

**SimMechanics™**  
Getting Started Guide



**MATLAB® & SIMULINK®**

R2015b



## How to Contact MathWorks



Latest news: [www.mathworks.com](http://www.mathworks.com)  
Sales and services: [www.mathworks.com/sales\\_and\\_services](http://www.mathworks.com/sales_and_services)  
User community: [www.mathworks.com/matlabcentral](http://www.mathworks.com/matlabcentral)  
Technical support: [www.mathworks.com/support/contact\\_us](http://www.mathworks.com/support/contact_us)



Phone: 508-647-7000



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

*SimMechanics™ Getting Started Guide*

© COPYRIGHT 2002–2015 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

**FEDERAL ACQUISITION:** This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

### Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [www.mathworks.com/trademarks](http://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](http://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)

## 1

### Introduction to SimMechanics Software

<b>SimMechanics Product Description</b> .....	<b>1-2</b>
Key Features .....	1-2
<b>Required and Related Products</b> .....	<b>1-3</b>
SimMechanics Visualization Requirements .....	1-3
Support for SimMechanics Animations .....	1-3
Related Products .....	1-3
<b>Start New Multibody Model</b> .....	<b>1-5</b>
<b>Multibody Model Anatomy</b> .....	<b>1-6</b>
Basic Model Components .....	1-6
Model Actuation .....	1-9
Dynamical Sensing .....	1-11
<b>Model Simple Link</b> .....	<b>1-14</b>
Model Overview .....	1-14
Build Model .....	1-14
Generate Subsystem .....	1-16
Visualize Model .....	1-17
Save Custom Block .....	1-18
<b>Model Simple Pendulum</b> .....	<b>1-19</b>
Model Overview .....	1-19
Build Model .....	1-20
Specify Gravity .....	1-21
Set Pendulum Starting Position .....	1-21
Configure Solver .....	1-21
Assemble Model .....	1-21
Simulate Model .....	1-22
Save Model .....	1-22

<b>Analyze Simple Pendulum</b> .....	<b>1-23</b>
Overview .....	<b>1-23</b>
Sense Pendulum Motion .....	<b>1-24</b>
Analyze Undamped Pendulum .....	<b>1-25</b>
Analyze Damped Pendulum .....	<b>1-28</b>
Analyze Damped and Driven Pendulum .....	<b>1-31</b>
 <b>SimMechanics First and Second Generation Comparison</b> .	 <b>1-35</b>

# Introduction to SimMechanics Software

---

- “SimMechanics Product Description” on page 1-2
- “Required and Related Products” on page 1-3
- “Start New Multibody Model” on page 1-5
- “Multibody Model Anatomy” on page 1-6
- “Model Simple Link” on page 1-14
- “Model Simple Pendulum” on page 1-19
- “Analyze Simple Pendulum” on page 1-23
- “SimMechanics First and Second Generation Comparison” on page 1-35

## SimMechanics Product Description

### Model and simulate multibody mechanical systems

SimMechanics provides a multibody simulation environment for 3D mechanical systems, such as robots, vehicle suspensions, construction equipment, and aircraft landing gear. You model the multibody system using blocks representing bodies, joints, constraints, and force elements, and then SimMechanics formulates and solves the equations of motion for the complete mechanical system. Models from CAD systems, including mass, inertia, joint, constraint, and 3D geometry, can be imported into SimMechanics. An automatically generated 3D animation lets you visualize the system dynamics.

You can parameterize your models using MATLAB<sup>®</sup> variables and expressions, and design control systems for your multibody system in Simulink<sup>®</sup>. You can add electrical, hydraulic, pneumatic, and other components to your mechanical model using Simscape<sup>™</sup> and test them all in a single simulation environment. To deploy your models to other simulation environments, including hardware-in-the-loop (HIL) systems, SimMechanics supports C-code generation (with Simulink Coder<sup>™</sup>).

### Key Features

- Blocks and modeling constructs for simulating and analyzing 3D mechanical systems in Simulink
- Rigid body definition using standard geometry and custom extrusions defined in MATLAB
- Automatic calculation of mass and inertia tensor
- Simulation modes for analyzing motion and calculating forces
- Visualization and animation of multibody system dynamics with 3D geometry
- SimMechanics Link utility, providing an interface to Pro/ENGINEER<sup>®</sup>, SolidWorks<sup>®</sup>, and Autodesk Inventor, and an API for interfacing with other CAD platforms
- Support for C-code generation (with Simulink Coder)

## Required and Related Products

### In this section...

“SimMechanics Visualization Requirements” on page 1-3

“Support for SimMechanics Animations” on page 1-3

“Related Products” on page 1-3

### SimMechanics Visualization Requirements

SimMechanics visualization requires Silicon Graphics OpenGL<sup>®</sup> graphics support on your system to display and animate SimMechanics models.

You can improve your speed and graphics resolution by adding a graphics accelerator hardware card to your system. Animation of simulations is sensitive to central processor and graphics card speed and memory. Experiment with graphics hardware and system settings to find a reasonable compromise between quality and speed for your system.

### Support for SimMechanics Animations

Mechanics Explorer, the SimMechanics visualization utility, enables you to record an animation of your model. The resulting animation video is in a compressed AVI format encoded using the M-JPEG codec. You can play back an animation video using the MATLAB function `implay` or an external AVI media player.

### Related Products

You can extend the capability of SimMechanics using other physical modeling products found in the Simscape family. Each physical modeling product gives you a set of block libraries with which you can model common components found in industry and academia: rigid bodies, gears, valves, solenoids, etc.

With the physical modeling products, you can model not only mechanical systems, but also electrical, hydraulic, and power systems. You can model each system separately, and then integrate the systems into a single multiphysics model where you can analyze combined system performance.

### Physical Modeling Product Family

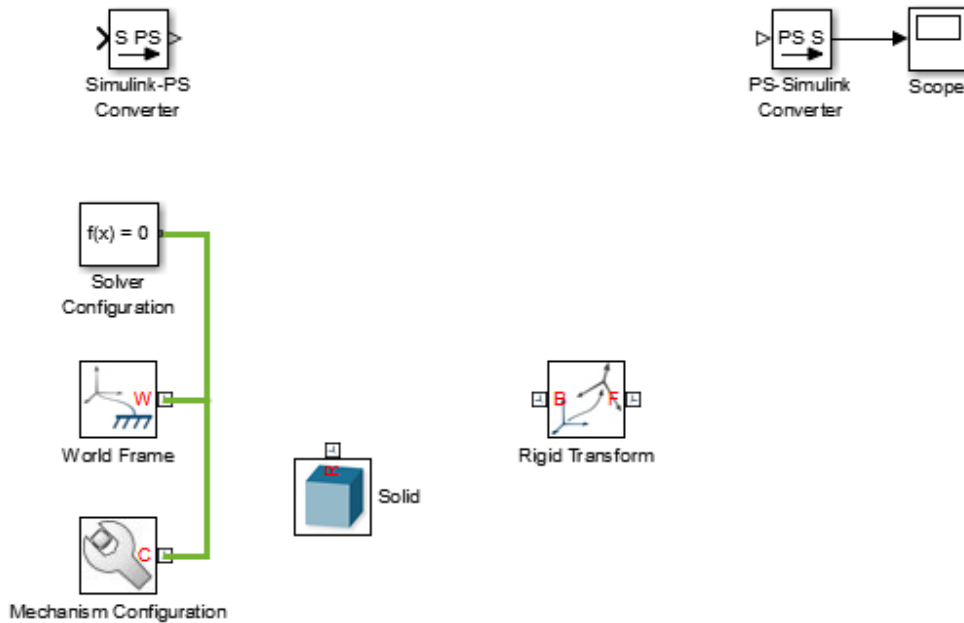
The physical modeling family includes five products:

- SimDriveline™, for modeling and simulating drivetrain systems
- SimElectronics®, for modeling and simulating electronic systems
- SimHydraulics®, for modeling and simulating hydraulic systems
- SimMechanics, for modeling and simulating three-dimensional mechanical systems
- SimPowerSystems™, for modeling and simulating electrical power systems



## Start New Multibody Model

You can start a new multibody model from the MATLAB command line. To do this, enter `smnew`. A new model window opens with commonly used blocks. The SimMechanics block library also opens. The figure shows the new model window that you see. Drag blocks from the library into the model window to begin modeling a new multibody system.



