Simscape™ Multibody™

Getting Started Guide

MATLAB&SIMULINK®

MathWorks®

How to Contact MathWorks

	Latest news:	www.mathworks.com
	Sales and services:	www.mathworks.com/sales_and_services
	User community:	www.mathworks.com/matlabcentral
	Technical support:	www.mathworks.com/support/contact_us
	Phone:	508-647-7000

The MathWorks, Inc.
1 Apple Hill Drive
Natick, MA 01760-2098

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Revision History

March 2012	Online only	New for Version 4.0 (Release R2012a)
September 2012	Online only	Revised for Version 4.1 (Release R2012b)
March 2013	Online only	Revised for Version 4.2 (Release R2013a)
September 2013	Online only	Revised for Version 4.3 (Release R2013b)
March 2014	Online only	Revised for Version 4.4 (Release R2014a)
October 2014	Online only	Revised for Version 4.5 (Release R2014b)
March 2015	Online only	Revised for Version 4.6 (Release R2015a)
September 2015	Online only	Revised for Version 4.7 (Release R2015b)
March 2016	Online only	Revised for Version 4.8 (Release R2016a)
		(Renamed from *SimMechanics™ Getting Started*)
September 2016	Online only	Revised for Version 4.9 (Release R2016b)
March 2017	Online only	Revised for Version 5.0 (Release R2017a)
September 2017	Online only	Revised for Version 5.1 (Release R2017b)
March 2018	Online only	Revised for Version 5.2 (Release R2018a)
September 2018	Online only	Revised for Version 6.0 (Release R2018b)
March 2019	Online only	Revised for Version 6.1 (Release R2019a)
September 2019	Online only	Revised for Version 7.0 (Release R2019b)
March 2020	Online only	Revised for Version 7.1 (Release R2020a)
September 2020	Online only	Revised for Version 7.2 (Release R2020b)
March 2021	Online only	Revised for Version 7.3 (Release R2021a)
September 2021	Online only	Revised for Version 7.4 (Release R2021b)
March 2022	Online only	Revised for Version 7.5 (Release R2022a)
September 2022	Online only	Revised for Version 7.6 (Release R2022b)
March 2023	Online only	Revised for Version 7.7 (Release R2023a)

Contents

Introduction to Simscape Multibody Software

1

Introduction to Simscape Multibody Software

Simscape Multibody Product Description

Model and simulate multibody mechanical systems

Simscape Multibody provides a multibody simulation environment for 3D mechanical systems, such as robots, vehicle suspensions, construction equipment, and aircraft landing gear. You can model multibody systems using blocks representing bodies, joints, constraints, force elements, and sensors. Simscape Multibody formulates and solves the equations of motion for the complete mechanical system. You can import complete CAD assemblies, including all masses, inertias, joints, constraints, and 3D geometry, into your model. An automatically generated 3D animation lets you visualize the system dynamics.

Simscape Multibody helps you develop control systems and test system-level performance. You can parameterize your models using MATLAB® variables and expressions, and design control systems for your multibody system in Simulink®. You can integrate hydraulic, electrical, pneumatic, and other physical systems into your model using components from the Simscape family of products. To deploy your models to other simulation environments, including hardware-in-the-loop (HIL) systems, Simscape Multibody supports C-code generation.

Key Features

- Single environment for simulating 3D mechanical systems together with multidomain physical systems and control algorithms in Simulink
- Part definition tools, including standard geometry, extrusions defined in MATLAB, and STEP files
- Part definition by reading native CAD files, including CATIA
- CAD assembly import from SolidWorks®, PTC Creo™ (Pro/ENGINEER™), Autodesk Inventor® , and Onshape®
- Mechanical joints and constraints covering standard and custom kinematic relationships
- 3D animation of multibody system dynamics
- Support for C-code generation (with Simulink Coder™)

Multibody Model Anatomy

| **In this section...** |
| --- |
| "Basic Model Components" on page 1-3 |
| "Model Actuation" on page 1-4 |
| "Dynamical Sensing" on page 1-5 |

A multibody system is a set of bodies linked through joints and bound by kinematic constraints such as gears. Forces and torques of various types enable you to actuate the various bodies, while sensors enable you to sense the resulting motion. You can use Simscape Multibody blocks to represent these components and analyze the dynamic responses of the system. For more information about multibody dynamics simulations, see "Multibody Dynamics".

Basic Model Components

Consider the Simscape Multibody four-bar featured example. You can open this model at the MATLAB command prompt by entering `sm_four_bar`. The model consists of solid body subsystems interconnected through Joint blocks. The Joint blocks define the degrees of freedom available between bodies. The body subsystems define, through an underlying block diagram, the solid properties of the body.

1: Body   2: Joint   3: Rigid Transform   4: World Frame
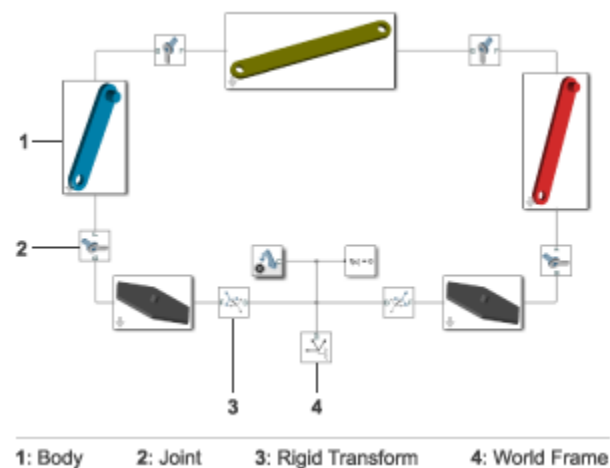
Basic Components of sm_four_bar Model

The body subsystems in turn consist of Extruded Solid blocks interconnected through Rigid Transform blocks. The Extruded Solid blocks provide the solid properties, including geometry, inertia, color, and frames—axis triads that encode all position and orientation information in a model. The Rigid Transform blocks rotate and translate the solids to ensure proper body assembly.

The figure shows the underlying block diagram for a binary link body subsystem. This diagram includes three Solid blocks, representing the central and distal portions of the binary link. Two Rigid Transform blocks translate the distal solids to their proper locations near the ends of the central solid.
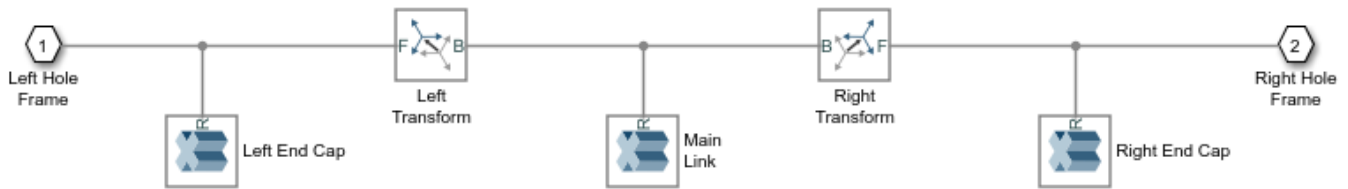
Body Subsystem Components

Other common blocks in a model include:

- World Frame — Adds an inertial reference frame to the model.
- Rigid Transform — Rotates and translates bodies and other frames relative to each other.
- Mechanism Configuration — Defines the gravitational constant in a model.
- Solver Configuration — Sets Simscape local solver options such as types, tolerances, and time steps. This is the only required block in a model.

