# Simulink Hands-On Workshop

This workshop will guide you through some of the basic and intermediate skills for plant modeling, control design, and simulation in Simulink

Version 2.4

Table of Contents

[Section 1: Construction Basics. 1](#_Toc249261042)

[Section 2: Controlling Simulation. 2](#_Toc249261043)

[Section 3: Vectorizing Signals and Model Hierarchy. 2](#_Toc249261044)

[Section 4: Library Basics. 3](#_Toc249261045)

[Section 5: Masking Subsystems. 3](#_Toc249261046)

[Section 6: Buses. 3](#_Toc249261047)

[Section 7: More on Subsystems. 4](#_Toc249261048)

[Section 8: Referencing Models. 4](#_Toc249261049)

[Section 9: Controlling Sample Rates. 5](#_Toc249261050)

[Section 10: MATLAB Functions and Bus Objects. 5](#_Toc249261051)

## Section 1: Construction Basics

This section will show the Library Browser, how to begin to construct simple models, and how to run and save your new model.

[Open a model ready to proceed with this step.](matlab:Step_01)

**Standard Instructions:**

1.      Open the Simulink Library browser.

2.      Create a new model.

3.      Add a Constant block from 'Sources.'

4.      Add a Second Order Integrator block from 'Continuous.'

5.      Add a Scope from 'Sinks.'

6.      Add a Terminator block from ‘Sinks.’

7.      Connect the blocks.

8.      Change the constant value to -9.8.

9.      Press the Play button and double-click the Scope to view the results.

10.  Save your model.

[Open the completed model.](matlab:Step_02)

**Advanced Skills:**

* Automatically [align/distribute/resize](matlab:web([docroot%20'/simulink/ug/aligning-distributing-and-resizing-groups-of-blocks-automatically.html'%5d)) the blocks. (Be sure to save first.)
* Hide the Integrator and Terminator block names.
* Play with the [keyboard and mouse shortcuts](matlab:web([docroot%20'/simulink/ug/summary-of-mouse-and-keyboard-actions.html'%5d)) for zooming and panning.

## Section 2: Controlling Simulation

This section focuses on changing how Simulink numerically simulates your model.

[Open a model ready to proceed with this step.](matlab:Step_02)

**Standard Instructions:**

1. Label acceleration, velocity, and position signals by double-clicking on the signal.
2. Rename the Constant block to ‘gravity.’
3. Define the initial condition of the position (in the Integrator block) to be 500 and of the velocity to be 5.
4. Change simulation stop time to 100 seconds.
5. Find a way to stop the simulation automatically when the position reaches 0.
   1. Look in 'Sinks' for a way to stop the simulation.
   2. Look in 'Logic and Bit Operations' for help with the stop condition.
   3. Note the output data type for the condition block & the input type for the "Stop" block.
6. Run your model to make sure it stops when the position reaches 0.
7. From the 'Simulation' menu, choose 'Configuration Parameters.' Change the numerical solver to ode23 and make the max step size 1 second.

[Open the completed model.](matlab:Step_03)

**Advanced Skills:**

* Does the simulation stop exactly when the position is 0? Disable [zero-crossing detection](web(%5bdocroot%20'/simulink/ug/simulating-dynamic-systems.html#f7-9506'])) in the Compare to Constant block.
* In the Scope parameters, turn off the limit on the number of data points plotted. (Look in the Data History tab.)
* Plot the position and velocity as separate axes in the same scope.
* Using the [Model Explorer](matlab:web([docroot%20'/simulink/ug/the-model-explorer-overview.html'%5d)), set up your own [default Configuration Parameters.](matlab:web([docroot%20'/simulink/gui/simulink-preferences-window-configuration-defaults-pane.html'%5d))

## Section 3: Vectorizing Signals and Model Hierarchy

Simulink can pass vectors and matrices as signals, saving you time and extra lines. This section will also show you how to create a subsystem.