## Multibody Model Anatomy

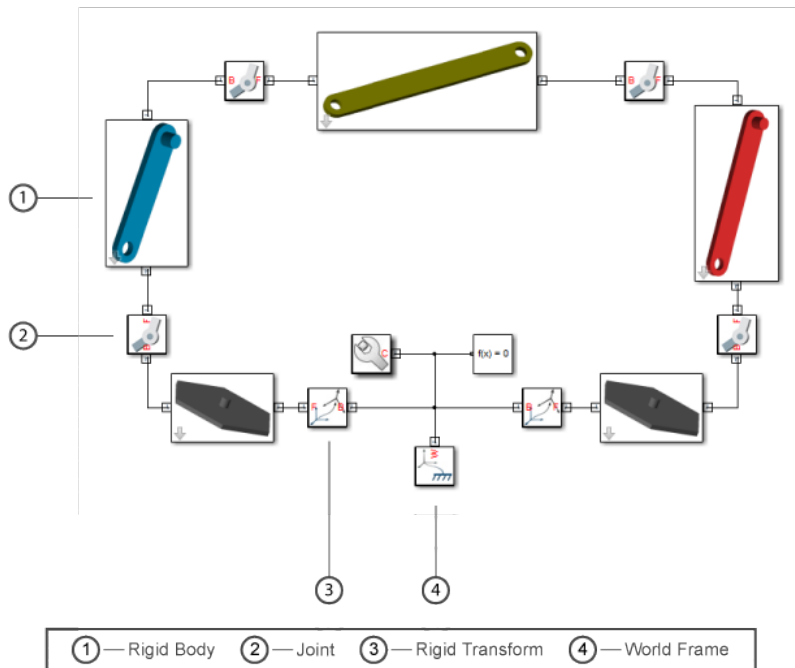
In this section...
“Basic Model Components” on page 1-6
“Model Actuation” on page 1-9
“Dynamical Sensing” on page 1-11

With SimMechanics, you represent a multibody system using blocks. Like all physical modeling products, each block represents a physical component or an abstract entity fundamental to physical modeling, e.g. frames and frame transforms.

By connecting the blocks with connection lines, you define the relationships that unite the physical components into a single system (or subsystem). In a basic model, these physical components include rigid bodies and joints. You can also add forces and torques, motion sensors, and kinematic constraints, e.g., to represent gears.

### Basic Model Components

The figure shows the block diagram of a multibody system—the four-bar linkage. This model contains subsystem blocks to represent the links and pivot mounts. These represent the rigid bodies of the model. The model contains also four Revolute Joint blocks. These represent the joints in the model. Combined, these blocks form the foundation of this model.

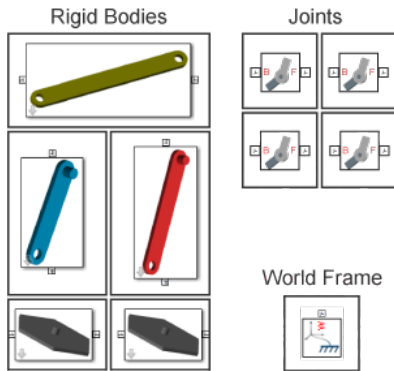


While important, rigid body subsystem and joint blocks are not sufficient to represent the four-bar linkage. Other blocks serve important purposes. These include World Frame, Rigid Transform, Mechanism Configuration, and Solver Configuration blocks. The table summarizes their functions in a multibody model.

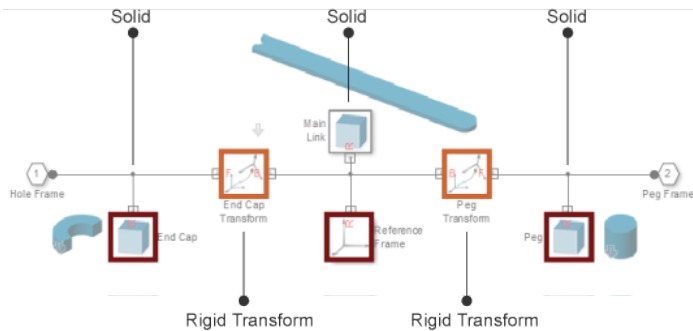
Block	Function
World Frame	Provides the ultimate reference frame in a model. All remaining frames are defined with respect to this frame. It is inertial and it defines absolute rest.
Rigid Transform	Applies a fixed spatial relationship between frames. This block defines the offset distance and angle between two frames.
Mechanism Configuration	Identifies the gravity vector in a model.

Block	Function
Solver Configuration	Provides essential simulation parameters required to simulate the model.

The figure breaks the four-bar model into its logical components. These are the physical components and abstract entities that you need in order to represent this system.



Each rigid body subsystem contains SimMechanics blocks that represent solids and their spatial relationships. The blocks are **Solid** and **Rigid Transform**. The figure shows the blocks that model one of the binary links. Three **Solid** blocks represent the three solid sections of this rigid body—main, peg, and hole sections. Two **Rigid Transform** blocks represent the fixed spatial relationships between the three solids. You use them to position the peg and hole sections at the ends of the main section.

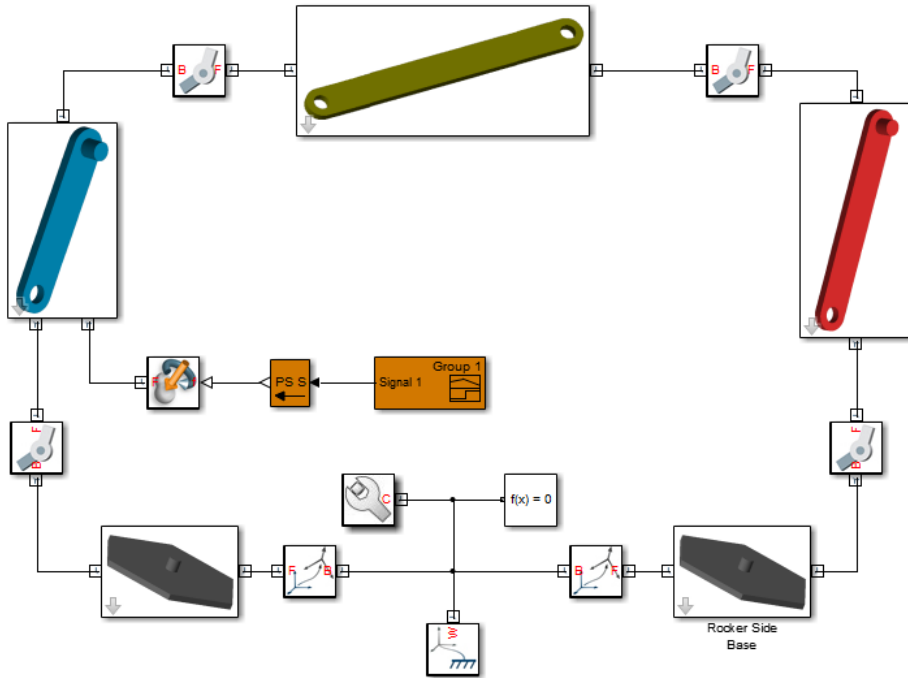


## Model Actuation

You can actuate a model by applying a force or torque to a rigid body or to a joint. To represent forces and torques acting on a rigid body, SimMechanics provides a Forces and Torques library. Drag a block from this library and connect it to the rigid 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 rigid 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.

