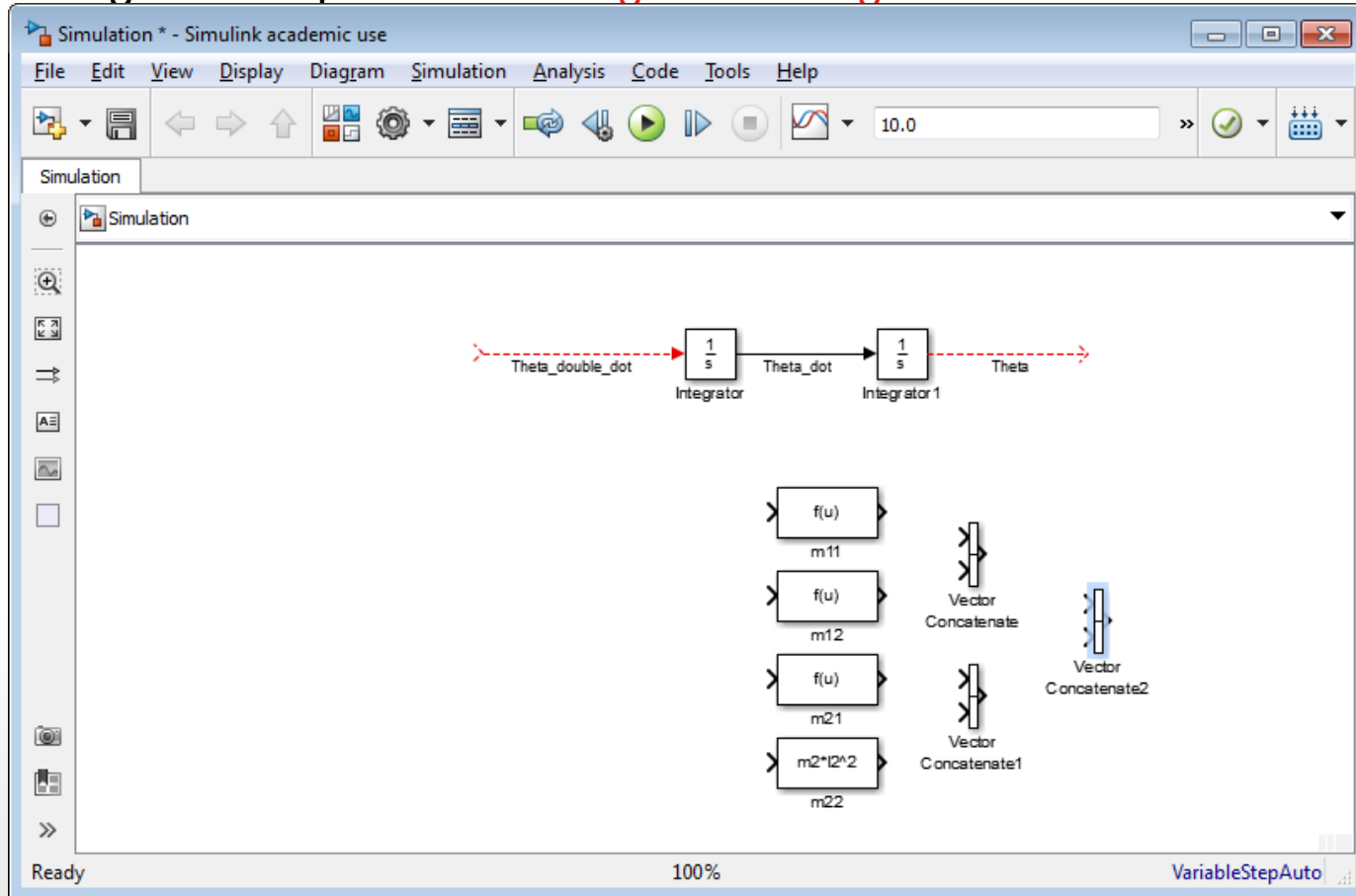
$$\underbrace{\begin{bmatrix} (m_1 + m_2)L_1^2 + m_2L_2^2 + 2m_2L_1L_2c_2 & m_2L_2^2 + m_2L_1L_2c_2 \\ m_2L_2^2 + m_2L_1L_2c_2 & m_2L_2^2 \end{bmatrix}}_{M(q)} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \underbrace{\begin{bmatrix} -m_2L_1L_2s_2\dot{\theta}_2^2 \\ m_2L_1L_2s_2\dot{\theta}_1^2 \end{bmatrix}}_{\text{Centrifugal}} + \underbrace{\begin{bmatrix} -2m_2L_1L_2s_1\dot{\theta}_1\dot{\theta}_2 \\ 0 \end{bmatrix}}_{\text{Coriolis}} + \underbrace{\begin{bmatrix} m_2gL_2c_{12} + (m_1 + m_2)gL_1c_1 \\ m_2gL_2c_{12} \end{bmatrix}}_{G(q)} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

$V(q, \dot{q})$



MATLAB / Simulink

- Having defined m11, m12, m21 and m22, our next step is to combine them to make the matrix $M(q)$.
- Drag and Drop the block “Signal Routing” → “Vector Concatenate” 3 times.



MATLAB / Simulink

- For the first two “Vector Concatenate” blocks, set the following parameters:
- (The concatenate dimension 2 means horizontal concatenation).

The screenshot displays the MATLAB/Simulink interface. The main workspace shows a Simulink model with a signal chain: $\text{Theta_double_dot} \rightarrow \text{Integrator} \rightarrow \text{Theta_dot} \rightarrow \text{Integrator 1} \rightarrow \text{Theta}$. Below this, there are four blocks labeled $f(u)$ with identifiers $m11$, $m12$, $m21$, and $m22$. Two 'Vector Concatenate' blocks are shown, with the top one circled in red. The 'Function Block Parameters: Vector Concatenate' dialog box is open on the right, showing the 'Concatenate' tab. The 'Number of inputs' is set to 2, the 'Mode' is 'Multidimensional array', and the 'Concatenate dimension' is set to 2. The dialog box also contains explanatory text about vector and multidimensional modes.

Function Block Parameters: Vector Concatenate

Concatenate

Concatenate input signals of the same data type to create a contiguous output signal. Select vector or multidimensional array mode.

In vector mode, all input signals must be either vectors or one-row [1xM] matrices or one-column [Mx1] matrices or a combination of vectors and either one-row matrices or one-column matrices. The output is a vector if all inputs are vectors. The output is a one-row or one-column matrix if any of the inputs are one-row or one-column matrices, respectively.

In multidimensional mode, use 'Concatenate dimension' to specify the output dimension along which to concatenate the input arrays. For example, to concatenate the input arrays vertically or horizontally, specify 1 or 2, respectively, as the concatenate dimensions.