[Open a model ready to proceed with this step.](matlab:Step_03)

**Standard Instructions:**

1.      Make gravity a 3 element vector, [0 0 -9.8].

2.      Turn on the labels for signal dimensions. From the 'Format' menu, go to Port/Signal Displays and select Signal dimensions. Press Ctrl+D to [update your diagram.](matlab:web([docroot%20'/simulink/ug/updating-a-block-diagram.html'%5d))

3.      Modify initial conditions of position to [0 0 500] and velocity to [8 -10 5].

4.      Run the simulation and observe results. Why does the simulation stop right away?

5.      Use the Selector block from the Signal Routing library to select the third element of the position signal. Label the output "altitude" and connect it to the Compare to Zero block.

6. Open the Scope Parameter menu by clicking the second icon in the toolbar. Turn on Legends.

[Open the completed model.](matlab:Step_04)

**Advanced Skills:**

* Supplement your scopes with Simulink signal viewers. Right-click on the signal, go to 'Create & Connect Viewer' and select a Simulink viewer. [Why might you want to do this?](matlab:web([docroot%20'/simulink/ug/about-scope-blocks-signal-viewers-signal-logging-and-test-points.html'%5d))
* Save the scope's data to the workspace via the History tab of the Scope Parameter menu. Select "Inspect Logged Signals" from the model Tool menu and use the Data Inspector to visualize results.

## Section 4: Library Basics

This section shows the basics of libraries.

[Open a model ready to proceed with this step.](matlab:Step_04)

**Standard Instructions:**

1.      [Open drag\_library.mdl.](matlab:open('drag_library.mdl'))

2.      Drag the Variable Density subsystem to your falling object model above the integrators. Invert it by typing Ctrl+I.

3.      Set the Library Link Display to 'User' via the Format menu.

4.      Connect your altitude and velocity signals to the drag inports.

5.      Use a Gain block to scale the output signal of the drag model by 1/m. Why?

6.      Use a Sum block to add your drag and gravitational accelerations together and feed this cumulative result into the velocity integrator.

7.      Define the parameters in MATLAB: Cd = 0.333; A = 1.5; m = 150

[Open the completed model.](matlab:Step_05)

**Advanced Skills:**

* Find the '[Adding Libraries to the Library Browser](matlab:web([docroot%20'/simulink/ug/adding-libraries-to-the-library-browser.html'%5d))' section of Help.
* Create an slblocks.m file that adds the drag library to the Simulink Browser.
* Disable the link to the library by right-clicking on the block from within your lander model.
* Make changes to the disabled library then push them to the library file by Resolving the Link.

## 

## Section 5: Masking Subsystems

This section shows how to mask subsystems and add model callbacks.

[Open a model ready to proceed with this step.](matlab:Step_05)

**Standard Instructions:**

1. [Open drag\_library.mdl](matlab:open('drag_library.mdl')) and unlock the library via the Edit menu.
2. Right-click on Variable Density Model and choose 'Create Mask.'
3. Right-click on Constant Density Model and choose 'Edit Mask.'
4. Create a mask on Variable Density similar to the one on Constant Density. [More help on masking.](matlab:web([docroot%20'/simulink/masking.html'%5d))
5. Save drag\_library.mdl and update your main model.
6. From the menu, go to 'File', 'Model Properties.' Click on the [callbacks](matlab:web([docroot%20'/simulink/ug/using-callback-functions.html'%5d)) tab and select 'PostLoadFcn.' Type MATLAB code to define the mass variable m in the given field.

[Open the library with masked subsystems.](matlab:drag_library_masked)

[Open the completed model.](matlab:Step_06)

**Advanced Skills:**

* Define the mass in the [model workspace.](matlab:web([docroot%20'/simulink/ug/using-model-workspaces.html'%5d))
* Add a configurable subsystem template to the drag library and make drag a configurable subsystem in your model. Use the set\_param command to programmatically switch between the two drag model configurations.
* Open variant\_drag.mdl and observe a Variant Subsystem. Add this block to your Simulink model. Right-click on block to check Subsystem Parameters and override variant choice.

## 

## Section 6: Buses

This section introduces buses as a means to organize your signals.