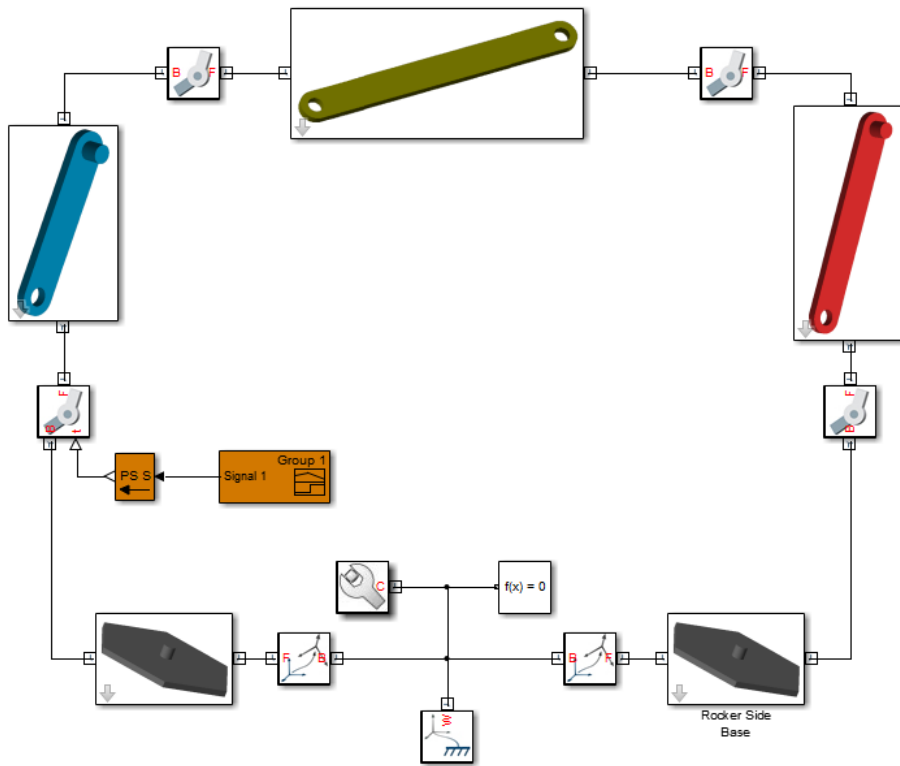


To specify the force or torque acting at a joint, SimMechanics provides a selection of actuation inputs directly in the joint blocks. Each joint primitive—the basic component of a joint block—provides a selection of actuation inputs specific to that primitive.

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 at 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:

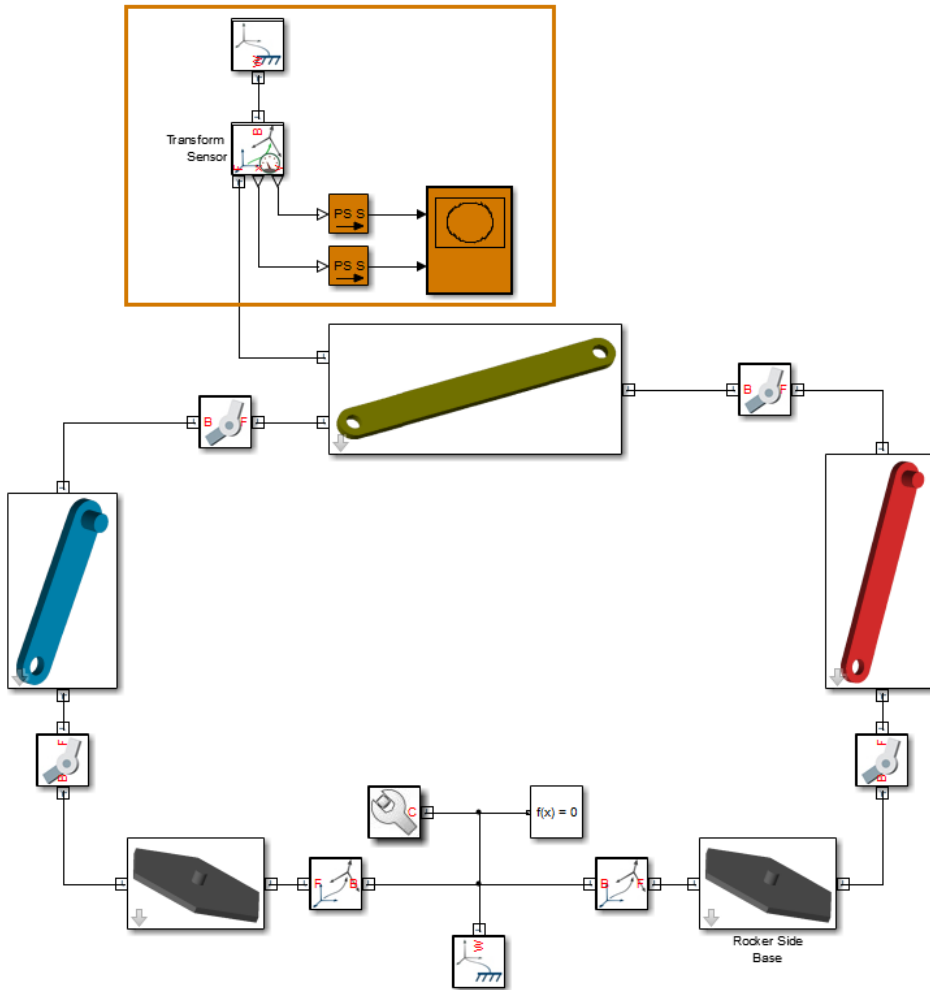
- Motion — Compute and output the relative position, velocity, or acceleration between two SimMechanics frames. You can sense motion between joint frames, by using the sensing capability of joint blocks, or between arbitrary frames, by using the Transform Sensor block.
- Force or torque — Compute and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