Model Actuation

You can actuate a model by applying a force or torque to a body or to a joint. To represent forces and torques acting on a body, Simscape Multibody provides a Forces and Torques library. Add a block from this library and connect it to the body frame(s) that you want to apply the force or torque to.

Block	Function
External Force and Torque	General force and/or torque originating outside of the multibody model
Internal Force	General force pair between two arbitrary frames
Spring and Damper Force	Spring-damper force pair between two arbitrary frames
Inverse Square Law Force	Force pair with inverse dependence on the square distance between two arbitrary frames (e.g., Coulomb electrostatic forces)
Gravitational Field	Gravitational pull of a point mass on all bodies as a function of their distances to the point mass itself

The figure shows a four-bar model with an External Force and Torque block for force and torque prescription at a crank link frame.
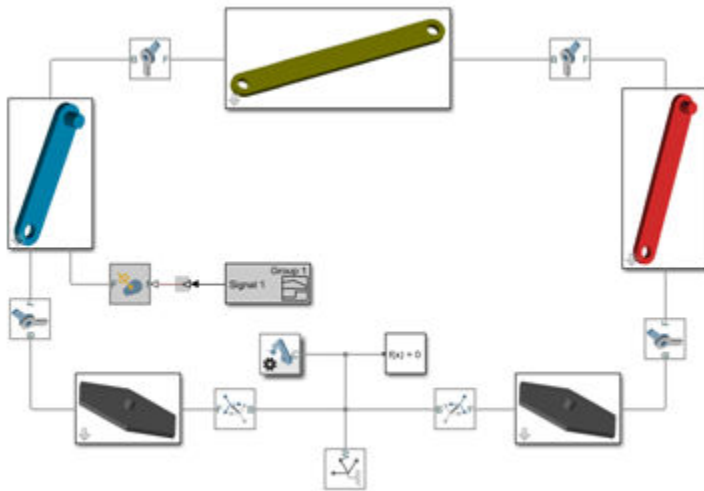
To specify the force or torque acting at a joint, Simscape Multibody provides a selection of actuation inputs directly in the joint blocks. Each joint primitive—the basic component of a joint block—provides a selection of actuation inputs specific to that primitive.

Joint actuation inputs can be of two types:

- Motion — Specify the time-varying trajectory of a given joint primitive.
- Force or torque — Specify the time-varying actuation force or torque acting at a given joint primitive.

The figure shows a four-bar model with an actuation torque acting on a revolute joint.
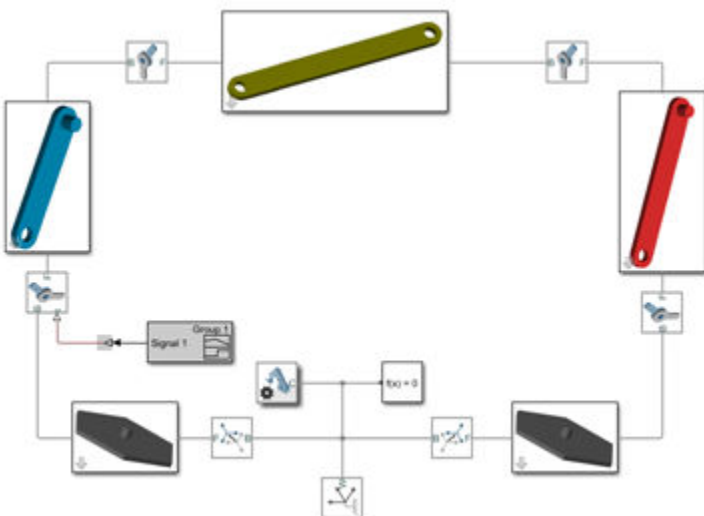
Dynamical Sensing

You can sense various dynamical variables between frame pairs, e.g., for analysis or control design. Sensing outputs can be of two types:

- Motion — Compute and output the relative position, velocity, or acceleration between two Simscape Multibody frames. You can sense motion between joint frames, by using the sensing capability of joint blocks, or between arbitrary frames, by using the Transform Sensor block.

- Force or torque — Compute and output the forces and torques acting between two Simscape Multibody frames. You can sense force and torque between the port frames of certain Forces and Torques blocks, such as the Inverse Square Law Force block, or between the port frames of a joint block.

Joint blocks enable you to sense different types of forces and torques between their respective port frames, including:

- Actuation force or torque acting at a given joint primitive.

- Constraint force and torque acting joint-wide to prevent motion normal to the joint degrees of freedom.

- Total force and torque, including constraint and joint primitive actuation contributions, acting joint-wide.

The figure shows a four-bar model with a Transform Sensor block for trajectory coordinate sensing between a coupler link frame and the world frame.
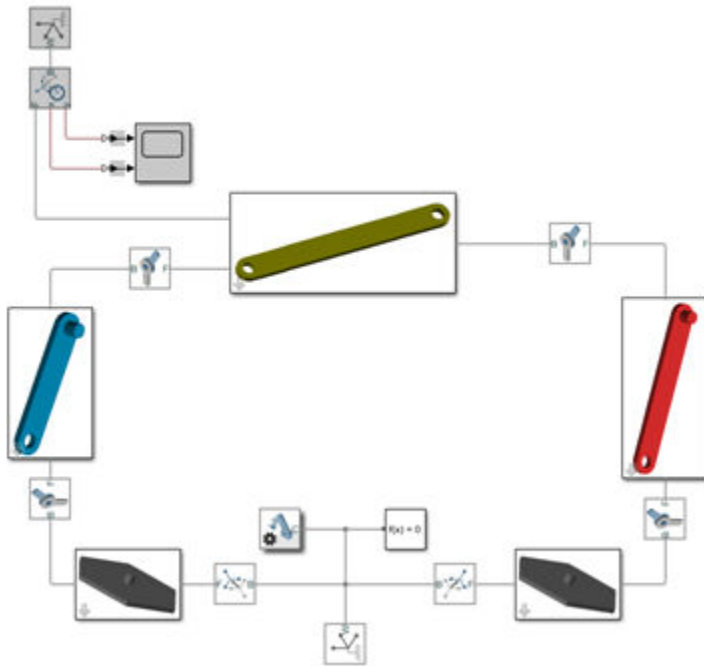
See Also

Related Examples

- "Model a Simple Link" on page 1-10
- "Model a Simple Pendulum" on page 1-15
- "Analyze a Simple Pendulum" on page 1-18

Creating a Multibody Model

In this section...

A multibody dynamics system is a set of interconnected bodies that have forces acting on them. You can use Simscape Multibody to model the components in the system and analyze their dynamic behaviors.

Step 1: Study the System to Model
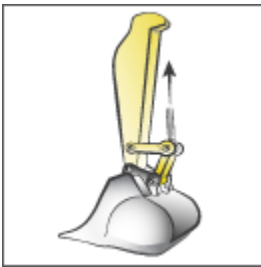
Identify the relevant bodies, joints, and constraints to incorporate in your model. Consider the forces and torques acting between bodies or directly at joints.

Step 2: Model the Bodies
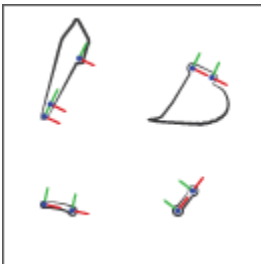
Specify the solid properties of the various bodies. Add frames to the bodies so that you can connect joints and constraints, apply forces and torques, and sense motion. See "Model a Simple Link" on page 1-10 for an example showing how to model a simple body—one comprising a single solid. See "Create a Compound Geometry" for a more elaborate example showing how to model a compound body—one comprising multiple solids.