Parameters

Number of inputs: 2

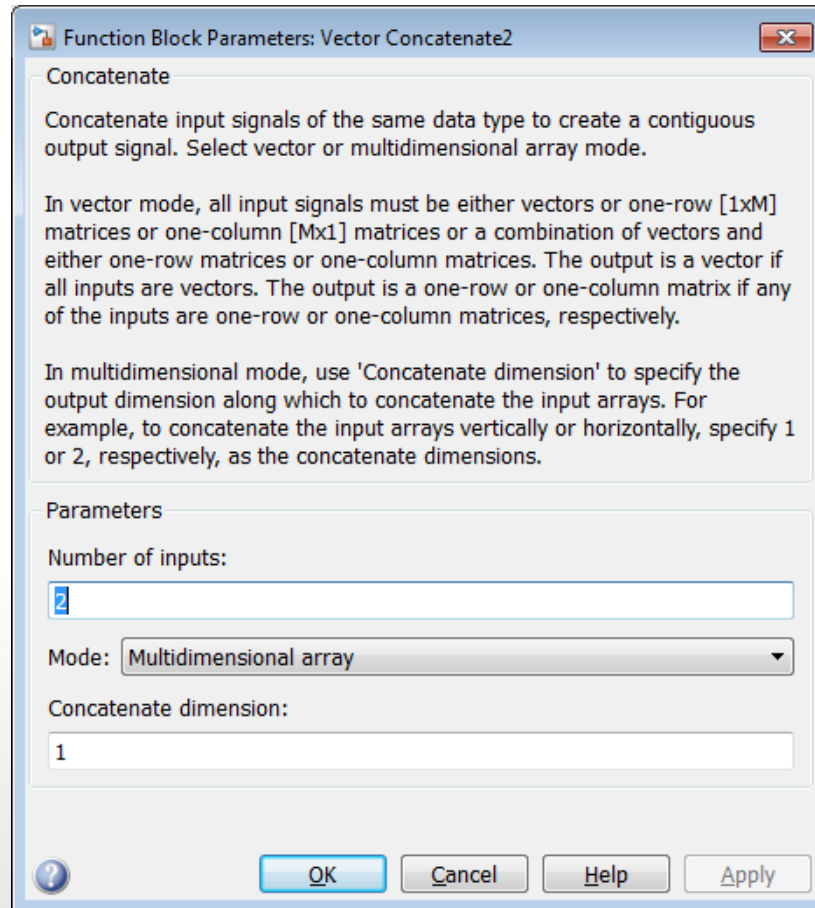
Mode: Multidimensional array

Concatenate dimension: 2

Buttons: OK, Cancel, Help, Apply

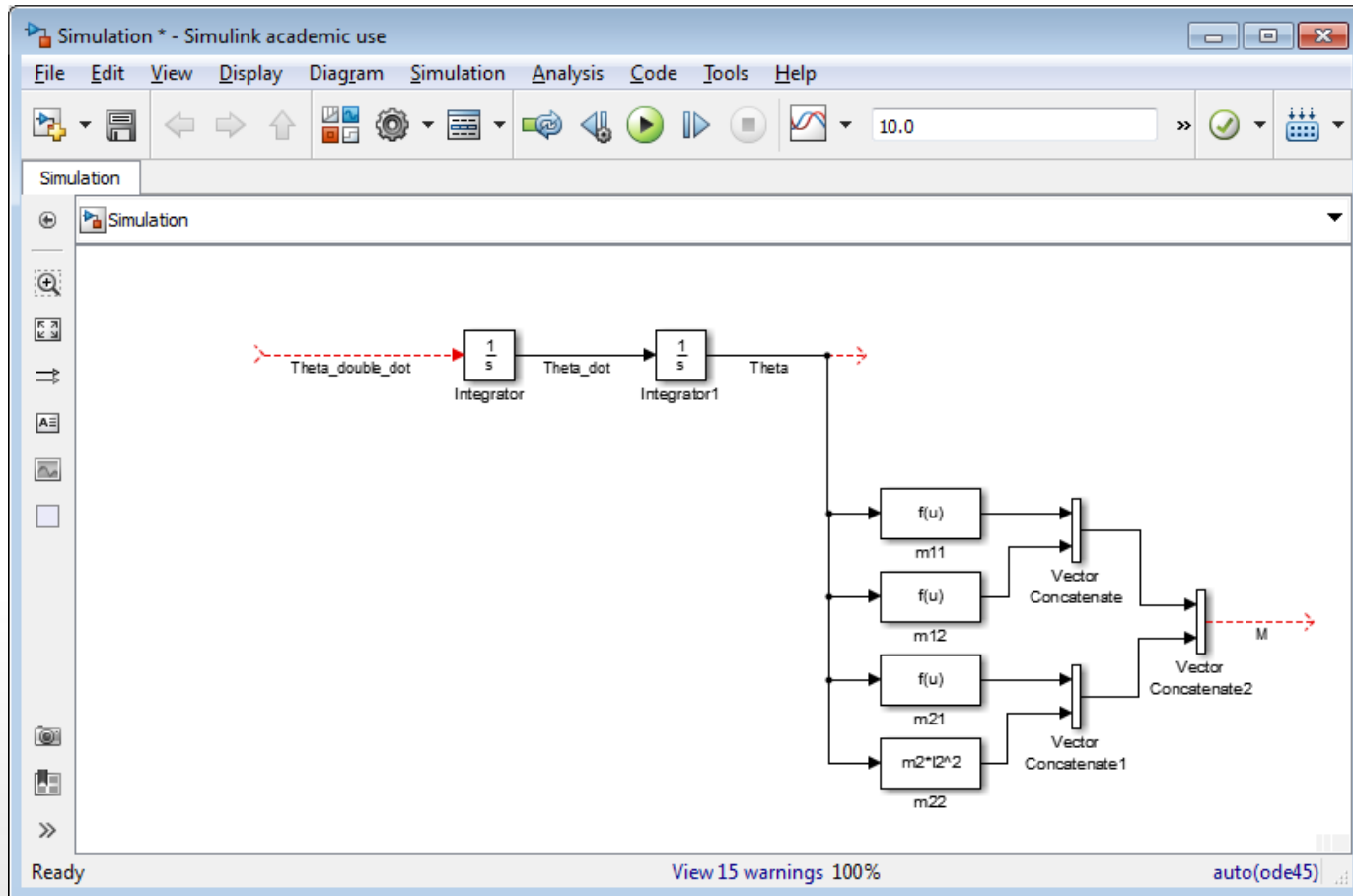
MATLAB / Simulink

- For the third “Vector Concatenate” blocks, set the following parameters:
- (The concatenate dimension 1 means vertical concatenation).



MATLAB / Simulink

- Now combine all the blocks as shown:



MATLAB / Simulink

- We are done with $M(q)$ now, and will move on to $V(q, \dot{q})$ and $G(q)$.

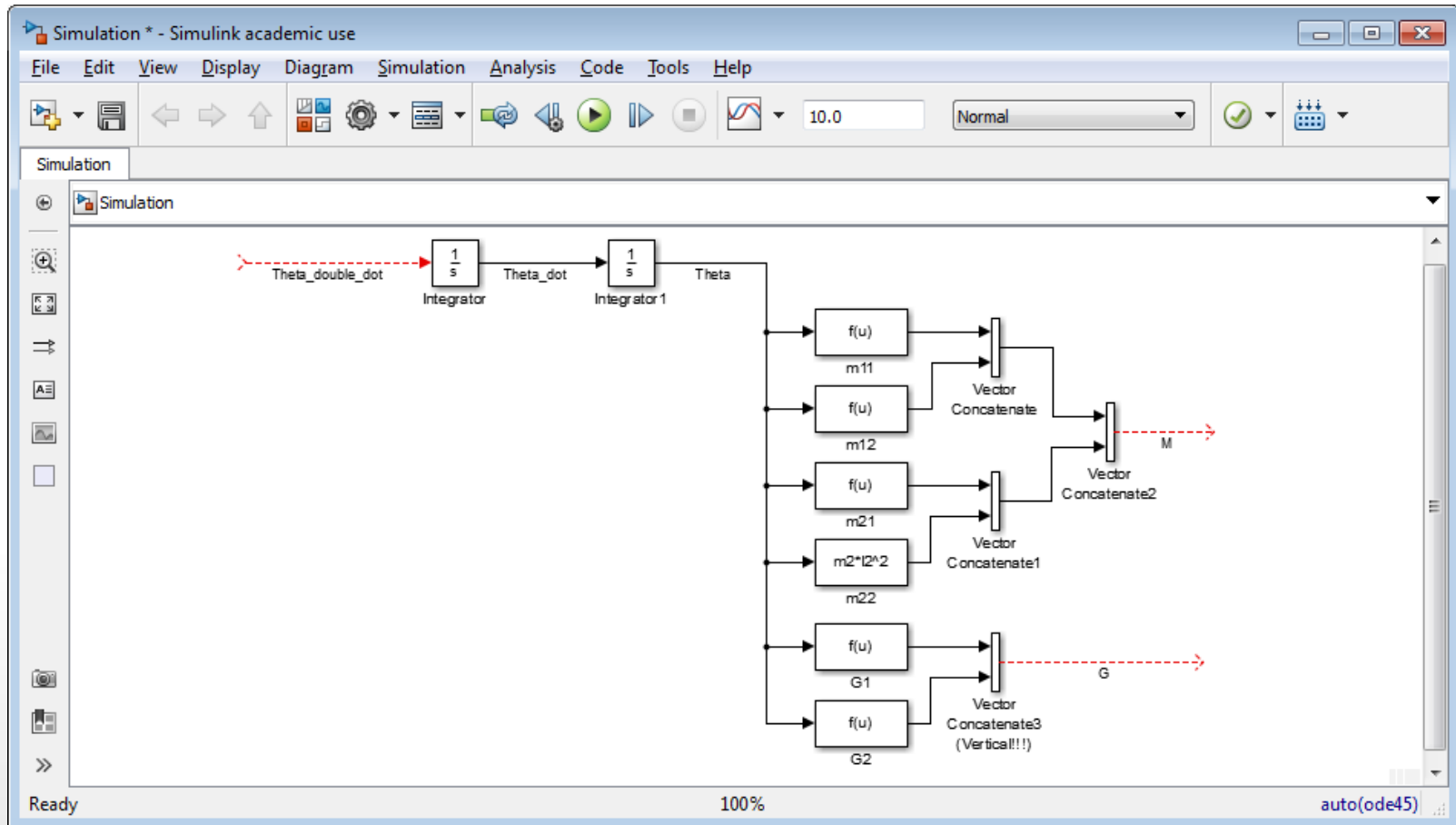
$$\underbrace{\begin{bmatrix} (m_1 + m_2)L_1^2 + m_2L_2^2 + 2m_2L_1L_2c_2 & m_2L_2^2 + m_2L_1L_2c_2 \\ m_2L_2^2 + m_2L_1L_2c_2 & m_2L_2^2 \end{bmatrix}}_{M(q)} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \underbrace{\begin{bmatrix} -m_2L_1L_2s_2\dot{\theta}_2^2 \\ m_2L_1L_2s_2\dot{\theta}_1^2 \end{bmatrix}}_{\text{Centrifugal}} + \underbrace{\begin{bmatrix} -2m_2L_1L_2s_1\dot{\theta}_1\dot{\theta}_2 \\ 0 \end{bmatrix}}_{\text{Coriolis}} + \underbrace{\begin{bmatrix} m_2gL_2c_{12} + (m_1 + m_2)gL_1c_1 \\ m_2gL_2c_{12} \end{bmatrix}}_{G(q)} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

$V(q, \dot{q})$

- Try to build $G(q)$ on your own. The steps are similar to $M(q)$, except that this is just a vector instead of matrix.

MATLAB / Simulink

- You should be getting this after adding in $G(q)$:



MATLAB / Simulink

- For $V(q, \dot{q})$, we need to combine theta and theta_dot into one longer vector.

$$\underbrace{\begin{bmatrix} (m_1 + m_2)L_1^2 + m_2L_2^2 + 2m_2L_1L_2c_2 & m_2L_2^2 + m_2L_1L_2c_2 \\ m_2L_2^2 + m_2L_1L_2c_2 & m_2L_2^2 \end{bmatrix}}_{M(q)} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \underbrace{\begin{bmatrix} -m_2L_1L_2s_2\dot{\theta}_2^2 \\ m_2L_1L_2s_2\dot{\theta}_1^2 \end{bmatrix}}_{\text{Centrifugal}} + \underbrace{\begin{bmatrix} -2m_2L_1L_2s_1\dot{\theta}_1\dot{\theta}_2 \\ 0 \end{bmatrix}}_{\text{Coriolis}} + \underbrace{\begin{bmatrix} m_2gL_2c_{12} + (m_1 + m_2)gL_1c_1 \\ m_2gL_2c_{12} \end{bmatrix}}_{G(q)} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

$V(q, \dot{q})$

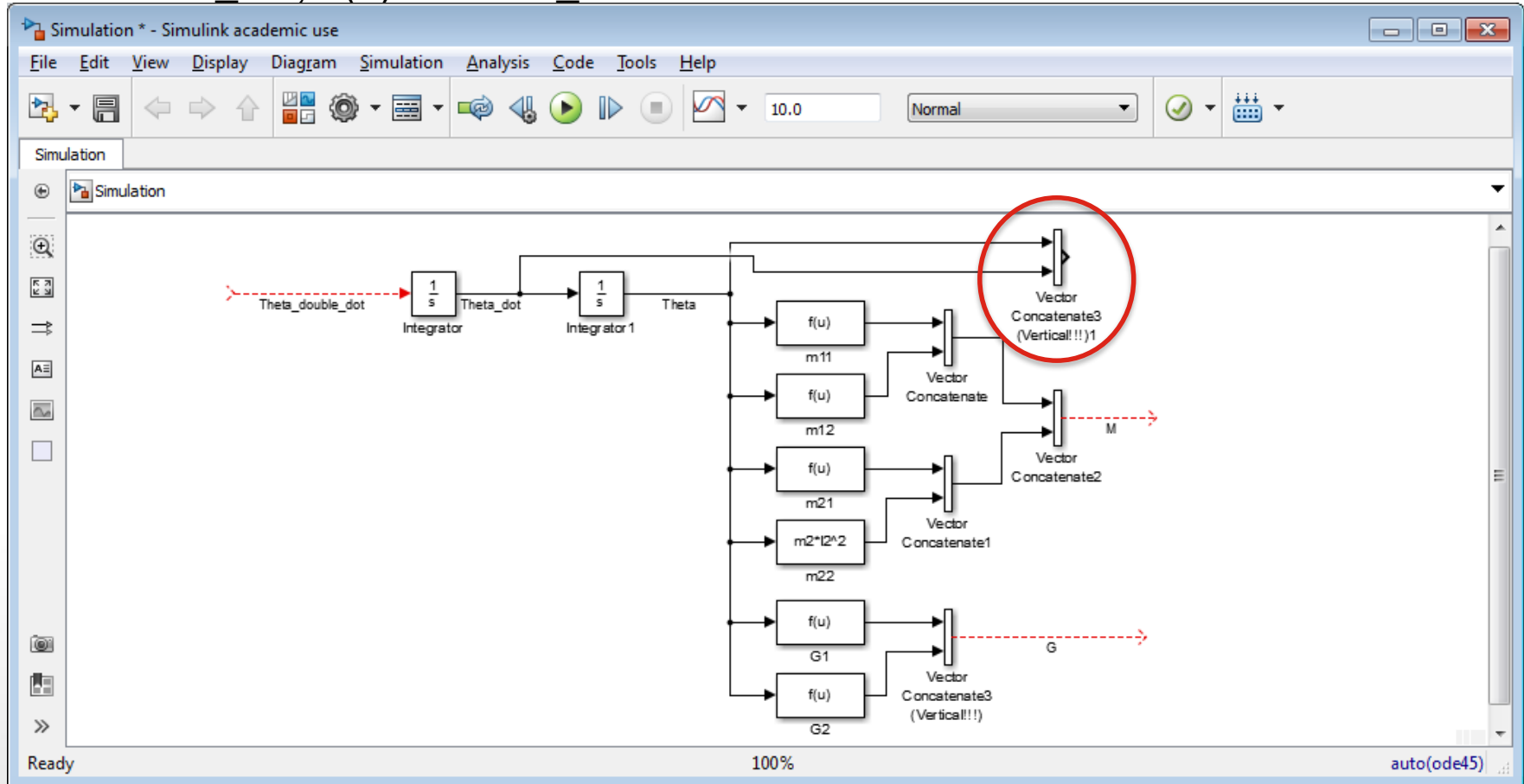
- You can also use “Vector Concatenation” in vertical direction, to create the vector $[\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2]^T$.