## Related Examples

- “Model Simple Link” on page 1-14
- “Model Simple Pendulum” on page 1-19

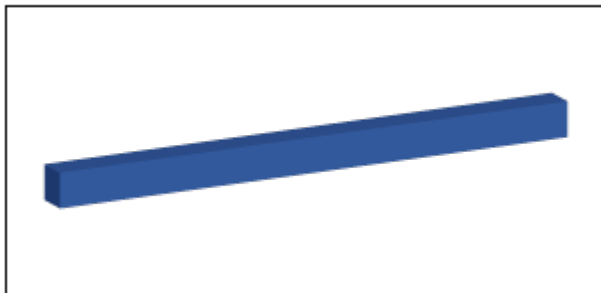
## Model Simple Link

### In this section...

- “Model Overview” on page 1-14
- “Build Model” on page 1-14
- “Generate Subsystem” on page 1-16
- “Visualize Model” on page 1-17
- “Save Custom Block” on page 1-18

### Model Overview

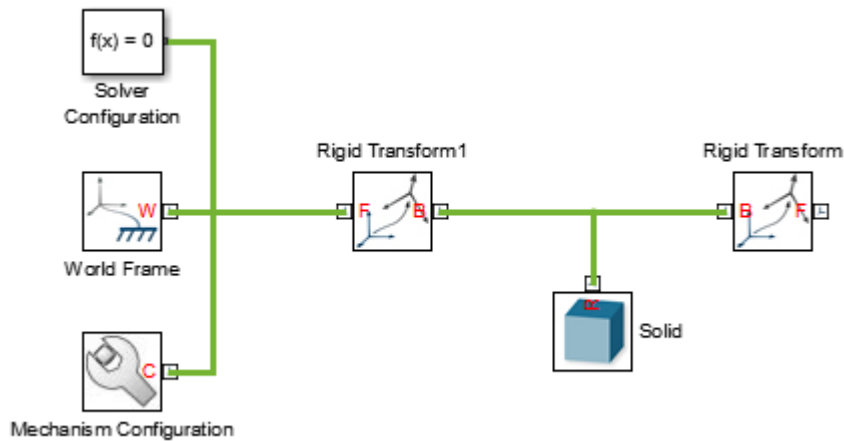
Mechanical links are common building blocks in linkages, mechanisms, and machines. The simple pendulum is an example with one link. In this tutorial, you model a simple link with two end frames that you can later connect to joints. Rigid Transform blocks provide the end frames, while a Solid block provides geometry, inertia, and color. For simplicity, the model assumes the link has a brick shape.



### Build Model

- 1 At the MATLAB command line, enter `smnew`. The SimMechanics block library and a model template with commonly used blocks open up.
- 2 Make a copy of the Rigid Transform block and paste it in the model. The Rigid Transform blocks enable you to create new frames to which you can connect joints during multibody assembly.

- 3 Delete the blocks Simulink-PS Converter, PS-Simulink Converter, and Scope. You do not need these blocks in this tutorial.
- 4 Connect the remaining blocks as shown in the figure. Ensure that the base frame ports (B) of the Rigid Transform blocks both face the Solid block frame port. Since each Rigid Transform block applies a spatial transformation with respect to its base frame, switching port connections generally changes the spatial relationship between the two frames.



- 5 In the Solid block dialog box, specify the following parameters. Later, you define the MATLAB variables shown using a Subsystem block that contains the Solid and Rigid Transform blocks. Among its advantages, this approach enables you to update variables used in multiple blocks from a single place—the Subsystem block dialog box.

Parameter	Value	Units
<b>Geometry &gt; Dimensions</b>	[ L W H ]	Change to cm
<b>Inertia &gt; Density</b>	rho	Default units
<b>Graphic &gt; Visual Properties &gt; Color</b>	rgb	Not applicable

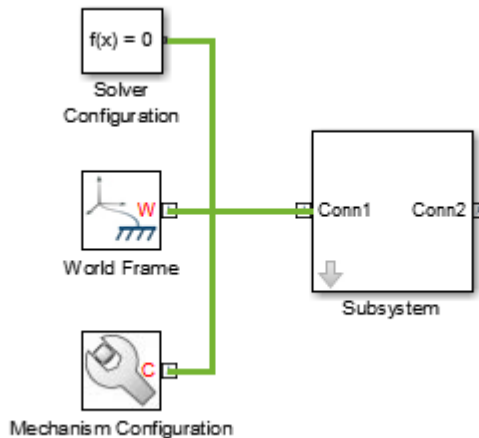
- 6 In the dialog boxes of the Rigid Transform and Rigid Transform1 blocks, specify the following parameters. These parameters encode the offset between the base and

follower port frames of the Rigid Transform blocks, located at the link ends, with respect to the Solid reference port frame.

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	Change to cm

## Generate Subsystem

- 1 Select the Solid block and the two Rigid Transform blocks.
- 2 Right-click the highlighted region and select **Create Subsystem from Selection**. Simulink adds a new Subsystem block containing the Solid and Rigid Transform blocks. At the end of the tutorial, this will be a custom block representing the simple link rigid body.



- 3 Right-click the Subsystem block, and select **Mask > Create Mask**. A mask editor opens up, enabling you to specify the numerical values of the MATLAB variables you entered in the Solid and Rigid Transform block dialog boxes.

- 4 In the **Parameters & Dialog** tab of the Mask Editor window, add five edit fields



to the **Parameters** folder. You can find this folder in the **Dialog box** pane. In the edit fields, specify the following parameters and click **OK**. **Prompt** is the desired text for each parameter in the Subsystem block dialog box. **Name** is the MATLAB variable associated with each Subsystem block parameter.

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

- 5 Double-click the Subsystem block dialog box and enter the following numerical values. These are the values of the MATLAB variables that you entered in the Solid and Rigid Transform block dialog boxes.

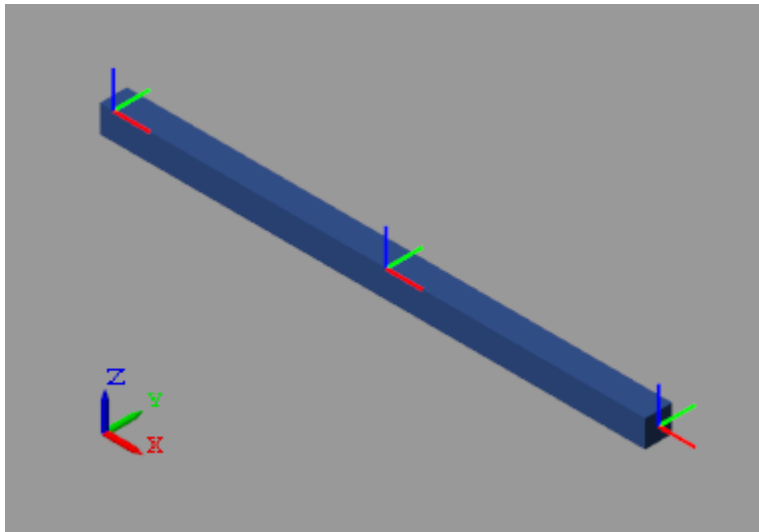
Parameter	Value
Length (cm)	20
Width (cm)	1
Thickness (cm)	1
Density (kg/m <sup>3</sup> )	2700
Color [R G B]	[0.25 0.40 0.70]

## Visualize Model

Update the block diagram. You can do this by selecting, in the Simulink menu bar, **Simulation > Update Diagram**. Mechanics Explorer opens with a front view of the simple link model. In the Mechanics Explorer toolstrip, select the isometric view button

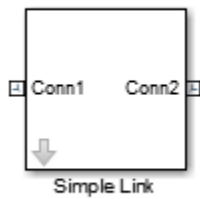


to obtain the 3-D view shown below. To view the frames present in the model—including those you created using the Rigid Transform blocks—select **View > Show Frames** in the Mechanics Explorer menu bar.