Step 3: Assemble the Multibody System
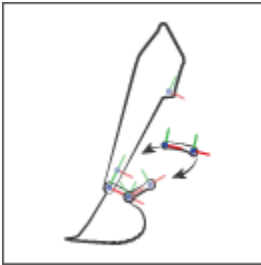
Connect the bodies by pairs through joints. Constrain body motion through gears and other kinematic constraints. See "Model an Open-Loop Kinematic Chain" for an example showing how to connect bodies through joints. See "Model a Compound Gear Train" for an example showing how to constrain motion through gears.

Step 4: Specify and Sense Dynamic Variables

Model external loads, interactions between bodies, and joint actuation inputs. Sense forces, torques, and motion variables as functions of time. See "Specify Joint Actuation Torque" for an example showing how to specify a joint actuation torque. See "Sense Motion Using a Transform Sensor Block" for an example showing how to sense frame position using the Transform Sensor block. For more information about multibody dynamics simulations, see "Multibody Dynamics".

See Also

More About

- "Model an Open-Loop Kinematic Chain"
- "Model a Compound Gear Train"
- "Specify Joint Actuation Torque"
- "Sense Motion Using a Transform Sensor Block"

Start a Model from a Template

You can start a new Simscape Multibody model directly from the MATLAB command prompt by entering `smnew`. The `smnew` function opens a Simscape Multibody model template with commonly used blocks and variable-step automatic solver selection for best performance. The figure shows the model template.
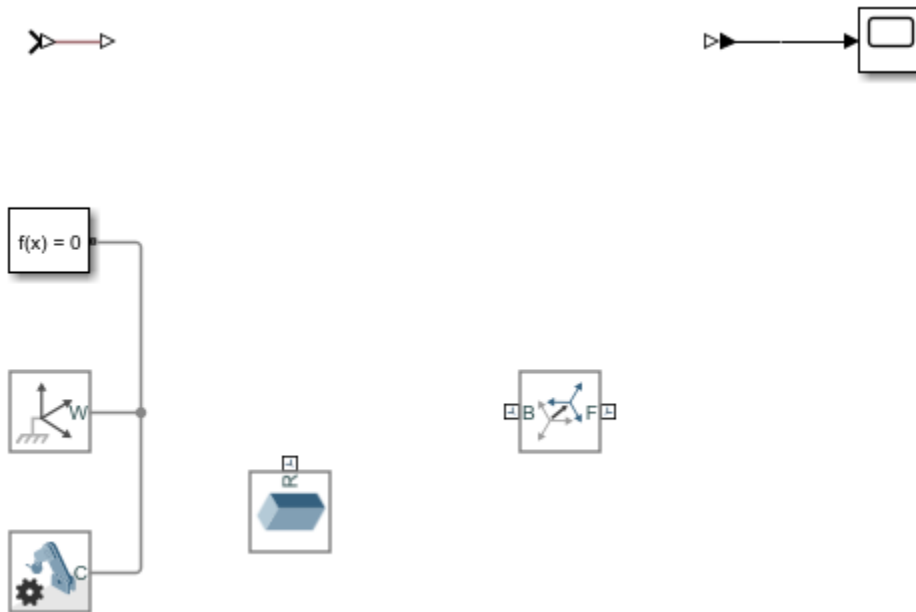
You can also open the Simscape Multibody block library from the MATLAB command prompt by entering `sm_lib`. Browse the library to select blocks for your model. Alternatively, type the block name in the Simulink Model window and select a match from the Simulink search results.

See "Model a Simple Link" on page 1-10 to model a body in the Simscape Multibody model template. See "Model a Simple Pendulum" on page 1-15 to integrate the body in a multibody mechanism.

See Also
Brick Solid | PS-Simulink Converter | Rigid Transform | Simulink-PS Converter | `smnew` | `sm_lib`

More About
- "Model a Simple Link" on page 1-10
- "Model a Simple Pendulum" on page 1-15

Model a Simple Link

| In this section... |
|---|
| "Build Model" on page 1-10 |
| "Generate Subsystem" on page 1-11 |
| "Visualize Model" on page 1-13 |
| "Save Custom Block" on page 1-13 |

Mechanical links are common building blocks in linkages, mechanisms, and machines. In this example, you learn how to model a simple mechanical link using Simscape Multibody. For simplicity, the model assumes the link has a brick shape and two end frames.
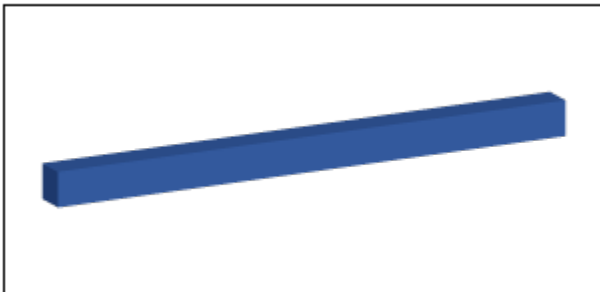
Build Model

1 At the MATLAB command line, enter `smnew`. A Simscape Multibody model template with commonly used blocks opens.

2 Delete the Simulink-PS Converter, PS-Simulink Converter, and Scope blocks because they are not used in this example.

3 Show the names of the Rigid Transform and Brick Solid blocks. Right-click the block and select **Format > Show Block Name > On**.

4 Duplicate the Rigid Transform block to add a second end frame to the link.

5 Flip the Rigid Transform1 block so that you can connect the **B** ports of two Rigid Transform blocks to each other and the Brick Solid block.

6 Connect the remaining blocks like the following figure.
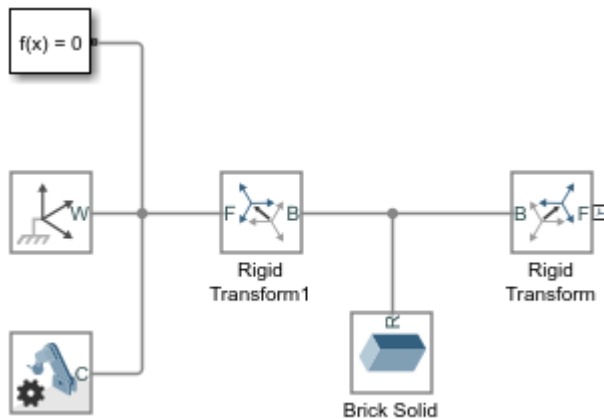
7 In the Brick Solid dialog box, specify the following parameters. These parameters define the simple link's physical properties, such as shape, mass, and appearance.

| Parameter	Value	Units
Geometry > Dimensions	`[L W H]`	`cm`
Inertia > Density	`rho`	`kg/m^3`
Graphic > Visual Properties > Color	`rgb`	Not applicable

8 In the Rigid Transform blocks, specify the following parameters. These parameters specify the locations of two end frames on the simple link. After you enter the parameters, the Brick Solid and Rigid Transform blocks will be highlighted in red because you have not defined the variables yet. This issue will be solved after imputing all the numerical values for the parameters, which will be described in the following section.

| Parameter	Rigid Transform1	Rigid Transform	Units
Translation > Method	`Standard Axis`	`Standard Axis`	Not applicable
Translation > Axis	`-X`	`+X`	Not applicable
Translation > Offset	`L/2`	`L/2`	`cm`

Generate Subsystem

You can build a complex multibody system using several simple models, such as a simple link model. The physical parameters of these simple models usually need to be adjusted to fit different design requirements. To simplify the parameter adjusting process, you can create Subsystem blocks for these simple models. A Subsystem block enables you to update many parameters in a single place—the Subsystem block dialog box. In this section, you learn how to create a Subsystem block for the simple link model.