[Open a model ready to proceed with this step.](matlab:Step_06)

**Standard Instructions:**

1. Click and hold the mouse as you drag the cursor across the model. Make sure all blocks are selected.
2. Convert to a subsystem by right-clicking on one of the blocks and selecting "Create Subsystem." Name the subsystem "Plant."
3. Double-click on Plant to open it. Go to Sources in the library browser and add an inport ("In1") to Plant and name it "thrust."
4. Add the thrust to the gravity and drag accelerations (remember to also scale it by 1/m). At the root level, make thrust a constant set to [0 0 1000].
5. From the Signal Routing library, grab a Bus Creator block and put it in the Plant subsystem. Connect the position and velocity signals to the Bus Creator inputs. Feed this [bus](matlab:web([docroot%20'/simulink/slref/simulink.bus.html'%5d)) into a new outport named "State".
6. Move the scope blocks from the Plant subsystem to the top model layer. Use a Bus Selector block at the root level to get the position and velocity signals back out of the bus and connect them to the Scopes.
7. Run your model and observe the signal lines.

[Open the completed model.](matlab:Step_07)

**Advanced Skills:**

* Name the bus signal and [log](matlab:web([docroot%20'/simulink/signal-logging.html'%5d)) it [to the workspace.](matlab:web([docroot%20'/simulink/slref/toworkspace.html'%5d)) How is the data stored?
* Use From and Goto blocks to connect position and velocity to the Bus Creator.

## 

## Section 7: More on Subsystems

This section talks more about types of subsystems and how they affect how Simulink simulates your model. Algebraic loops are also covered in the Advanced section.

[Open a model ready to proceed with this step.](matlab:Step_07)

**Standard Instructions:**

1.      Change the constant ''thrust'' block to 'desired velocity' and set it to [0 0 -0.5].

2.      Add a Sum block and set the second sign to negative.

3.      Difference desired velocity with velocity and label the Sum's output as 'error.'

4.      Add a PID Controller block in the Continuous library and make ''error'' its input. Supply its output to the Plant model.

5.      Double-click on the Controller block and set the controller to PI. Set the gains to 50 and 5.

6.      Create a subsystem that includes the reference velocity, sum block, and PID block. Label this subsystem 'Controller.'

7.      Connect Scopes to 'thrust' and 'error.'

8.      Show the block execution order. Go to the menu, under Format, Block Displays, choose Sorted Order. Update the diagram with Ctrl+D.

9.      Right-click on the Controller subsystem and go to Subsystem Parameters. Check the box for ''Treat as atomic unit.'' Update the diagram.

[Open the completed model.](matlab:Step_08)

**Advanced Skills:**

* Make the Plant subsystem an atomic subsystem too. Update the diagram. [What error do you get?](matlab:web(%5bdocroot%20'/simulink/ug/simulating-dynamic-systems.html#f7-19688']))
* Add a Unit Delay block to the feedback signal from the Plant to the Controller.
* Alternatively, in the Plant Subsystem Parameters, check the box for 'Minimize algebraic loop occurrences.' Update/play model. Uncheck that box in the Plant subsystem and try checking it in the Controller subsystem. Do you get an error?
* Uncheck the algebraic loop box in the Controller subsystem and change the Plant back to a virtual subsystem.
* Use the [Simulink Debugger](matlab:web([docroot%20'/simulink/debugging.html'%5d)) to view block execution order.

[Open a model ready for control design.](matlab:Step_08_controls)

**Control Design Skills:**

* Split the velocity signal into X, Y, and Z components and use 3 PID control blocks (or [open existing model](matlab:Step_08_controls)).
* Make sure all controllers are PI with gain values of 1 and 1.
* Double-click on the Z controller and set the controller’s lower saturation limit to zero in the PID Advanced tab.
* Push the Tune button in the Main tab.
* Adjust response time. Adjust phase margin and bandwidth by setting the Design Mode to Extended.
* Attempt to tune the X controller and see if the response is stable.
* If it isn’t, set the Z controller gains to 50 and 5. Push the “Design with new plant model” button and linearize about a simulation snapshot of 10 secs.
* Set the Plot to Step disturbance rejection and tune the controller.