## Save Custom Block

Rename the Subsystem block Simple Link and save it in a custom block library. You reuse this block in tutorial “Model Simple Pendulum” on page 1-19.



## Simple Link Custom Block

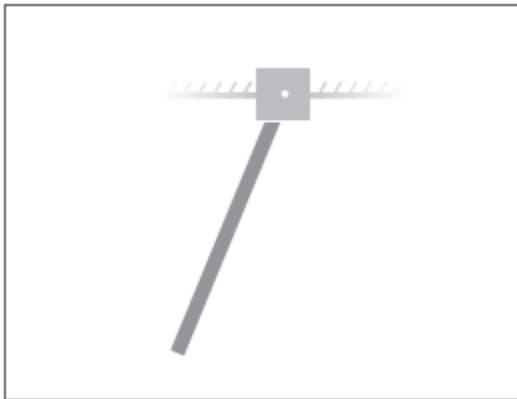
# Model Simple Pendulum

## In this section...

“Model Overview” on page 1-19  
“Build Model” on page 1-20  
“Specify Gravity” on page 1-21  
“Set Pendulum Starting Position” on page 1-21  
“Configure Solver” on page 1-21  
“Assemble Model” on page 1-21  
“Simulate Model” on page 1-22  
“Save Model” on page 1-22

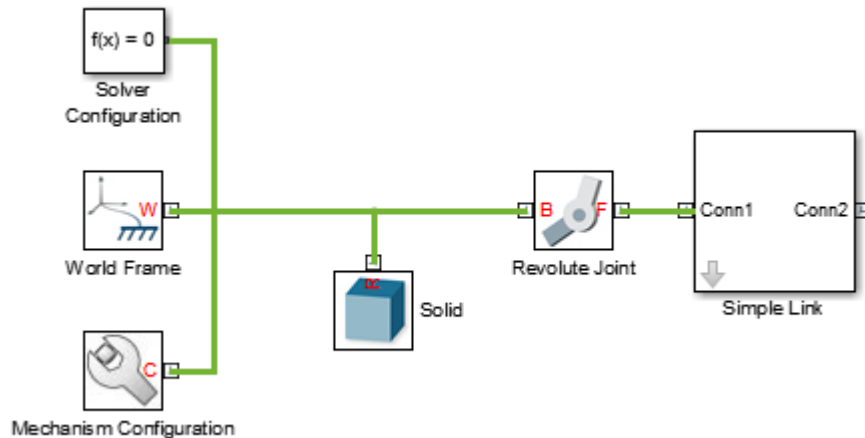
## Model Overview

The pendulum is the simplest mechanical system you can model. This system contains two rigid bodies, a link and a fixed pivot, connected by a revolute joint. In this tutorial, you model and simulate a pendulum using the custom link block you created in “Model Simple Link” on page 1-14. A Revolute Joint block provides the rotational degree of freedom between the link and the world frame.



## Build Model

- 1 At the MATLAB command prompt, enter `smnew`. The SimMechanics block library and a model template with commonly used blocks open up.
- 2 Delete blocks Simulink-PS Converter, PS-Simulink Converter, Scope, and Rigid Transform. You do not need them in this tutorial.
- 3 Drag the Simple Link custom block you created in the tutorial “Model Simple Link” on page 1-14 into the model.
- 4 Drag a Revolute Joint block into the model. You can find this block in the **SimMechanics Second Generation > Joints** library. This block provides one rotational degree of freedom between its port frames.
- 5 Connect the blocks as shown in the figure. 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 Solid block dialog box, specify the following parameters. This block connects rigidly to the World frame and therefore has no effect on model dynamics. 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]	Not applicable

## 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 the **Uniform Gravity > Gravity** parameter 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.

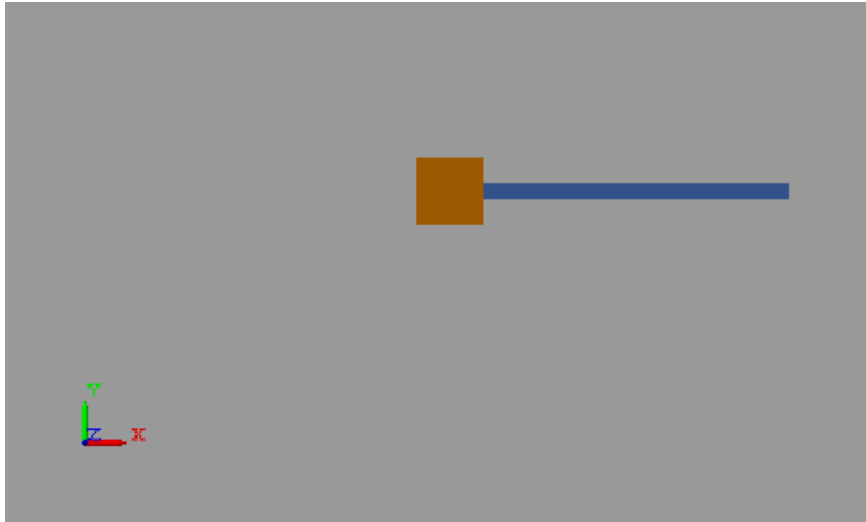
## Configure Solver

- 1 In the Simulink Editor menu bar, select **Simulation > Model Configuration Parameters**.
- 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. You can do this in the Simulink Editor menu bar, by selecting **Simulation > Update diagram**. 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. You can do this through the Simulink Editor menu bar, by selecting **Simulation > Run**. 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 Simple Pendulum” on page 1-23.

## Analyze Simple Pendulum

### In this section...

“Overview” on page 1-23

“Sense Pendulum Motion” on page 1-24

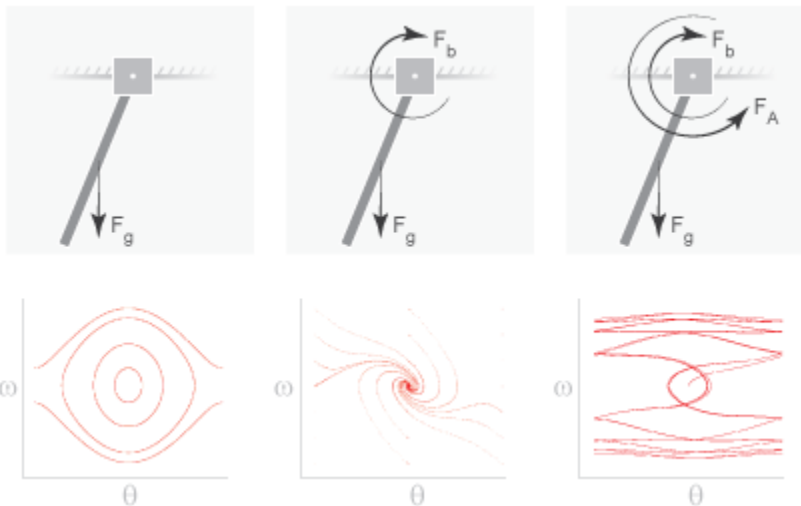
“Analyze Undamped Pendulum” on page 1-25

“Analyze Damped Pendulum” on page 1-28

“Analyze Damped and Driven Pendulum” on page 1-31

### Overview

In this tutorial, you explore the various forces and torques that you can add to a model. Then, using blocks with motion sensing capability, you analyze the resulting dynamic response of the model. The end result is a set of time-domain and phase plots, one for each combination of forces and torques. You create these plots using MATLAB commands with SimMechanics motion outputs as arguments.