1 Select the Brick Solid block and the two Rigid Transform blocks by holding shift and clicking the blocks.

2 Right-click one of the selected blocks and select **Create Subsystem from Selection**. Simulink adds a new Subsystem block that contains the Brick Solid and Rigid Transform blocks.
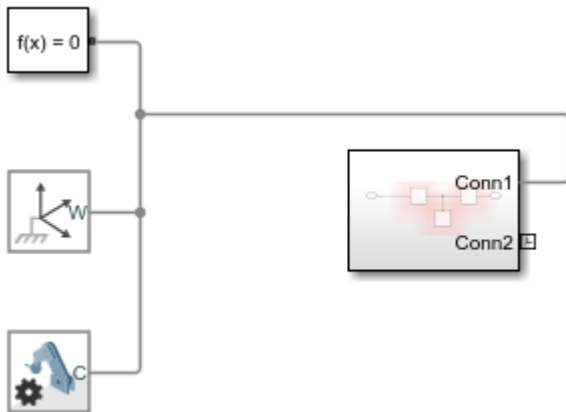
3   Double-click the Subsystem box. A new tab displays the children blocks of the Subsystem block. Double-click the Conn1 and set **Port location on parent subsystem** to left. Click **OK** to apply the change and navigate back to the parent model by clicking the up arrow button next to the Subsystem tab.
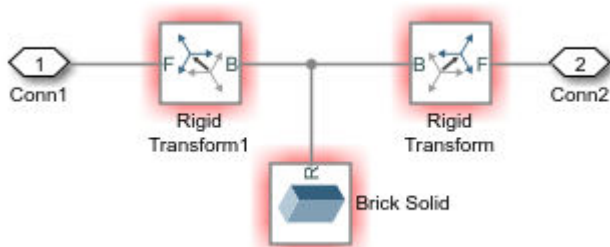
4   Adjust the size and location of the Subsystem block as shown in the figure.
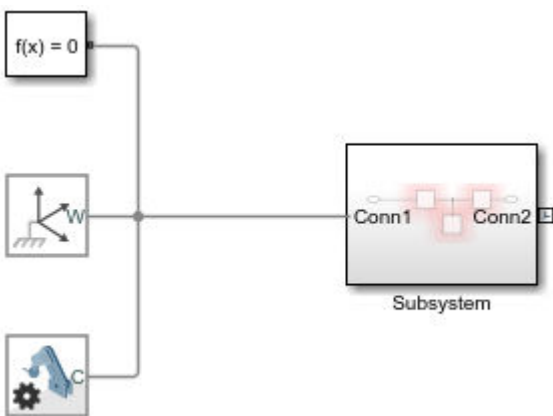
5   Right-click the Subsystem block and select **Mask > Create Mask**. The Mask Editor window opens, where you can define the MATLAB variables that you entered in the Brick Solid and Rigid Transform block dialog boxes.

6   Click the **Parameters & Dialog** tab and click **Edit** five times to create five **Edit** fields. In the **Edit** fields, specify the following parameters and click **OK**. The **Prompt** property specifies the names of the parameters that you can enter in the Subsystem block parameter window. The **Name** property specifies their corresponding MATLAB variables.

Prompt	Name
Length (cm)	L
Width (cm)	W
Thickness (cm)	H
Density (kg/m^3)	rho
Color [R G B]	rgb

7   Double-click the Subsystem block. Enter the following numerical values in the Subsystem Block Parameters window and click **OK**. These values specify the shape of Brick Solid and the location of Rigid Transform blocks.

| Parameter	Value
Length (cm)	20
Width (cm)	1
Thickness (cm)	1
Density (kg/m^3)	2700
Color [R G B]	[0.25 0.40 0.70]

Visualize Model

Run the model. The Mechanics Explorer opens with a front view of the simple link model. To see the

3D view of the model, click the isometric view button ![icon]. To view the frames in the model, in the Mechanics Explorers tab, select **View > Show Frames** in the Mechanics Explorer menu bar.
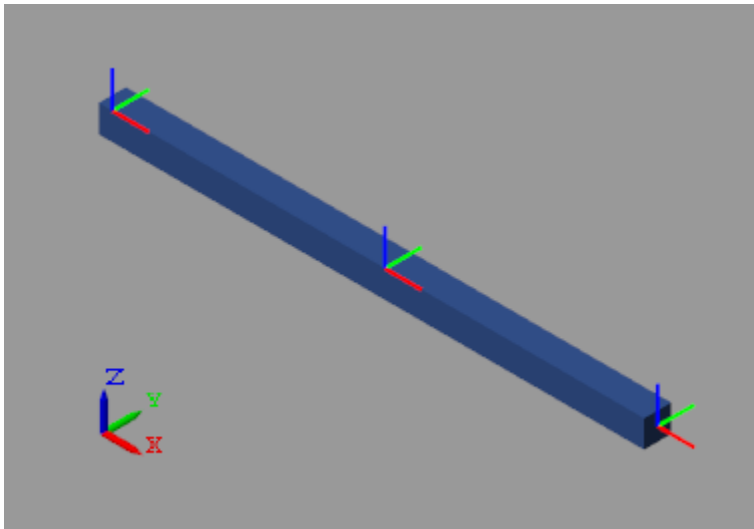
Save Custom Block

To reuse this Subsystem block in other examples, such as the "Model a Simple Pendulum" on page 1-15 example, you can change its name to Simple Link and save it as a custom block. For more information, see "Design and Create a Custom Block".
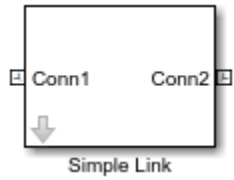
Simple Link Custom Block

See Also

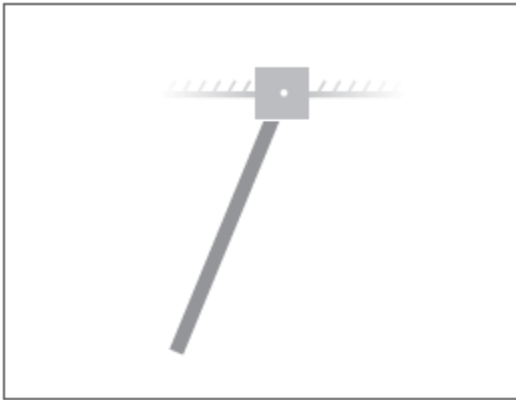Brick Solid | PS-Simulink Converter | Rigid Transform | Scope | Simulink-PS Converter

More About

- "Model a Simple Pendulum" on page 1-15
- "Design and Create a Custom Block"

Model a Simple Pendulum

In this section...

The pendulum is the simplest mechanical system you can model. This system contains two bodies, a link and a fixed pivot, connected by a revolute joint. In this example, you learn how to model and simulate a pendulum using the custom link block you created in the "Model a Simple Link" on page 1-10. A Revolute Joint block provides the rotational degree of freedom between the link and the pivot fixed at the world frame.

Build Model

1   At the MATLAB command prompt, enter `smnew`. The Simscape Multibody model template with commonly used blocks opens up.

2   Delete the Simulink-PS Converter, PS-Simulink Converter, Scope, and Rigid Transform blocks. You do not need them in this example.

3   Add to the model the Simple Link custom block created in the "Model a Simple Link" on page 1-10.

4   Add a Revolute Joint block to the model. You can find this block in the **Simscape > Multibody > Joints** library. This block provides one rotational degree of freedom between its frames.