$$G1 = m2*g*l2*cos(u(1)+u(2))+(m1+m2)*g*l1*cos(u(1))$$

$$G2 = m2*g*l2*cos(u(1)+u(2))$$

MATLAB / Simulink

- The combined vector is built as shown:
- Then, the output of the block will be $u(1) = \text{theta1}$, $u(2) = \text{theta2}$, $u(3) = \text{theta1_dot}$, $u(4) = \text{theta2_dot}$.



MATLAB / Simulink

- With this, you should be able to give the expression for $V(q, \dot{q})$:

$$\underbrace{\begin{bmatrix} (m_1 + m_2)L_1^2 + m_2L_2^2 + 2m_2L_1L_2c_2 & m_2L_2^2 + m_2L_1L_2c_2 \\ m_2L_2^2 + m_2L_1L_2c_2 & m_2L_2^2 \end{bmatrix}}_{M(q)} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + \underbrace{\begin{bmatrix} -m_2L_1L_2s_2\dot{\theta}_2^2 \\ m_2L_1L_2s_2\dot{\theta}_1^2 \end{bmatrix}}_{\text{Centrifugal}} + \underbrace{\begin{bmatrix} -2m_2L_1L_2s_1\dot{\theta}_1\dot{\theta}_2 \\ 0 \end{bmatrix}}_{\text{Coriolis}} + \underbrace{\begin{bmatrix} m_2gL_2c_{12} + (m_1 + m_2)gL_1c_1 \\ m_2gL_2c_{12} \end{bmatrix}}_{G(q)} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}$$

$V(q, \dot{q})$

Function Block Parameters: V1

Fcn
General expression block. Use "u" as the input variable name.
Example: sin(u(1)*exp(2.3*(-u(2))))

Parameters

Expression:

`-m2*I1*I2*sin(u(2))*u(4)^2-2*m2*I1*I2*sin(u(1))*u(3)*u(4)`

V1 = -m2*I1*I2*sin(u(2))*u(4)^2-2*m2*I1*I2*sin(u(1))*u(3)*u(4)

OK Cancel Help Apply

Function Block Parameters: V2

Fcn
General expression block. Use "u" as the input variable name.
Example: sin(u(1)*exp(2.3*(-u(2))))

Parameters

Expression:

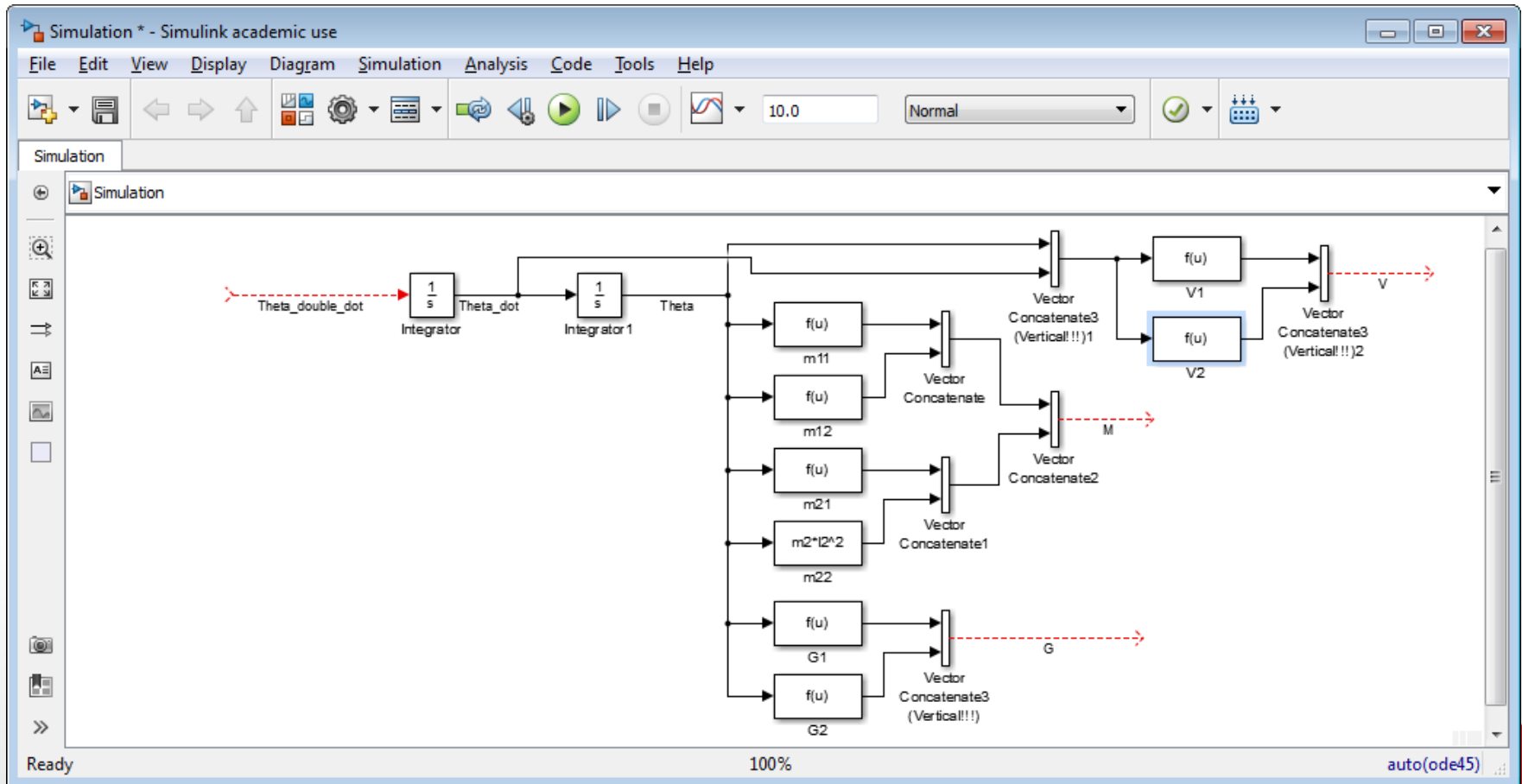
`m2*I1*I2*sin(u(2))*u(3)^2`

V2 = m2*I1*I2*sin(u(2))*u(3)^2

OK Cancel Help Apply

MATLAB / Simulink

- The block diagram now looks like this:



MATLAB / Simulink

- We have built M, V and G by now.
- Next, we need to add in the torques, and see how they affect the acceleration.

- From our dynamic equation: $M(q)\ddot{q} + V(q, \dot{q}) + G(q) = \tau$

- We can write:

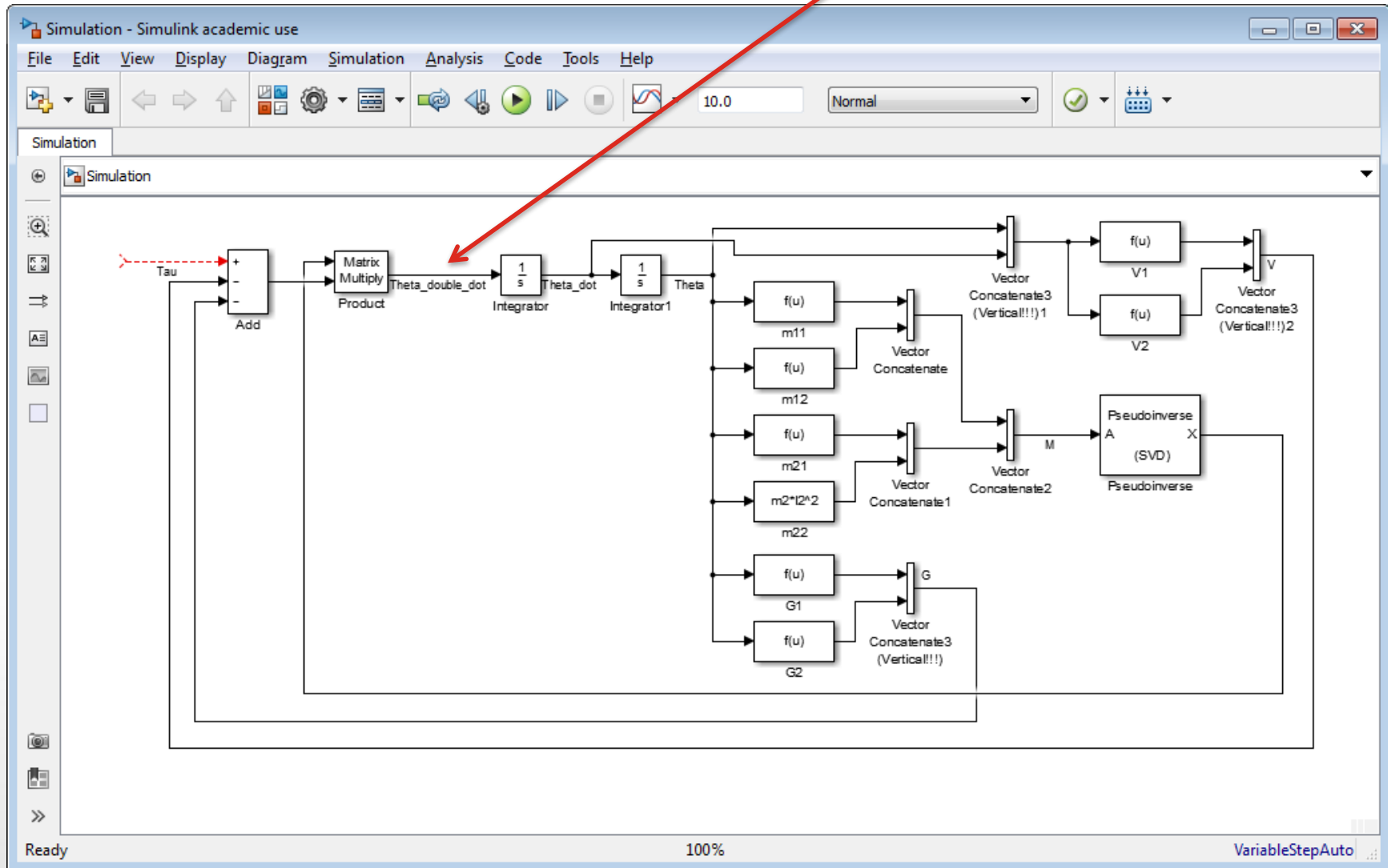
$$\ddot{q} = M(q)^{-1}[\tau - V(q, \dot{q}) - G(q)]$$

- Thus we need to invert the M matrix, and multiply this with (Tau – V – G).
- All these are done in a straightforward manner in Simulink.
- Just get “Add”, “Product” from “Math Operation” and connect everything as shown in next page. ** The “Product” parameters should be changed to “Matrix”.
- We also need “Pseudoinverse” block. Use search function to look for it.