Your starting point is the simple pendulum model that you built in “Model Simple Pendulum” on page 1-19. 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, acting on every rigid body in direct proportion to its mass, that 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, that 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, that you prescribe directly as a Simscape physical signal. You prescribe this signal using the Revolute Joint block that connects the pendulum to the joint fixture.

## Sense Pendulum Motion

- 1 Open the `simple_pendulum` model that you created in tutorial “Model Simple Pendulum” on page 1-19.
- 2 In the Sensing menu of the Revolute Joint block dialog box, select the following variables:
  - **Position**
  - **Velocity**

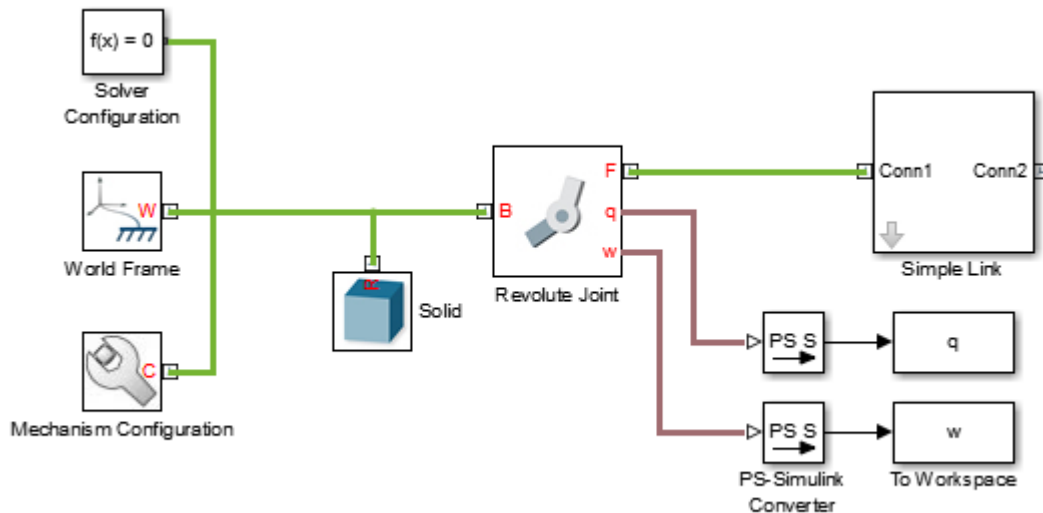
The block exposes two additional physical signal ports, labeled  $q$  and  $w$ , that output the angular position and velocity of the pendulum with respect to the world frame.

- 3 Drag the following blocks into the model. You use them 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  $w$ .

- Connect the blocks as shown in the figure. Ensure that the To Workspace block with 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  $w$ .

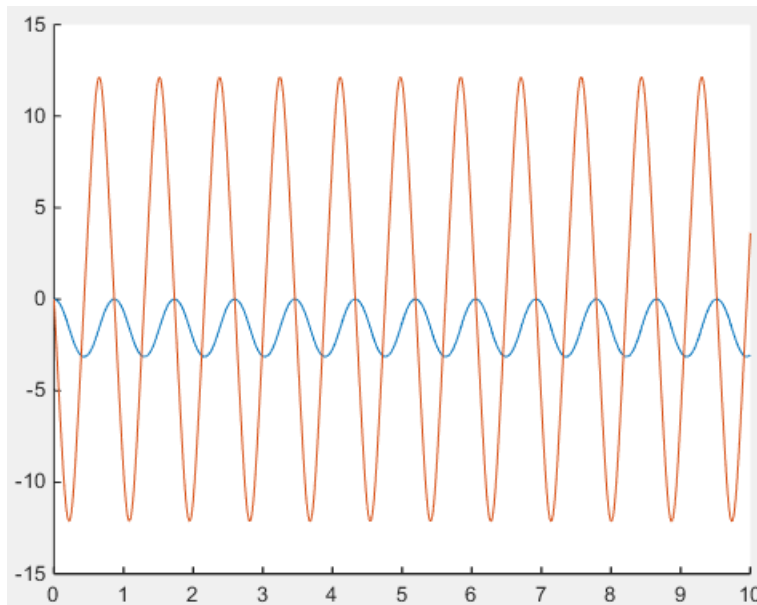


- Save the model under a different name, e.g., `simple_pendulum_analysis`, in a convenient folder.

## Analyze Undamped Pendulum

- Run the simulation. You can do this in the SimMechanics Editor menu bar by selecting **Simulation > Run**. Mechanics Explorer opens with a 3-D animation of the simple pendulum model.
- Plot the joint position and velocity with respect to time, e.g., by entering the following code at the MATLAB command prompt:

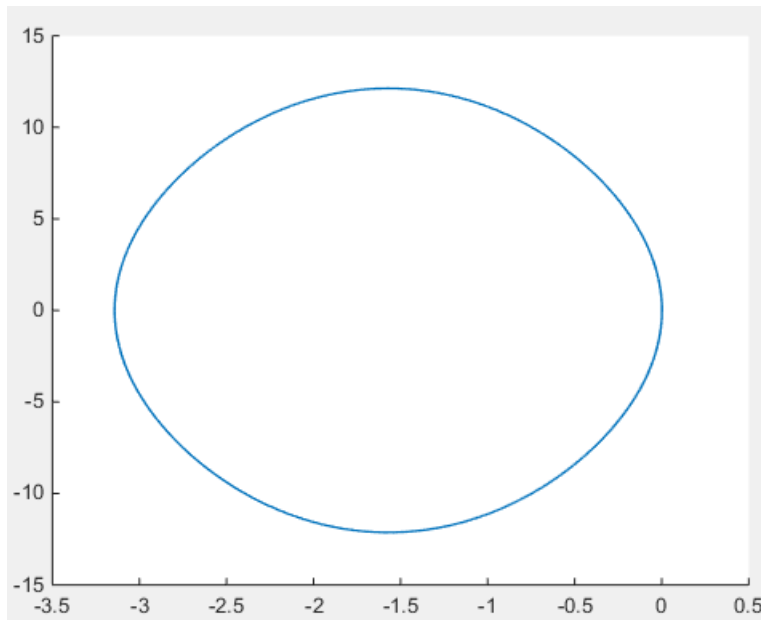
```
figure; % Open a new figure
hold on;
plot(q); % Plot the pendulum angle
plot(w); % Plot the pendulum angular velocity
The figure shows the resulting plot.
```



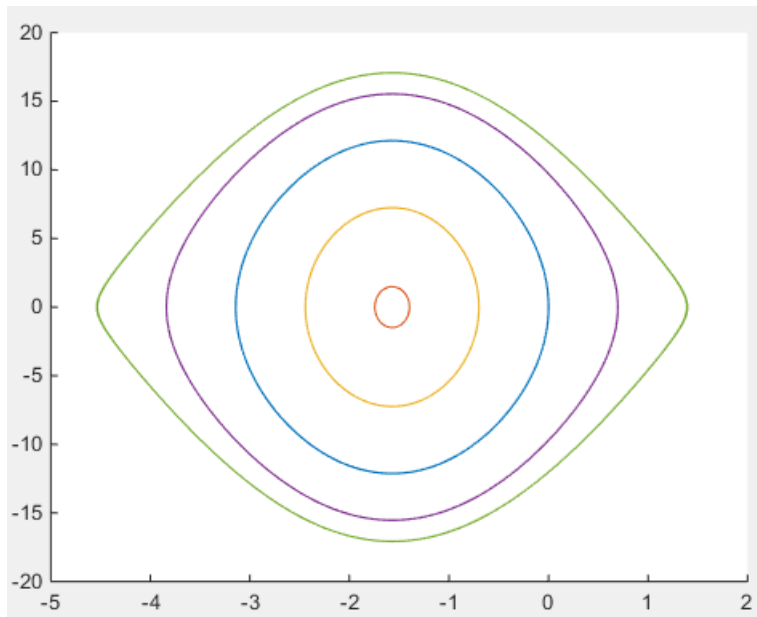
- 3 Plot the joint angular velocity with respect to the angular position, e.g., by entering the following code at the MATLAB command prompt.