5   Connect the blocks as shown in the image. The port orientation of the Revolute Joint block becomes important when you specify joint state targets, prescribe joint actuation inputs, or sense joint dynamic variables. The Revolute Joint block interprets each quantity as that applied to the follower frame with respect to the base frame, so switching the port connections can affect model assembly and simulation.
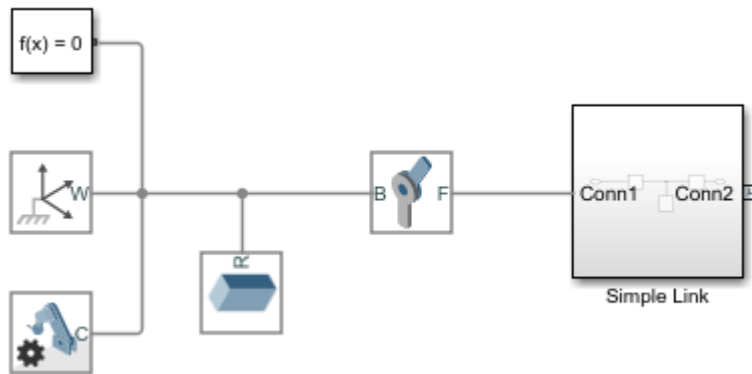
6 In the Brick Solid block dialog box, specify the following parameters. This block connects rigidly to the World frame and therefore has no effect on the dynamics of the multibody model. For more information about multibody dynamics simulations, see "Multibody Modeling". You can leave the inertia parameters in their default values.

Parameter	Value	Units
Geometry > Dimensions	`[4 4 4]`	Change to `cm`
Graphic > Visual Properties > Color	`[0.80 0.45 0]`	N/A

Specify Gravity

The Revolute Joint block uses the common Z-axis of the base and follower frames as the joint rotation axis. To ensure the pendulum oscillates under the effect of gravity, change the gravity vector so it no longer aligns with the Z-axis. To do this, in the Mechanism Configuration block dialog box, set **Uniform Gravity > Gravity** to `[0 -9.81 0]`.

Set Pendulum Starting Position

You can specify the desired joint angle using the **State Targets** menu in the Revolute Joint block dialog box. To do this, select **State Targets > Position** and enter the desired joint angle. For this tutorial, you can leave the angle in its default value, which corresponds to a horizontal pendulum starting position.

Configure Solver

1 Open the Configuration Parameters. In the **Modeling** tab, click **Model Settings**.

2 In the **Solver** tab, set the **Solver** parameter to `ode15s (stiff/NDF)`. This solver is the recommended choice for physical models.

3 Set **Max step size** to `0.01` and click **OK**. The small step size increases the simulation accuracy and produces a smoother animation in Mechanics Explorer. Small step sizes can have a detrimental effect on simulation speed but, in such a simple model, a value of `0.01` provides a good balance between simulation speed and accuracy.

Assemble Model

Update the block diagram. In the **Modeling** tab, click **Update Model**. Mechanics Explorer opens with a 3-D view of the model in its initial configuration.

In the Mechanics Explorer toolstrip, check that the **View convention** parameter is set to `Y up (XY Front)`. This view convention ensures that gravity is vertically aligned on your screen. Select a standard view button to refresh the Mechanics Explorer display. The figure shows a front view of the model. Save the visualization settings by clicking the Save explorer configuration to model button

.
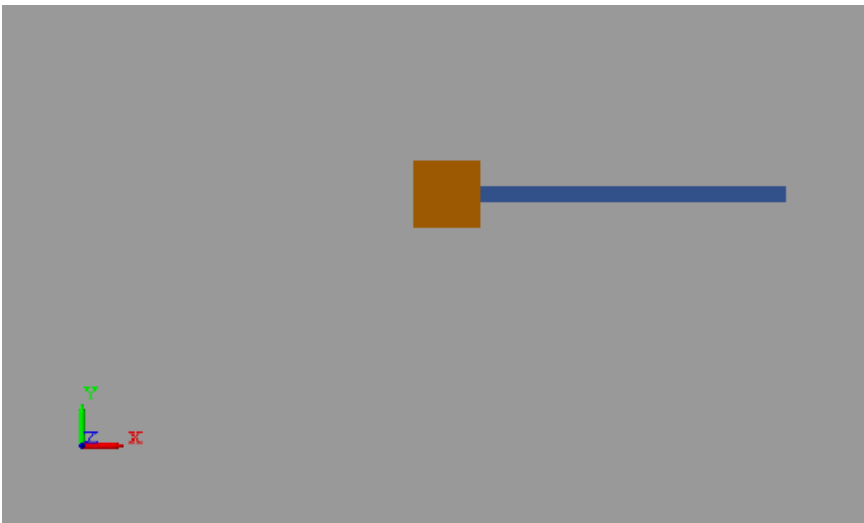
Simulate Model

Run the simulation. Mechanics Explorer plays a physics-based animation of the pendulum model.

Save Model

Save the model in a convenient folder under the name simple_pendulum. You reuse this model in the tutorial "Analyze a Simple Pendulum" on page 1-18.

See Also

Revolute Joint | Prismatic Joint

More About

- "Analyze a Simple Pendulum" on page 1-18
- "Model a Simple Link" on page 1-10

Analyze a Simple Pendulum

This example shows how to add forces and torques to a multibody model. You can use blocks that have motion-sensing capabilities to analyze the dynamic response of the multibody model. For more information about multibody dynamics simulations, see "Multibody Dynamics". The end result is a set of time-domain and phase plots, one for each combination of forces and torques. You create these plots by using MATLAB commands with Simscape Multibody motion outputs as arguments.
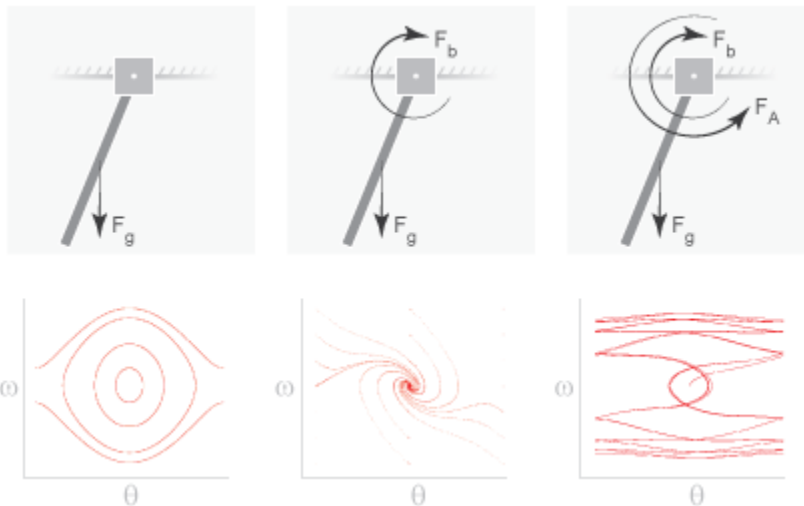
First, create the simple pendulum model in "Model a Simple Pendulum" on page 1-15. By adding forces and torques to this model, you incrementally change the pendulum from undamped and free to damped and driven. The forces and torques that you apply include:

- Gravitational force (F_g) — Global force that acts on every body in direct proportion to its mass,which you specify in terms of the acceleration vector g. You specify this vector using the Mechanism Configuration block.

- Joint damping (F_b) — Internal torque between the pendulum and the joint fixture which you parameterize in terms of a linear damping coefficient. You specify this parameter using the Revolute Joint block that connects the pendulum to the joint fixture.