MATLAB / Simulink

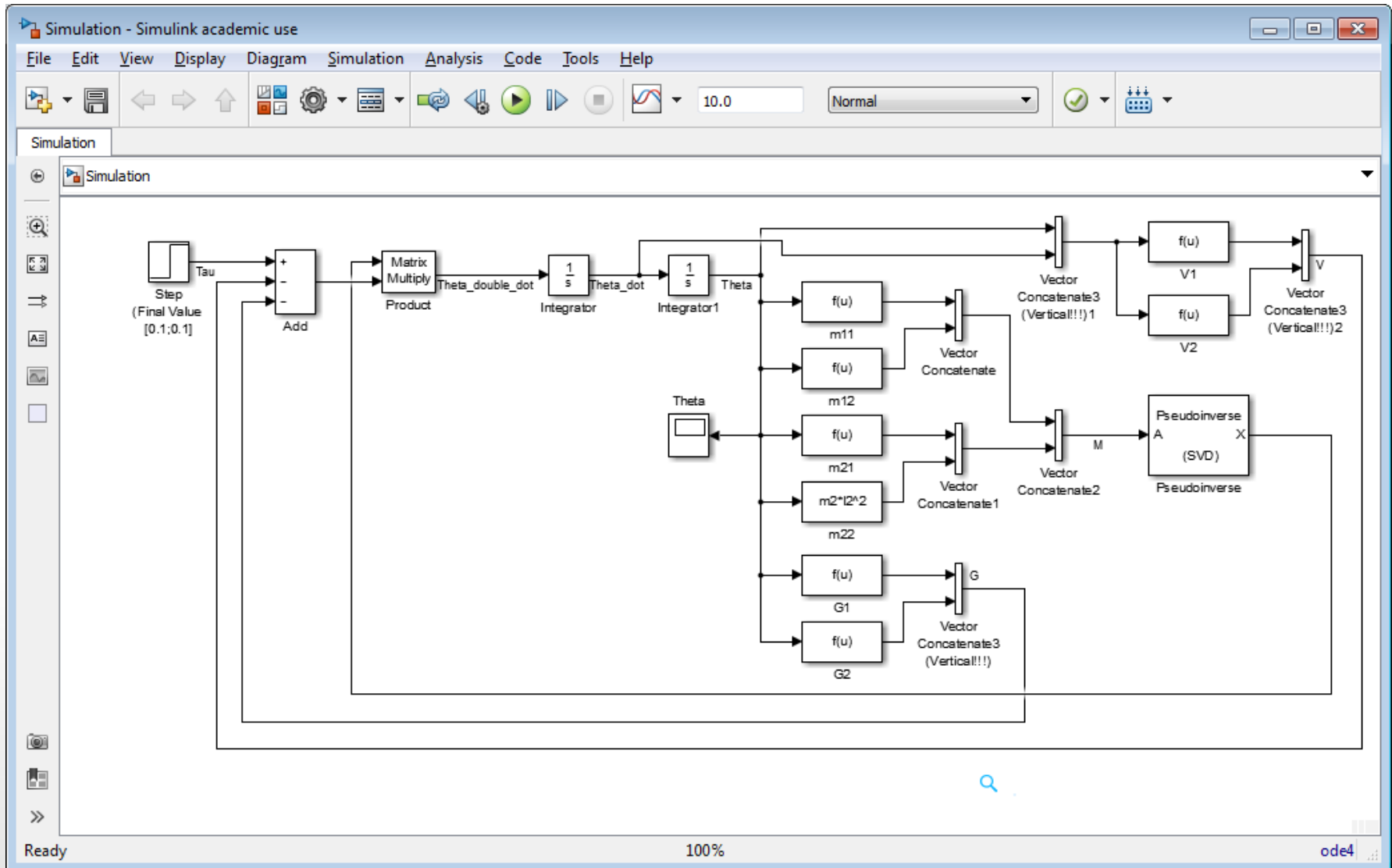
- The Blocks should be connected as follows:

$$\ddot{q} = M(q)^{-1}[\tau - V(q, \dot{q}) - G(q)]$$



MATLAB / Simulink

- The final steps are to add “Source” for torques and “Scope” to view Theta.

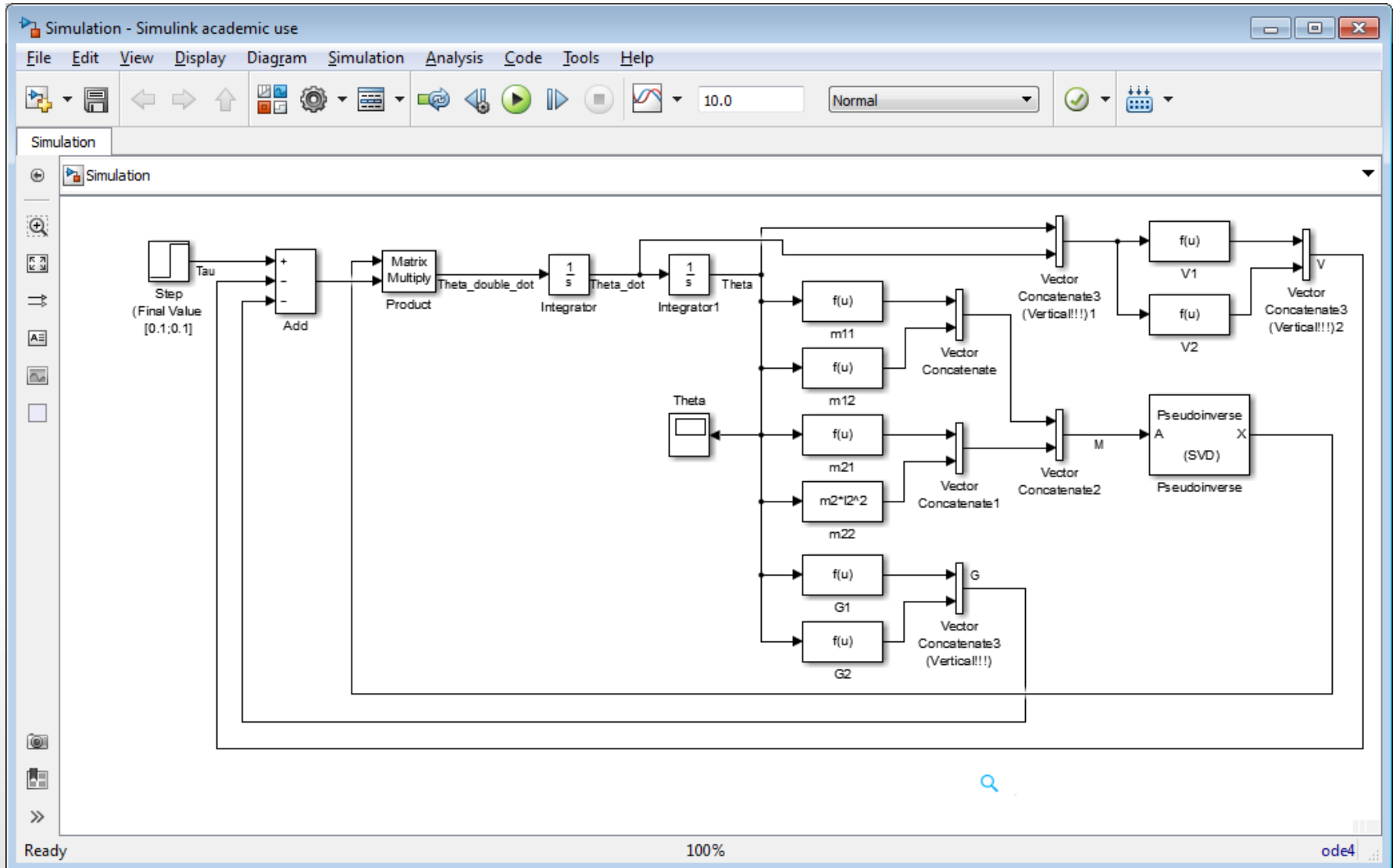


-
- The screenshot displays the Simulink academic use interface. The top menu bar includes File, Edit, View, Display, Diagram, Simulation, Analysis, Code, Tools, and Help. The toolbar contains various icons for simulation and editing. The main workspace shows a Simulink model titled "Simulation". The model is a control system for a robotic arm, featuring a step input for τ , an add block, a matrix multiply block, two integrators for $\ddot{\theta}$ and $\dot{\theta}$, and a feedback loop for θ . The forward path includes blocks for $f(u)$ with gains m_{11} , m_{12} , m_{21} , m_{22} , G_1 , and G_2 , concatenated into vectors, and a pseudoinverse block A (SVD). The status bar at the bottom shows "Ready", "100%", and "ode4".