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

The result, shown in the figure, is the phase plot of the joint corresponding to a starting position of zero degrees with respect to the horizontal plane.



Try simulating the model using different starting angles. You can change the starting angle in the **State Targets > Position** menu of the Revolute Joint block dialog box. The figure shows a compound phase plot for starting angles of -80, -40, 0, 40, and 80 degrees.



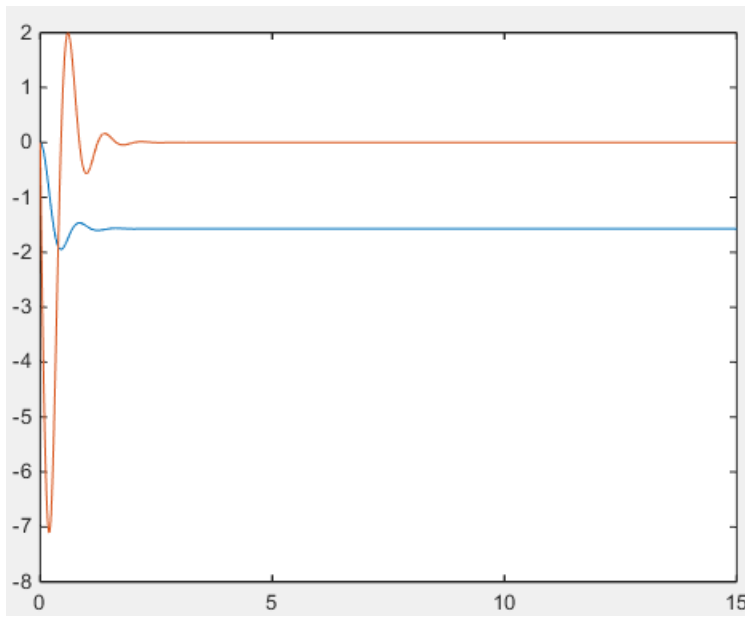
## Analyze Damped Pendulum

- 1 In the Revolute Joint block dialog box, set **Internal Mechanics > Damping** to  $8e-5$  (N\*m)/(deg/s). The damping coefficient causes energy dissipation during motion, resulting in a gradual decay of the pendulum oscillation amplitude.
- 2 Ensure that **State Targets > Position > Value** is set to 0 deg.
- 3 Run the simulation.
- 4 Plot the joint position and velocity with respect to time. To do this, at the MATLAB command prompt, you can enter this code:

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

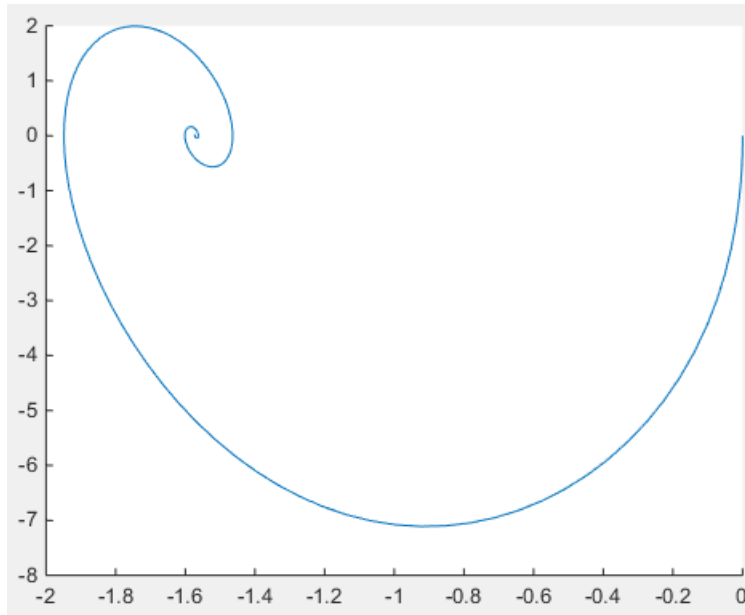
The figure shows the resulting plot. Note that the pendulum oscillations decay with time due to damping. At larger damping values, the pendulum becomes overdamped, and the oscillations disappear altogether.



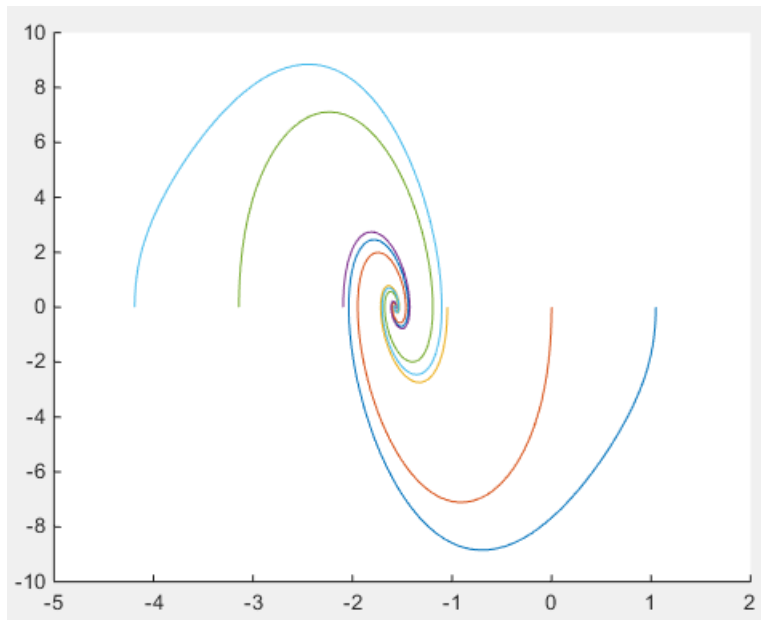


- 5 Plot the joint phase plot. To do this, at the MATLAB command prompt, you can enter this code:

```
figure;  
plot(q.data, w.data);  
The figure shows the resulting plot.
```



Try simulating the model using different starting angles. You can change the starting angle in the **State Targets > Position** menu 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.