- Actuation torque (F_A) — Driving torque between the pendulum and the joint fixture, which you specify directly as a Simscape physical signal using the Revolute Joint block that connects the pendulum to the joint fixture.

Output Sensing Information for the Pendulum Motion

First, update the `simple_pendulum` model to output position and velocity data, then output this data to the MATLAB base workspace.

1. Open the `simple_pendulum` model that you created in tutorial "Model a Simple Pendulum" on page 1-15.

2. In the **Z Revolute Primitive (Rz) > Sensing** section of the Revolute Joint block dialog box, select the following parameters:

    *   **Position**
    *   **Velocity**

    The block exposes two additional physical signal ports, **q** and **w**, that output the angular position and velocity of the pendulum with respect to the world frame.

3. Add these blocks to the model. You use the blocks to output the joint position and velocity to the MATLAB base workspace.

Library	Block	Quantity
Simscape > Utilities	PS-Simulink Converter	2
Simulink > Sinks	To Workspace	2

4. Change the **Variable name** parameters in the To Workspace block dialog boxes to `q` and `w`. These variables make it easy to identify the joint variables that the To Workspace blocks output during simulation—position, through the Revolute Joint block port **q**, and velocity, through the Revolute Joint block port **w**. To remove the keyword, `out`, from the variable names, on the **Modeling** tab of the Simulink Toolstrip, click **Model Settings**, then select **Data Import/Export** and clear the **Single simulation output** parameter.

5. Connect the blocks as shown in the figure. Ensure that the To Workspace block with the variable name `q` connects, through the PS-Simulink Converter block, to the Revolute Joint block port **q**, and that the To Workspace block with variable name `w` connects to the Revolute Joint block port **w**.
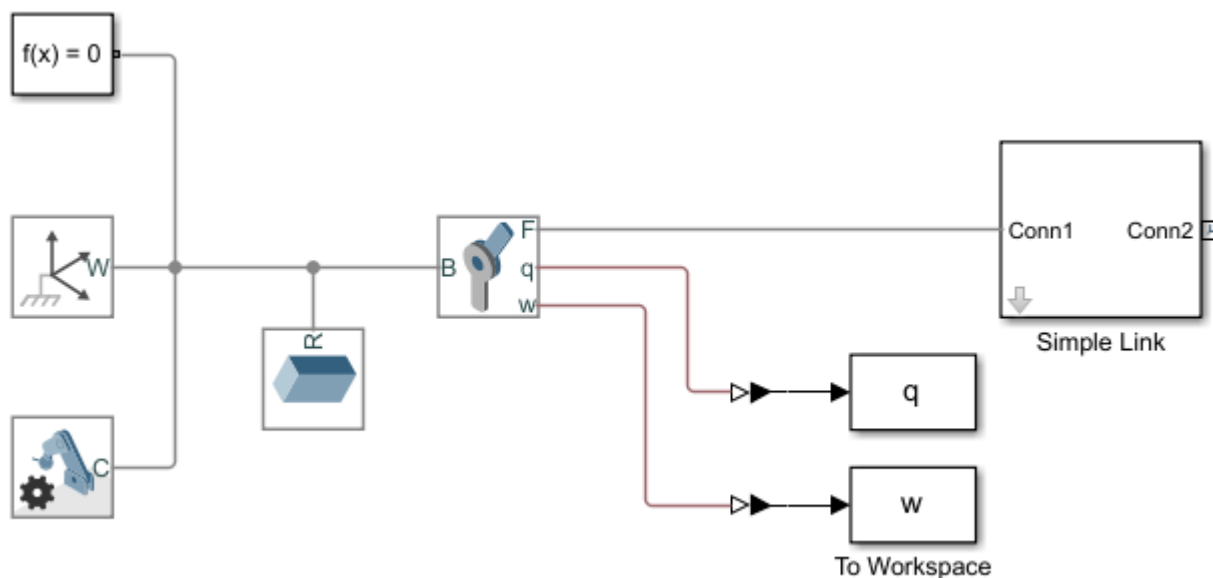
6. Change the value of the **Vector format** parameter to `1-D array` for both PS-Simulink Converter blocks.

7   Save the model under a different name, such as `simple_pendulum_analysis`.

Analyze Undamped Pendulum

1   Run the simulation. Mechanics Explorer opens with a 3-D animation of the simple pendulum model.

2   Plot the joint position and velocity with respect to time by entering the following code at the MATLAB command prompt:

```
figure;
hold on;
plot(q);
plot(w);
```
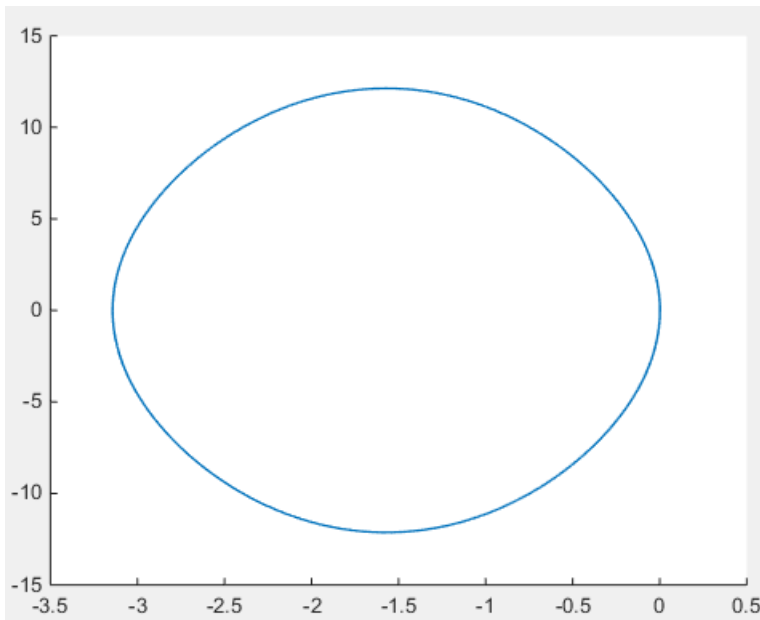
The figure shows the resulting plot.
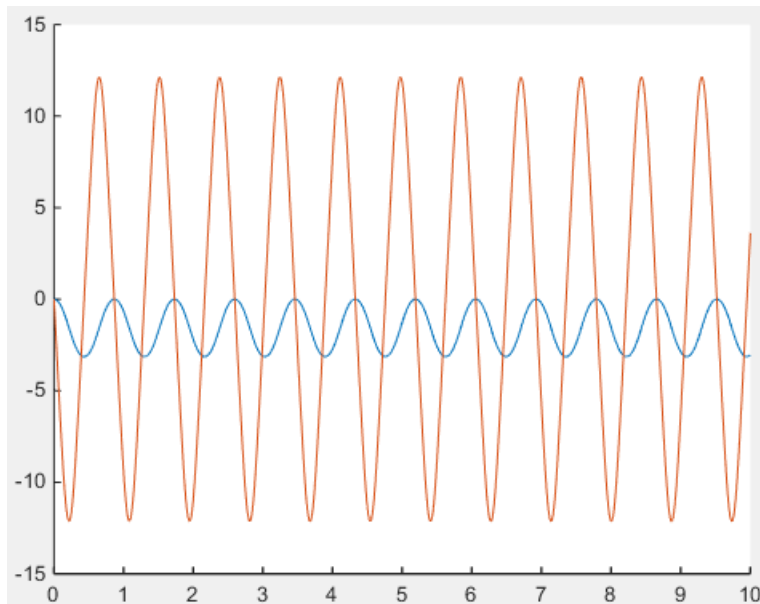
3   Plot the joint angular velocity with respect to the angular position by entering this code at the MATLAB command prompt.

```
figure;
plot(q.data, w.data);
```

The result is the phase plot of the joint that corresponds to a starting position of zero degrees with respect to the horizontal plane.