MATLAB / Simulink

Only if there is an error in Vector Concatenate due to MATLAB Version

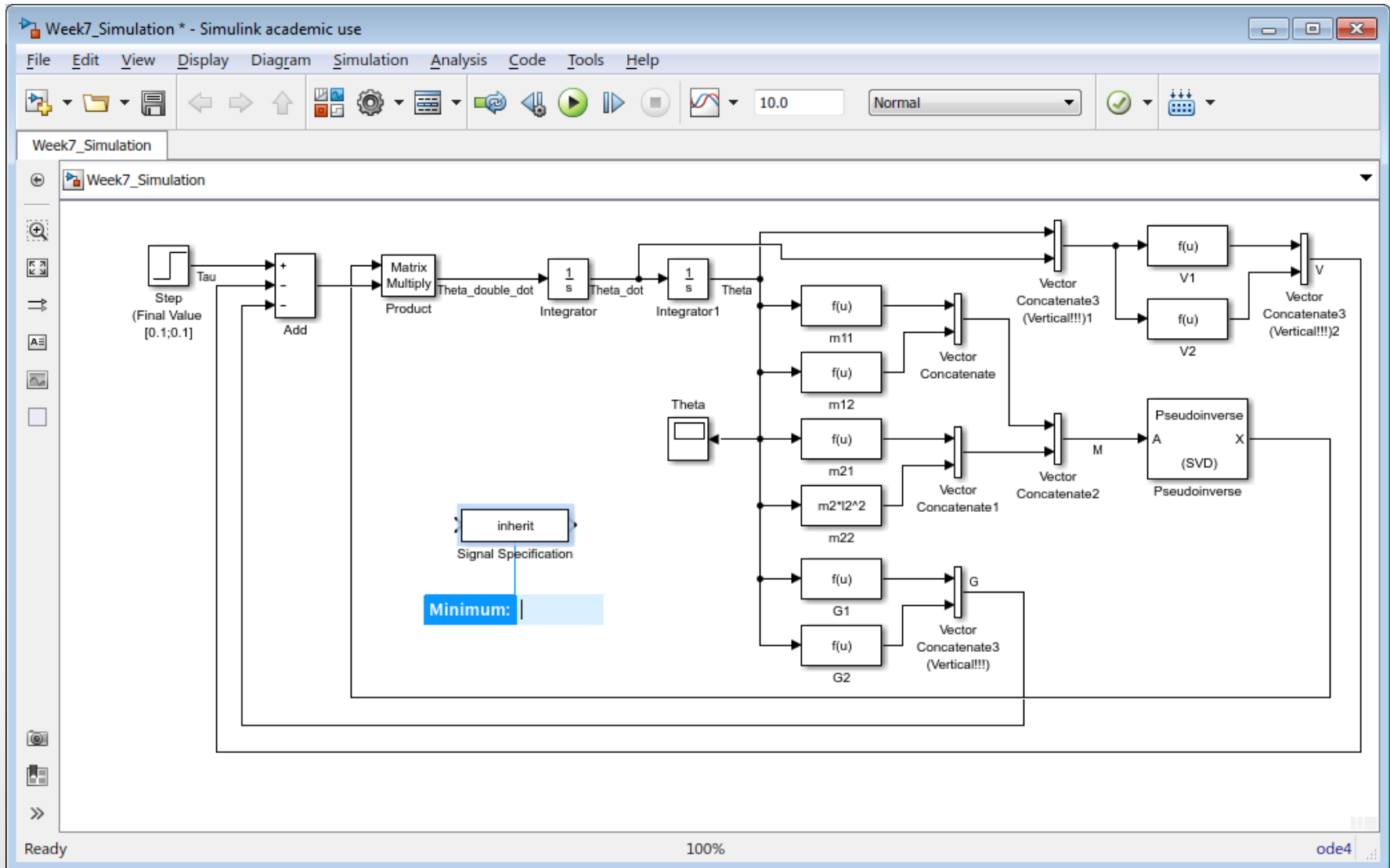
- But for some software/MATLAB versions, we need to add a few more items.



MATLAB / Simulink

Only if there is an error in Vector Concatenate due to MATLAB Version

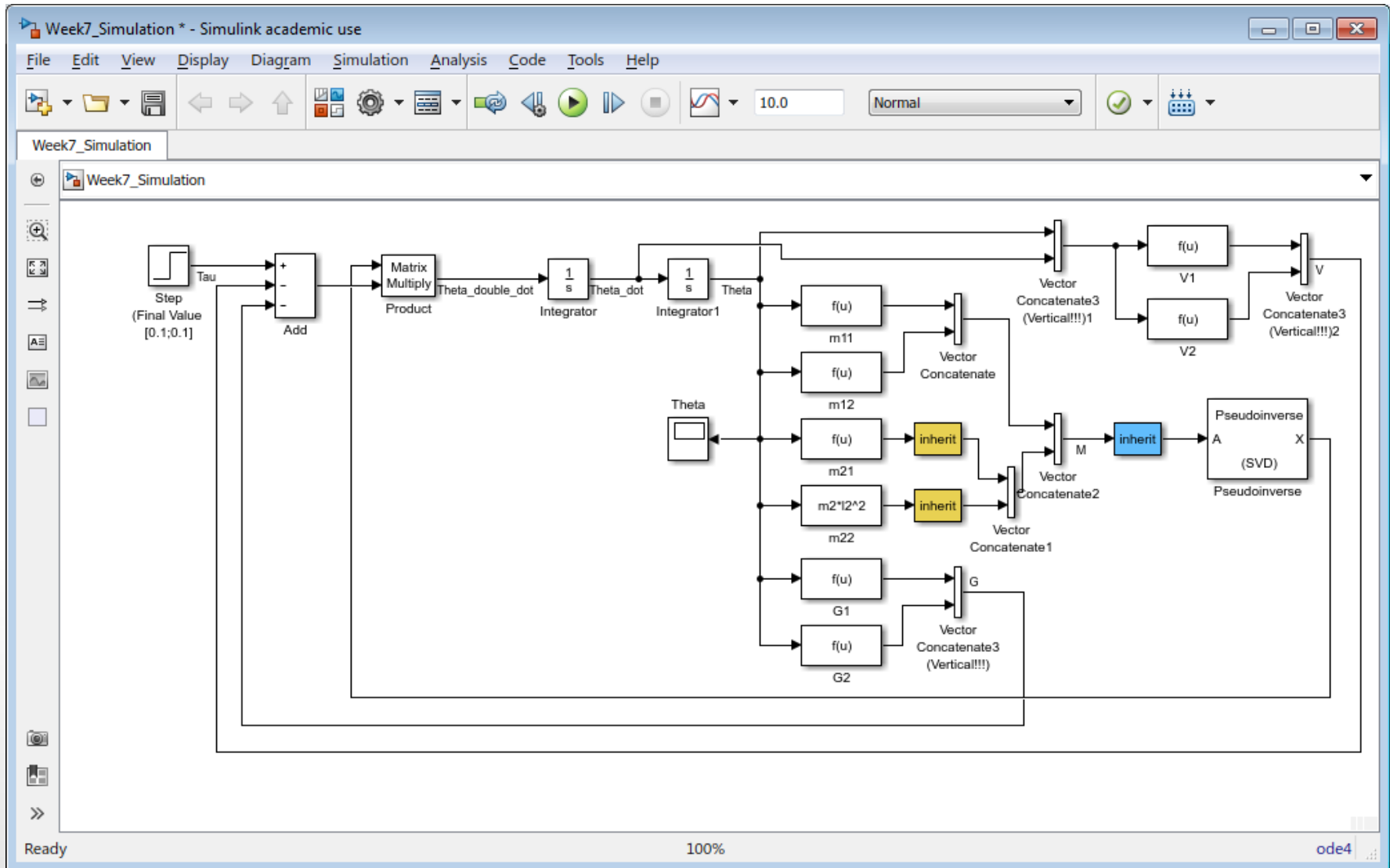
- Bring in “Signal Specification” block under “Signal Attributes”.



MATLAB / Simulink

Only if there is an error in Vector Concatenate due to MATLAB Version

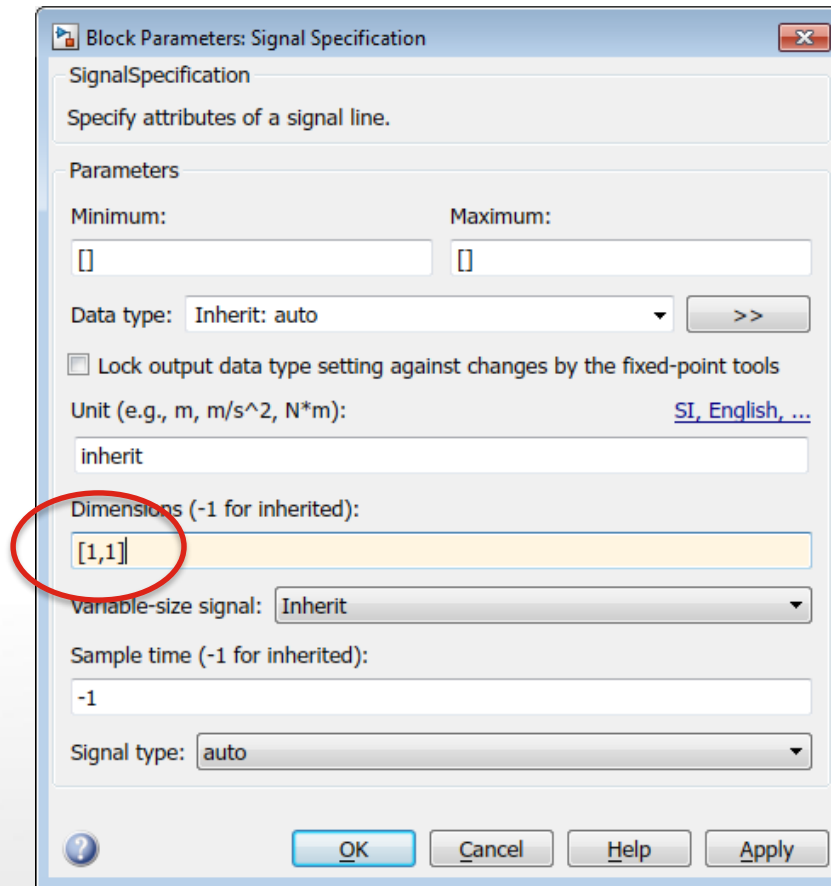
- Add this block at three locations as shown:



MATLAB / Simulink

Only if there is an error in Vector Concatenate due to MATLAB Version

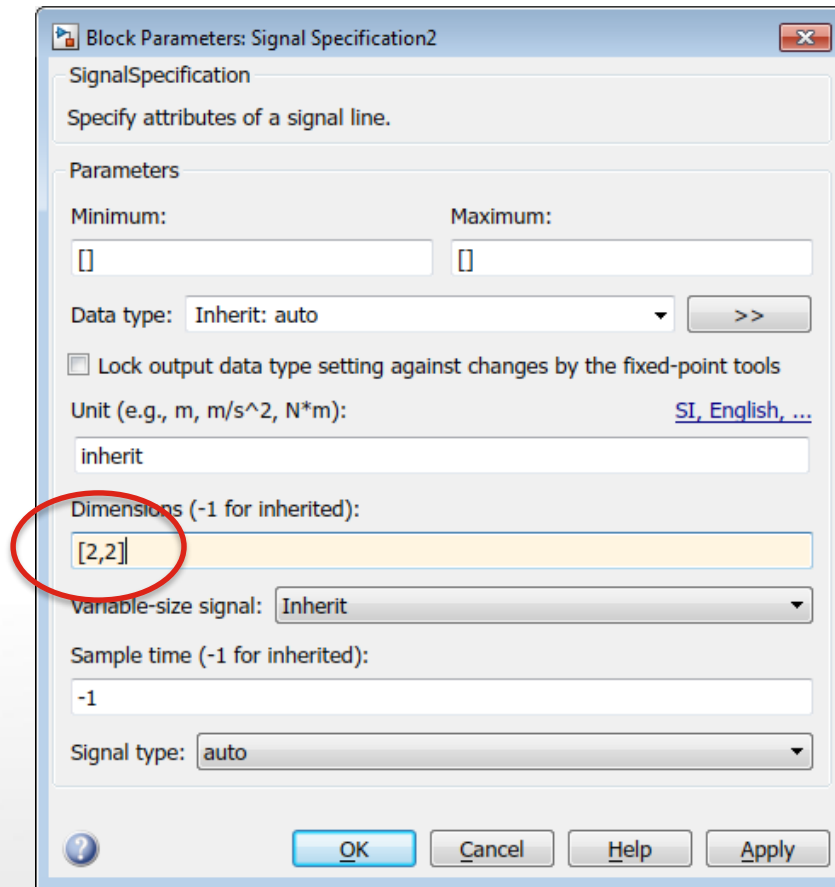
- Double click on the yellow blocks, and set the following parameter:



MATLAB / Simulink

Only if there is an error in Vector Concatenate due to MATLAB Version

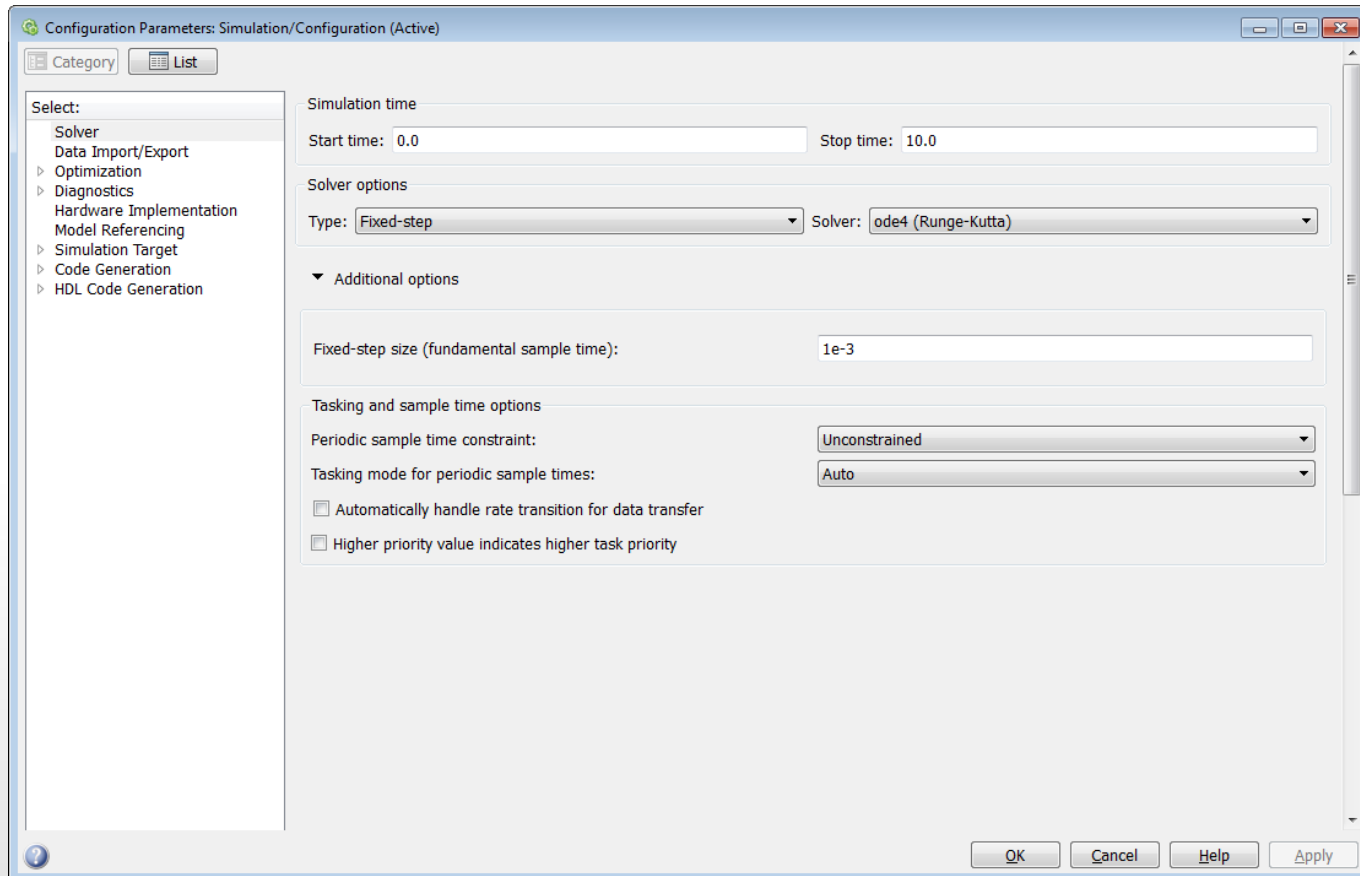
- Double click on the blue block, and set the following parameter:



MATLAB / Simulink

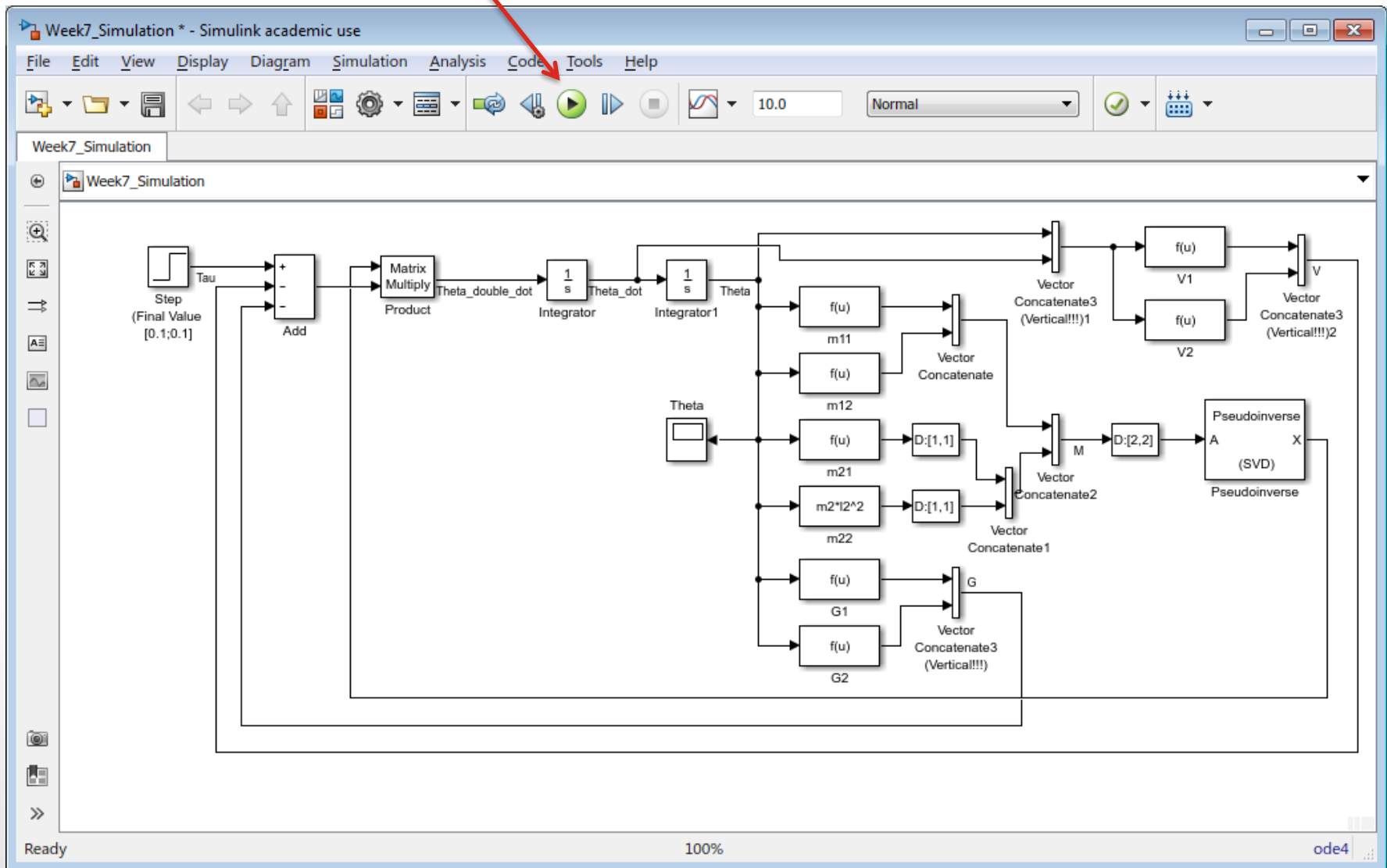
Continue from here

- Before we run the simulation, we would like to configure the simulation parameters.
- Click “**Simulation**” on top of Simulink window → **Model configuration parameters** → set the parameters as follows:



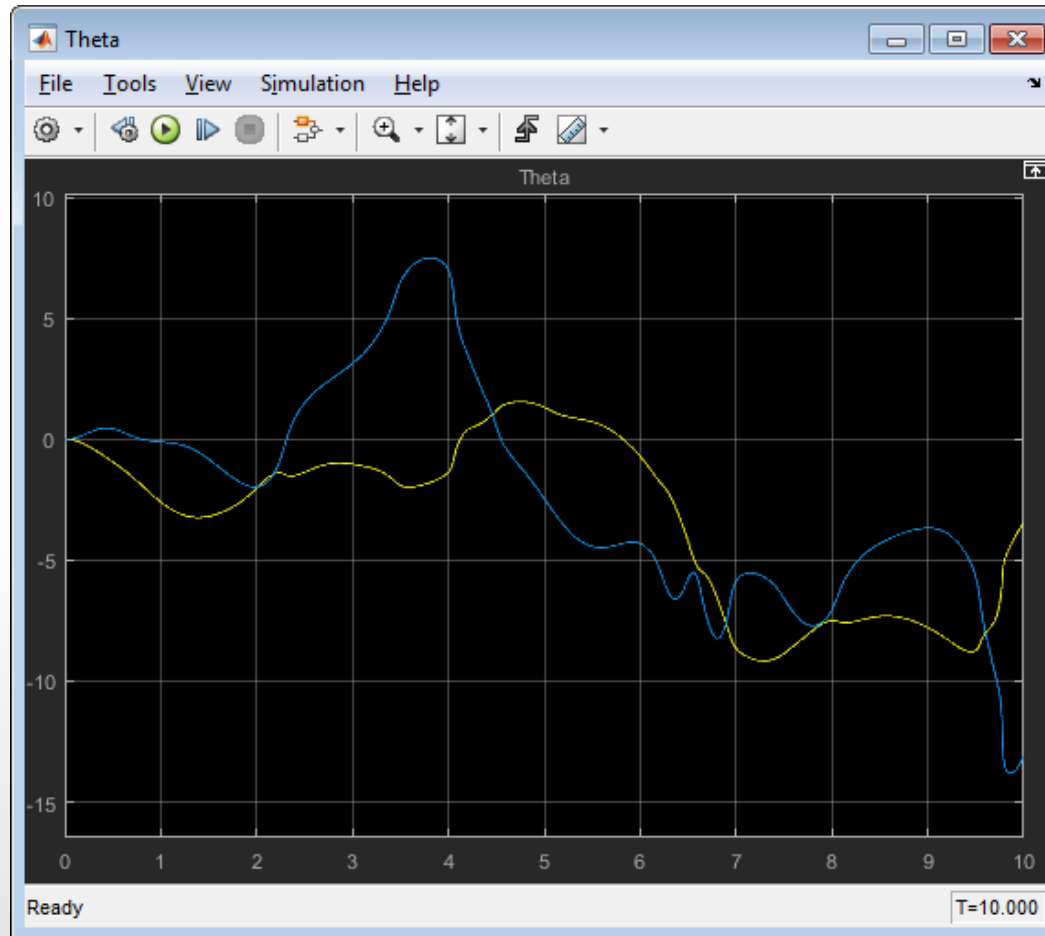
MATLAB / Simulink

- Now click the “play” button.