**Advanced Control Design Skills:**

* Try other types of controllers (PID, P, PD, etc).
* Observe cross-coupling by changing one controller and trying to Tune another.
* Discretize the controllers and discover what conditions lead to instability in the linearized model.

## 

## Section 8: Referencing Models

This section introduces referenced models, creating them and simulating with them.

[Open a model ready to proceed with this step.](matlab:Step_08)

**Standard Instructions:**

1. Right-click on the Controller subsystem and select 'Convert to Model Block.' Take note of any errors or warnings you get, but continue the conversion.
2. Delete the Controller subsystem and drag in the block from the newly created model, untitled.mdl.
3. Double-click on the new Controller block. Open up the Plant subsystem. What do you notice about the title bars of each? Also look in the Current Directory Browser.
4. Go to Configuration Parameters for the Controller model. Under Optimization -> Signals and Parameters, check "Inline parameters." Run the simulation.
5. Right-click on the Controller Model block and select "ModelReference Parameters". Change the Simulation Mode from Normal to Accelerator. Run the simulation

[Open the completed model.](matlab:Step_09)

**Advanced Skills:**

* What files are needed to run your model now? Check out Tools -> Model Dependencies -> [Model Dependency Viewer](matlab:web([docroot%20'/simulink/ug/model-dependency-viewer.html'%5d)) and [Generate Manifest.](matlab:web([docroot%20'/simulink/ug/model-dependencies.html#bq2ik30']))
* What's the [command line way](matlab:web([docroot%20'/simulink/slref/simulink.subsystem.converttomodelreference.html'%5d)) to convert an atomic subsystem to a model reference? Can you go back to a subsystem from a model?

## Section 9: Controlling Sample Rates

This section talks about how to view and control sample rates within a model.

[Open a model ready to proceed with this step.](matlab:Step_09)

**Standard Instructions:**

1. Double-click on the PI Controller. Set the Time-domain to Discrete. Set the sample time to 0.5.
2. In the main model, Set [Sample Time Display](matlab:web([docroot%20'/simulink/ug/how-to-view-sample-time-information.html'%5d)) to "Color" via the Format menu. Note the error message.
3. Double-click the Controller's thrust outport and set the Sample Time to inherited via the Signal Attributes tab. Update the diagram.
4. Double-click the Controller's velocity inport and set the Sample Time to inherited via the Signal Attributes tab. Update the diagram.
5. Add Rate Transition blocks from the Signal Attributes library before and after the referenced model. Update the diagram.
6. Change the Controller's Solver options to Fixed Step Discrete to avoid warnings.

[Open the completed model.](matlab:Step_10)

**Advanced Skills:**

* What if you had continuous states in your referenced model?
* What are [some ways](matlab:web([docroot%20'/simulink/configuration-sets.html'%5d)) to organize your configuration sets? Specifically, try using cset=Simulink.ConfigSet to create the configuration sets to be used in each of your models and then load that configuration set in a callback and point your model to it.

## Section 10: MATLAB Functions and Bus Objects

This section shows how to bring MATLAB code into Simulink and how to create bus objects.

[Open a model ready to proceed with this step.](matlab:Step_10)

**Standard Instructions:**

1. Change the Controller such that desired velocity is a function of altitude. Right-click on the reference model, select ModelReference parameters, and change the Model Name to 'Controller\_Step\_11.' Note the inport's use of stateBus in the Signal Attributes.
2. Change the input to the Controller; supply State instead of velocity.
3. Go to Tools -> Bus Editor and create a new bus named stateBus. It should have two signals in it, one called position with has dimension of 3 and one called velocity which has dimension of 3. Hit Apply and look in your workspace for the new bus object.
4. Save stateBus to a mat file. Add a ''load <mat file>'' to the main model''s PostLoadFcn.
5. Run your model and see how it looks!

[Open the completed model.](matlab:Step_11)

**Advanced Skills:**

* When would you want to use MATLAB Function and when would you want to use an [S-Function?](matlab:web([docroot%20'/simulink/sfg/using-s-functions-in-models.html'%5d))
* Click the link in the model to generate C code from your Controller.
* Click the link in the model to generate PLC code from your Controller.
* Click the link in the model to generate HDL code from your Controller.