Try simulating the model using different starting angles. In the **Z Revolute Primitive (Rz) >
State Targets** section, under **Specify Position Target**, specify the **Value** parameter of the
Revolute Joint block dialog box. This figure shows a compound phase plot for starting angles of
-80, -40, 0, 40, and 80 degrees.
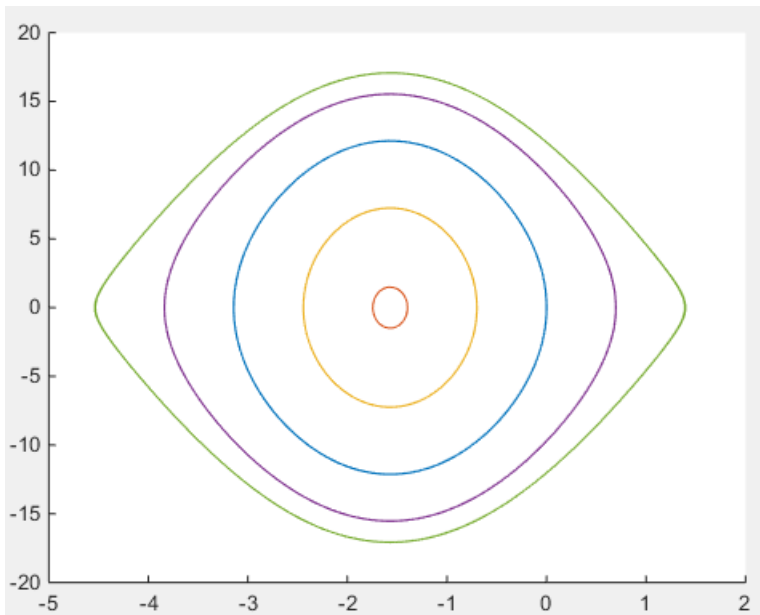
Analyze Damped Pendulum

1   In the Revolute Joint block dialog box, set **Z Revolute Primitive (Rz) > Internal Mechanics >
Damping Coefficient** to 8e-5 (N*m)/(deg/s). The damping coefficient causes energy dissipation
during motion which results in a gradual decay of the pendulum oscillation amplitude.

2 Ensure that **Z Revolute Primitive (Rz) > State Targets > Specify Position Target > Value** is set to 0 deg.

3 Run the simulation.

4 Plot the joint position and velocity with respect to time. To do this, enter this code at the MATLAB command prompt.

```
figure;
hold on;
plot(q);
plot(w);
```

The figure shows the resulting plot. Note that the pendulum oscillations decay with time due to damping. At larger damping values, the pendulum becomes overdamped and the oscillations disappear.
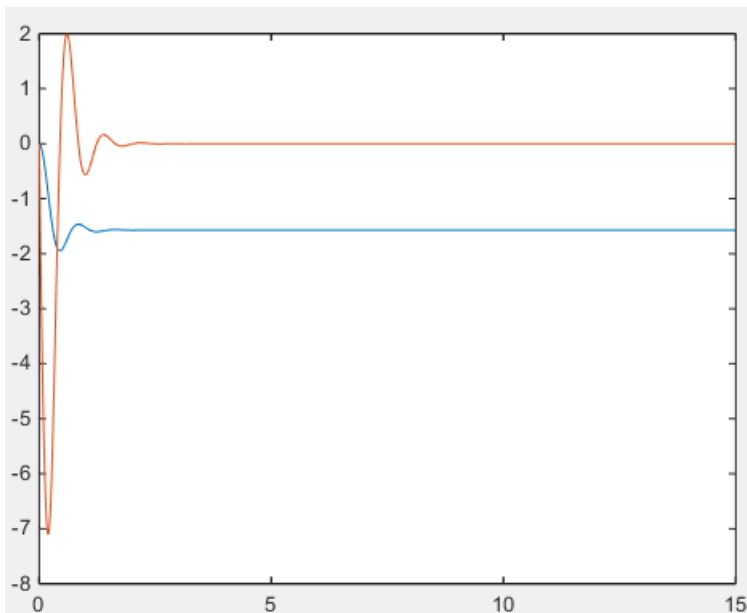

5 Plot the joint phase plot. To do this, enter this code at the MATLAB command prompt.

```
figure;
plot(q.data, w.data);
```

The figure shows the resulting plot.
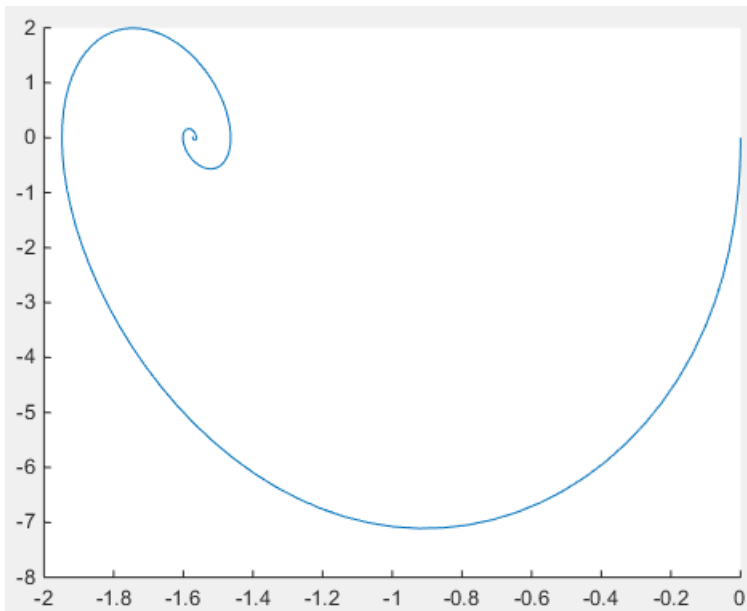
Try simulating the model using different starting angles. In the **Z Revolute Primitive (Rz) >
State Targets** section, under **Specify Position Target**, specify **Value** parameter of the Revolute
Joint block dialog box. The figure shows a compound phase plot for starting angles of `-240`,
`-180`, `-120`, `-60`, `0`, and `60` degrees.
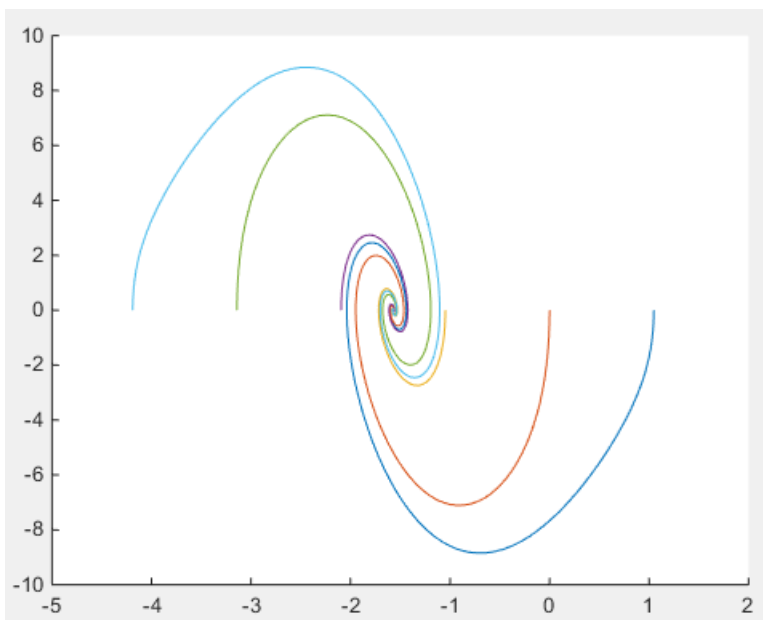

Analyze Damped and Driven Pendulum

1 In the Revolute Joint block dialog box, in the **Actuation** section, set the **Torque** parameter to
`Provided by Input`. The block exposes a physical signal input port, **t**, that you can use to
specify the joint actuation torque.