MATLAB / Simulink

- When simulation ends, double click “**Theta**” scope and you can see the Theta variations.
- This is the reaction of the robot, when given constant torques of 0.1Nm at each joint.

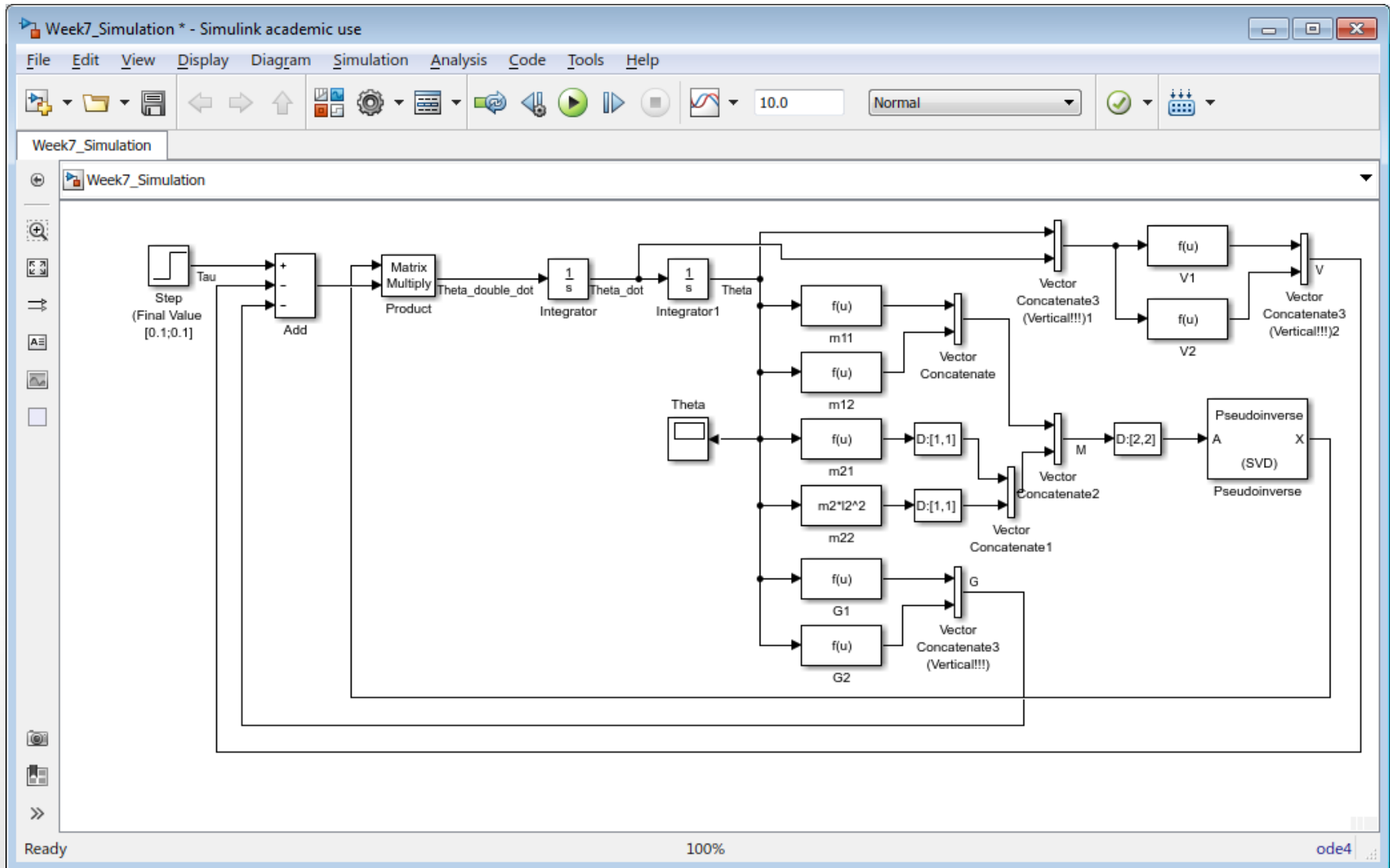


Yellow =
Theta1

Blue =
Theta2

MATLAB / Simulink

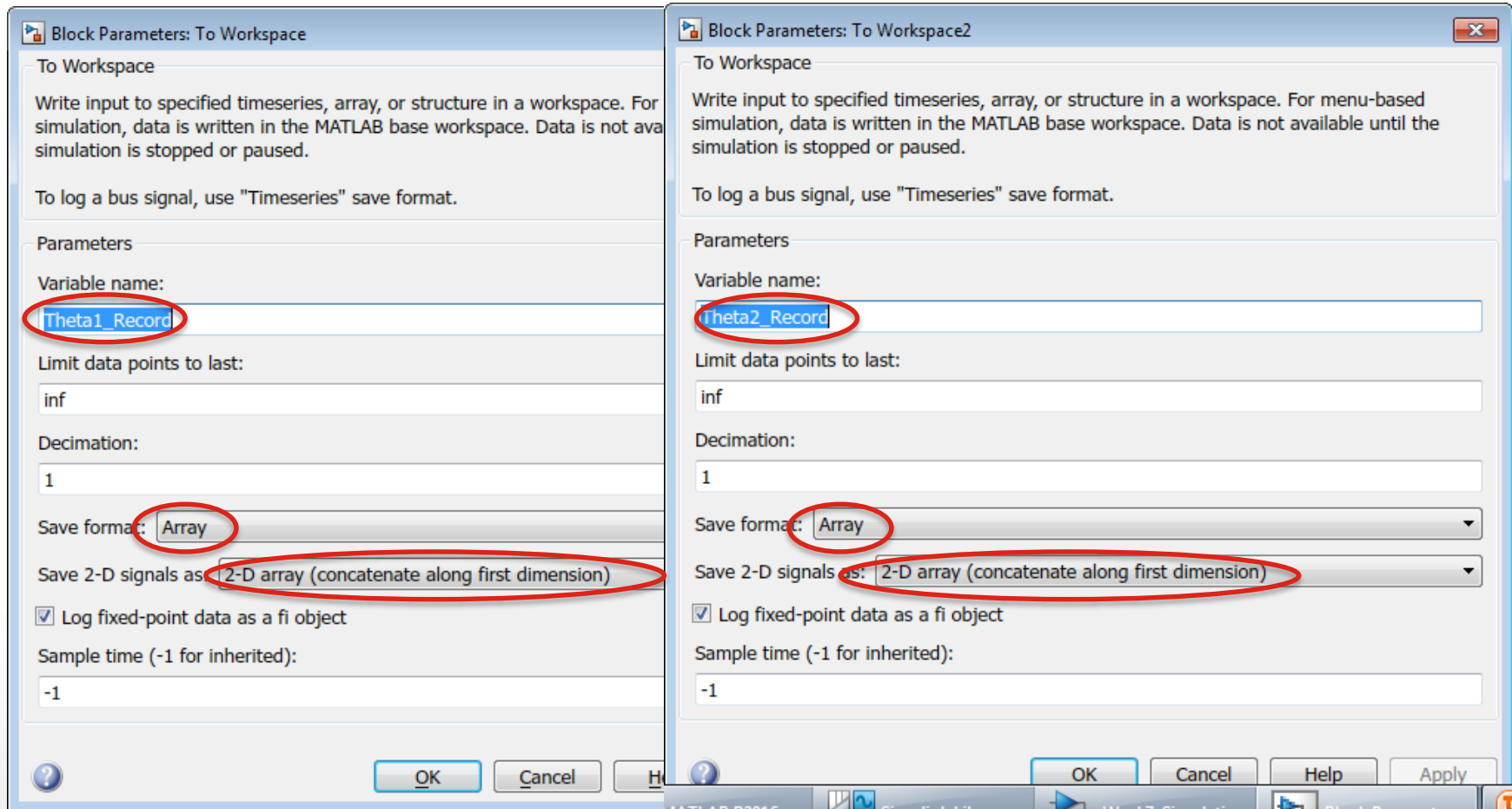
- In order to better visualize the robot motion, we can **run animation**.



-

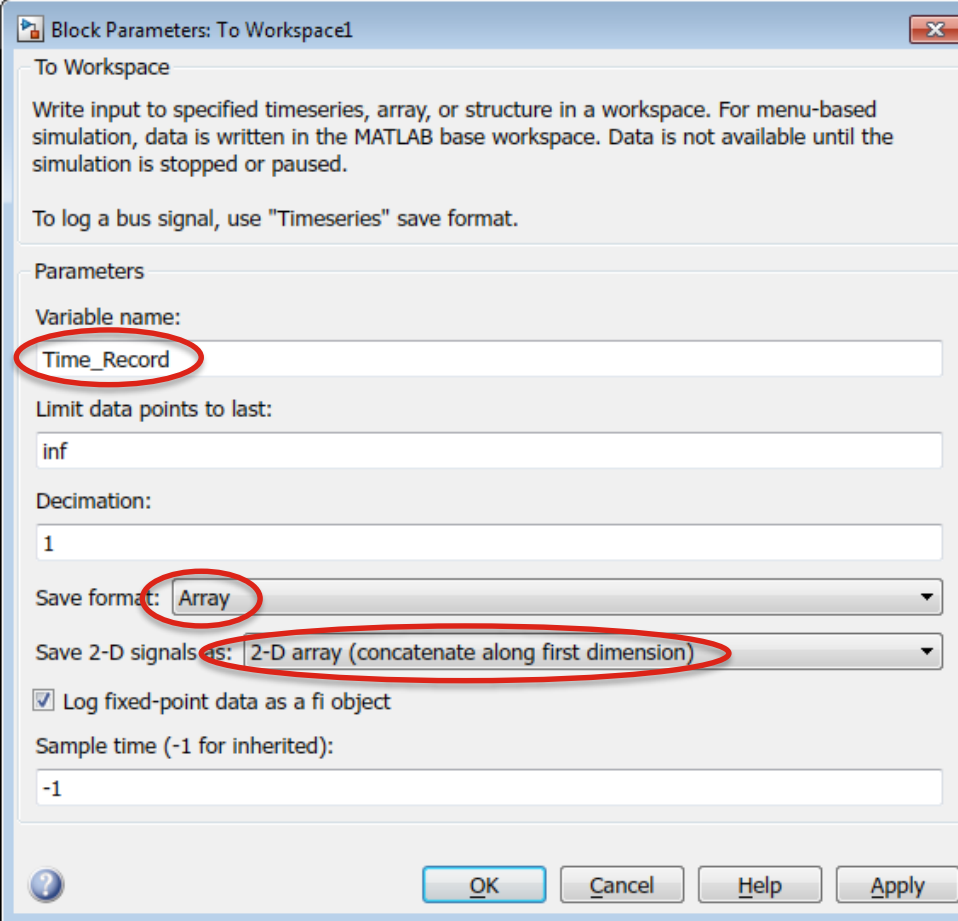
MATLAB / Simulink

- Configure the yellow blocks as follows:



MATLAB / Simulink

- Configure the blue block as follows:



The image shows the 'Block Parameters: To Workspace1' dialog box in MATLAB/Simulink. The dialog is titled 'Block Parameters: To Workspace1' and has a close button (X) in the top right corner. It contains several sections: 'To Workspace' with a description of data storage, 'Parameters' with input fields for 'Variable name' (set to 'Time_Record'), 'Limit data points to last' (set to 'inf'), 'Decimation' (set to '1'), 'Save format' (set to 'Array'), 'Save 2-D signals as' (set to '2-D array (concatenate along first dimension)'), a checked box for 'Log fixed-point data as a fi object', and 'Sample time (-1 for inherited)' (set to '-1'). At the bottom are buttons for '?', 'OK', 'Cancel', 'Help', and 'Apply'. Red circles highlight the 'Time_Record' variable name, the 'Array' save format, and the '2-D array (concatenate along first dimension)' save format.