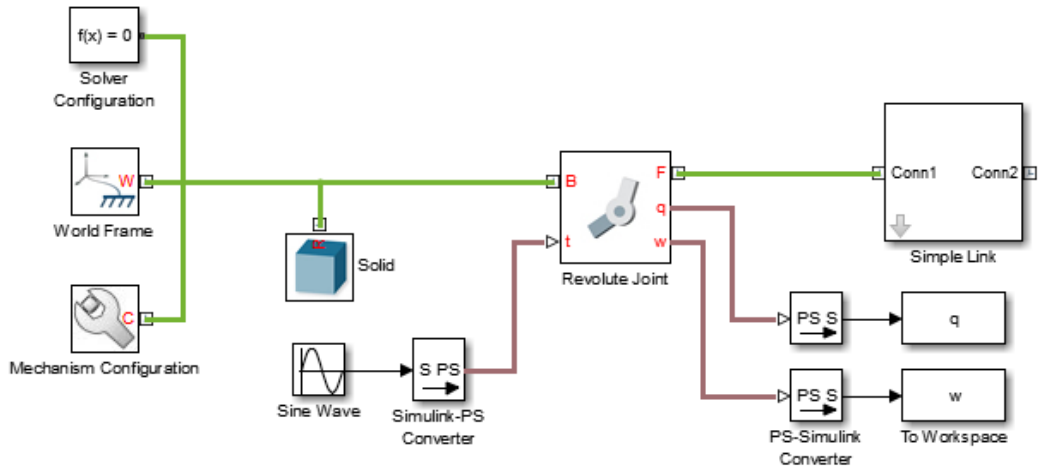
## Analyze Damped and Driven Pendulum

- 1 In the Revolute Joint block dialog box, set **Actuation** > **Torque** to **Provided by Input**. The block exposes a physical signal input port that you can use to prescribe the joint actuation torque.
- 2 Drag these blocks into 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 Simscape physical signal compatible with SimMechanics blocks.

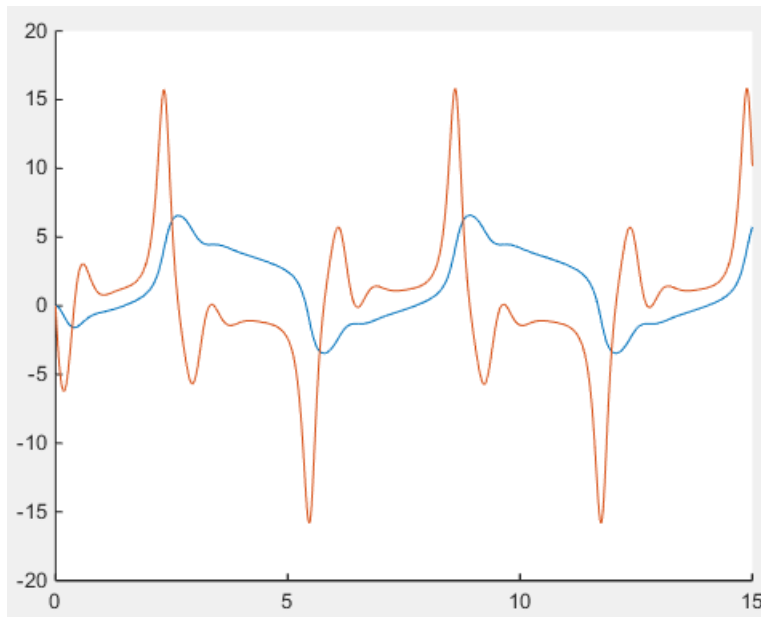
- 3 Connect the blocks as shown in the figure.



- 4 In the Sine Wave block dialog box, set **Amplitude** to 0.06. This amplitude corresponds to an actuation torque oscillating between -0.06 N and 0.06 N.
- 5 In the Revolute Joint block dialog box, ensure that **State Targets > Position > Value** is set to 0 deg.
- 6 Run the simulation.
- 7 Plot the joint position and velocity with respect to time. To do this, at the MATLAB command prompt, you can enter this code:

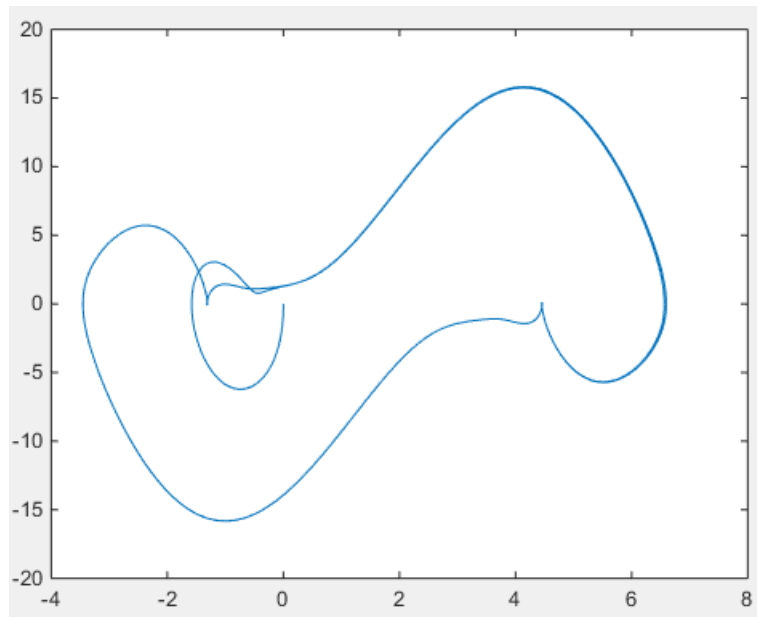
```
figure;
hold on;
plot(q);
plot(w);
```

The figure shows the resulting plot.



- 8** Plot the joint phase plot. To do this, at the MATLAB command prompt, you can enter this code:

```
figure;  
plot(q.data, w.data);  
The figure shows the resulting plot.
```



## SimMechanics First and Second Generation Comparison

SimMechanics software contains two technologies: First Generation and Second Generation. First-generation technology includes the block library and visualization utility found in SimMechanics releases prior to R2012a. Second-generation technology introduces a simpler modeling paradigm with a new block library, a powerful computational engine, an advanced visualization utility based on computer graphics, and tighter integration with Simscape products.

SimMechanics first- and second-generation technologies have different sets of capabilities. Which technology to use depends on the effects you need to model. SimMechanics Second Generation is recommended for all models except those requiring time-varying constraints or variable mass. You should also use SimMechanics First Generation if automatically updating CAD-imported models is important to you.

The table summarizes the differences in functionality between the two SimMechanics generations.

Feature	SimMechanics First Generation	SimMechanics Second Generation
Mass/Inertia Calculation	Manual only	Automatic or manual
Solid Geometry	No	Yes
Animation Replay	No	Yes
3-D Model Exploration	Limited	Yes
Initial State Targets	Limited	Yes
Simscape Logging	No	Yes
Code Generation	Yes	Yes
CAD Import	Yes	Yes <sup>1</sup>
Motion Actuation	Yes	Yes
Force/Torque Sensing	Yes	Yes
Complex Constraints <sup>2</sup>	Yes	Yes <sup>3</sup>
Variable Mass/Gravity	Yes	Yes <sup>4</sup>
Gravitational Fields	No	Yes

<sup>1</sup>CAD update supported only in SimMechanics First Generation

<sup>2</sup>Point-on-curve, gear, velocity, and screw constraints

<sup>3</sup>All except velocity constraints

<sup>4</sup>Variable gravity only

SimMechanics continues to support first-generation technology. You can maintain and simulate legacy models built with first-generation blocks. You also can still create a new first-generation model using the SimMechanics First Generation block library.