2 Add these blocks to the model.

Library	Block
Simscape > Utilities	Simulink-PS Converter
Simulink > Sources	Sine Wave

The Sine Wave block provides a periodic torque input as a Simulink signal. The Simulink-PS Converter block converts the Simulink signal to a physical signal compatible with Simscape Multibody blocks.

3 Connect the blocks as shown in the figure.
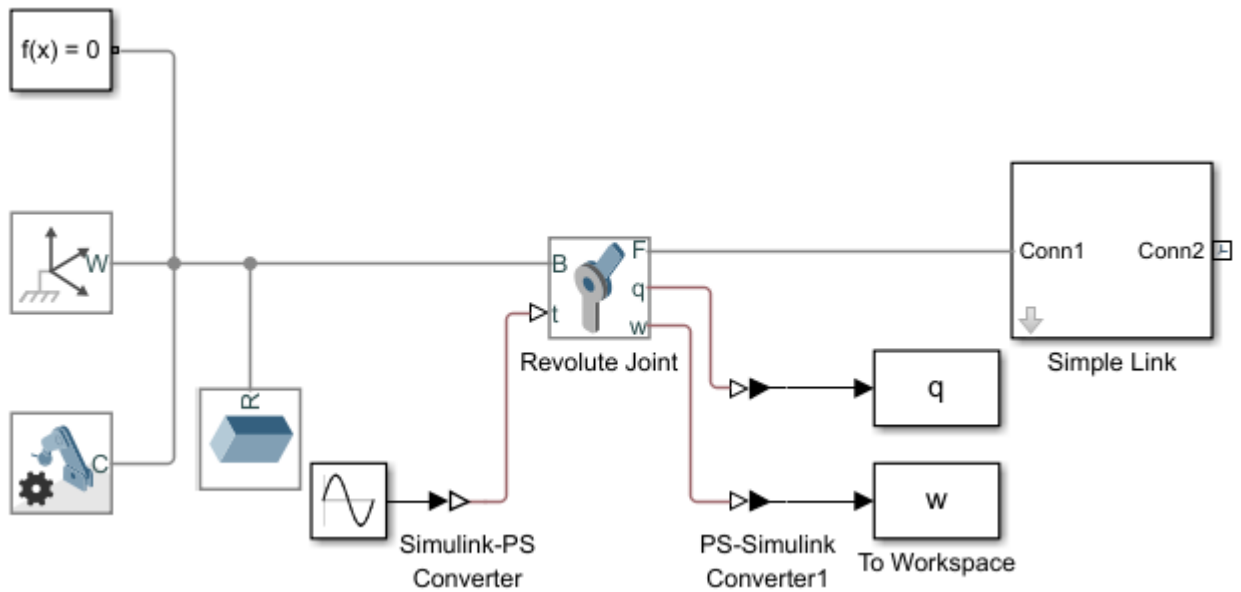
4 In the Sine Wave block dialog box, set the **Amplitude** parameter to `0.06`. This amplitude corresponds to an actuation torque that oscillates between `-0.06` N and `0.06` N.

5 In the Revolute Joint block dialog box, in the **Z Revolute Primitive (Rz) > State Targets** section, under **Specify Position Target**, set **Value** parameter to `0` deg.

6 Run the simulation.

7 Plot the joint position and velocity with respect to time. To do this, enter this code at the MATLAB command prompt.
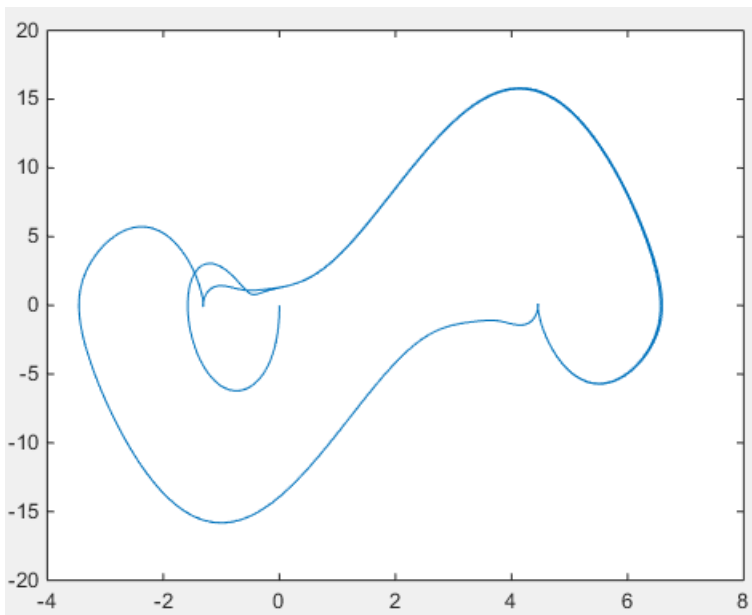
```
figure;
hold on;
plot(q);
plot(w);
```

The figure shows the resulting plot.
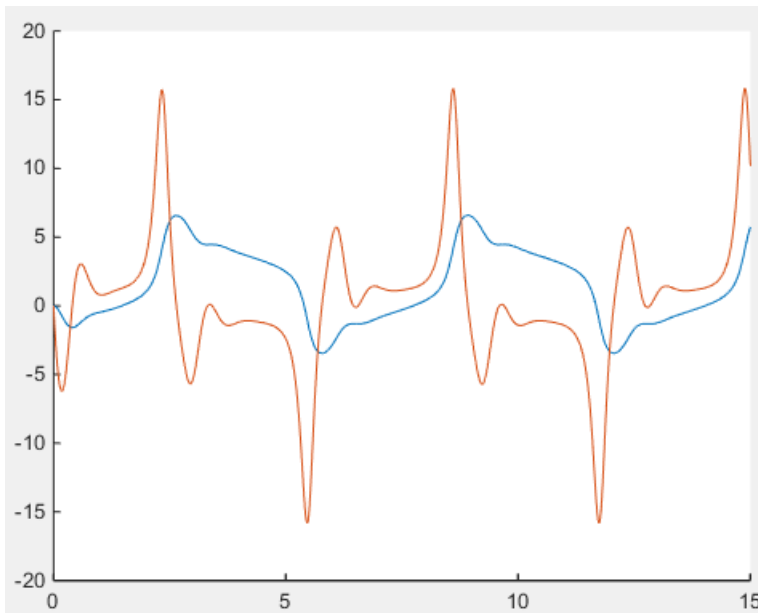
8 Plot the joint phase plot. To do this, enter this code at the MATLAB command prompt:

```
figure;
plot(q.data, w.data);
```

The figure shows the resulting plot.

See Also
PS-Simulink Converter | Revolute Joint | To Workspace

More About

- "Force and Torque Sensing"
- "Model a Simple Pendulum" on page 1-15