Block Parameters: To Workspace1

To Workspace

Write input to specified timeseries, array, or structure in a workspace. For menu-based simulation, data is written in the MATLAB base workspace. Data is not available until the simulation is stopped or paused.

To log a bus signal, use "Timeseries" save format.

Parameters

Variable name:

Time_Record

Limit data points to last:

inf

Decimation:

1

Save format: Array

Save 2-D signals as: 2-D array (concatenate along first dimension)

☒ Log fixed-point data as a fi object

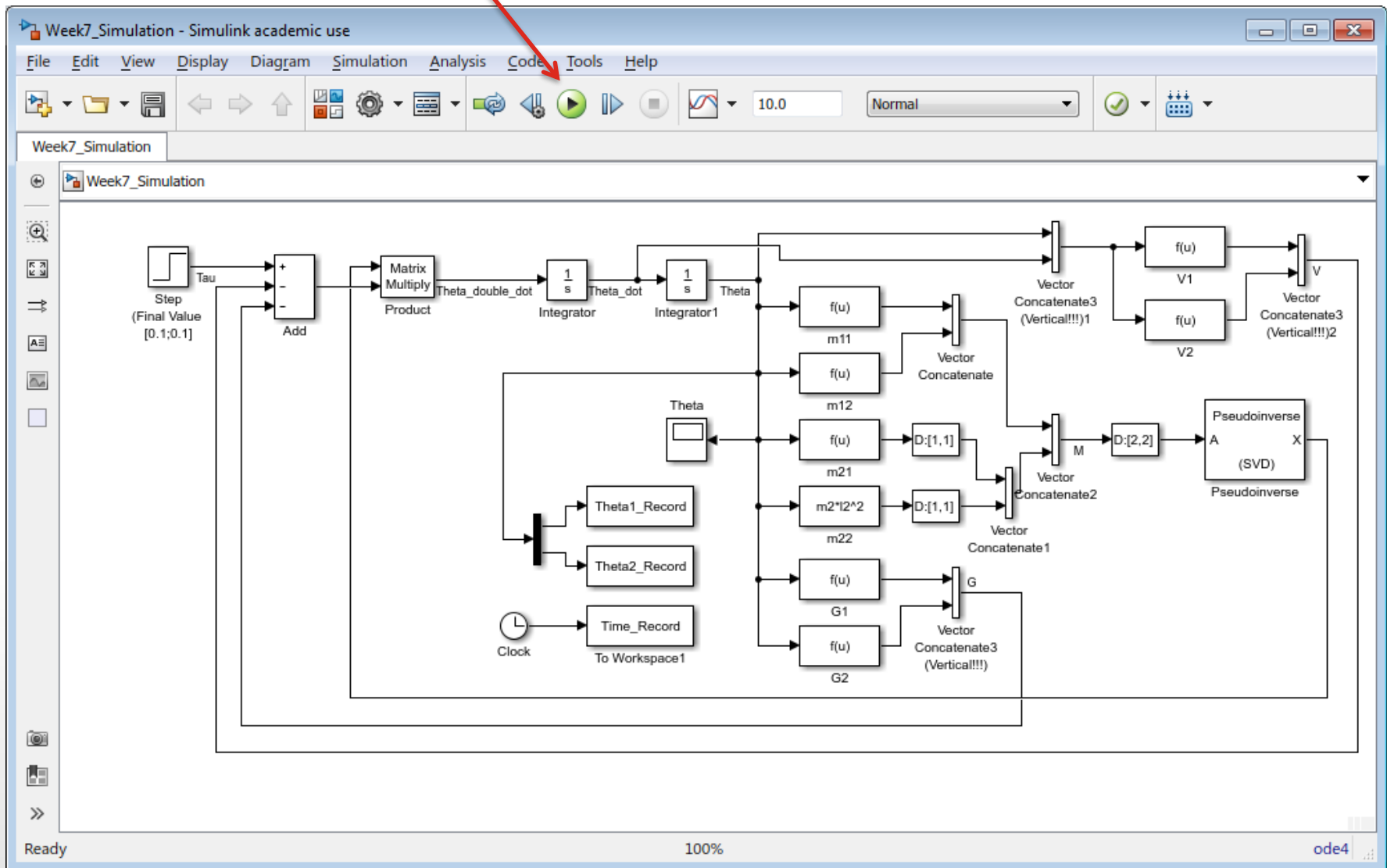
Sample time (-1 for inherited):

-1

OK Cancel Help Apply

MATLAB / Simulink

- Now click the “play” button.



MATLAB / Simulink

- Create another new script (m-file) and name it “animation”.

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Robot Parameters %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
|
m1 = 1;      % kg
m2 = 0.8;    % kg
l1 = 1;      % m
l2 = 0.8;    % m
g = 9.8;     % m/s^2

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calculate End Points %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

x1 = l1*cos(Theta1_Record);
y1 = l1*sin(Theta1_Record);
x2 = x1 + l2*cos(Theta1_Record+Theta2_Record);
y2 = y1 + l2*sin(Theta1_Record+Theta2_Record);

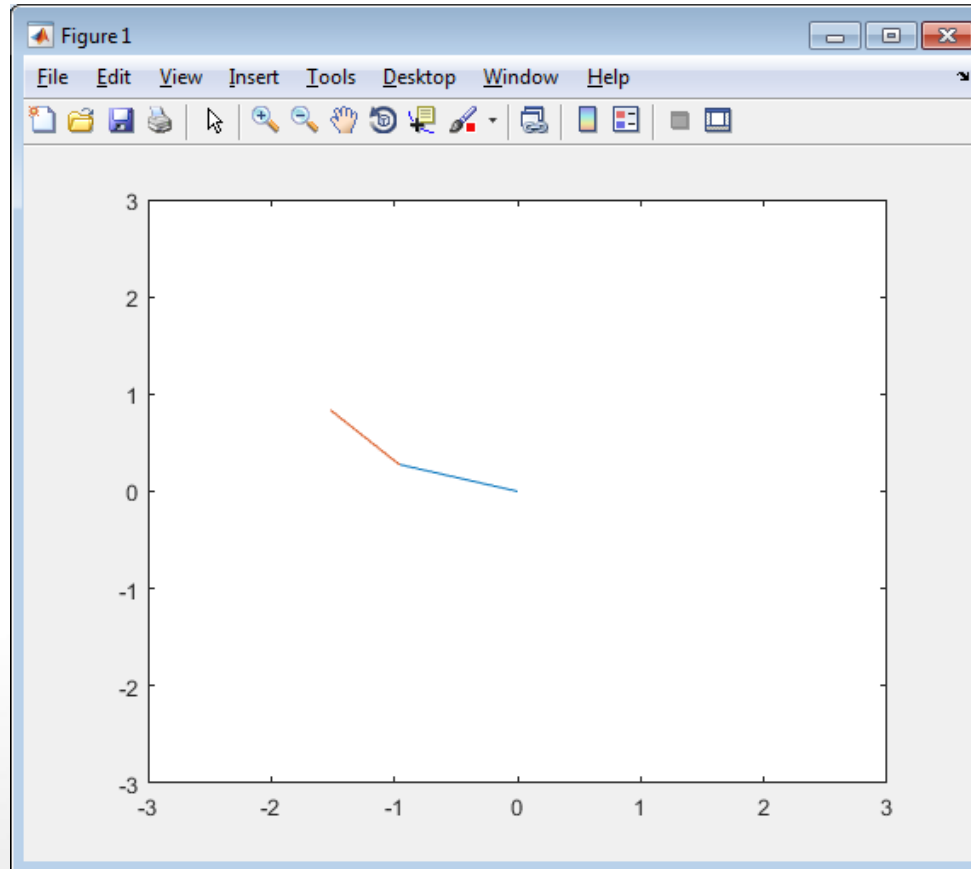
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plot %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = 1:100:length(x1)
    clf;                % Clear figure before new plot
    plot([0 x1(i)], [0 y1(i)]);
    axis([-3 3 -3 3]);
    hold on, plot([x1(i) x2(i)], [y1(i) y2(i)]);
    pause(0.1);
end

```

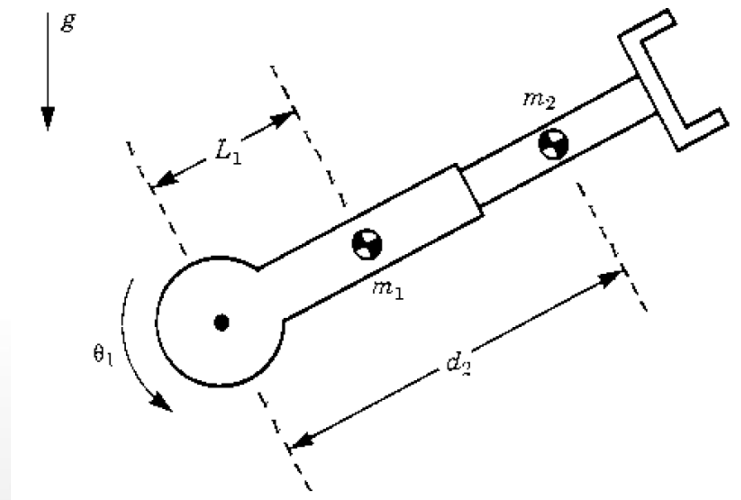
MATLAB / Simulink

- Run the m-file and you shall see the animation:



Exercise

- Try different torque values (including zero) and see the response.
- Add in viscous friction and see the response.
- Rebuild the model and animation based on the second robot example of today's class.



$$\underbrace{\begin{bmatrix} m_1 l_1^2 + I_{zz_1} + m_2 d_2^2 + I_{yy_2} & 0 \\ 0 & m_2 \end{bmatrix}}_{M(q)} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{d}_2 \end{bmatrix} + \underbrace{\begin{bmatrix} 2m_2 d_2 \dot{d}_2 \dot{\theta}_1 \\ -m_2 d_2 \dot{\theta}_1^2 \end{bmatrix}}_{V(q, \dot{q})} + \underbrace{\begin{bmatrix} m_1 l_1 g c_1 + m_2 d_2 g c_1 \\ m_2 g s_1 \end{bmatrix}}_{G(q)} = \tau$$

Thank you!

Have a good evening.

