

The first three tutorials are selected from the MATLAB/Simulink help. The fourth example is a simple SimMechanics example which can help you learn the SimMechanics more. You can find more detailed information in the software help.

There is no need to submit any report and you just need to finish the three tutorials as well as the example.

## Modeling and Simulating a Simple Machine

### On this page...

[Modeling the Simple Pendulum](#)  
[Opening the SimMechanics Block Library](#)  
[The World Coordinate System and Gravity](#)  
[Configuring the Ground](#)  
[Configuring the Body](#)  
[Configuring the Joint](#)  
[Adding a Sensor and Starting the Simulation](#)

### Modeling the Simple Pendulum

In this first tutorial, you drag, drop, and configure the most basic blocks needed for any mechanical model, as well as add some sensors to measure motion. The tutorial guides you through the most basic aspects of model-building.

The end result is a model of a simple pendulum, a system with one body and open [topology](#). The pendulum is a swinging steel rod. We strongly recommend that you work through this tutorial first before moving on to the next section, [Visualizing a Simple Machine](#).

#### A Simple Pendulum: A Swinging Steel Rod

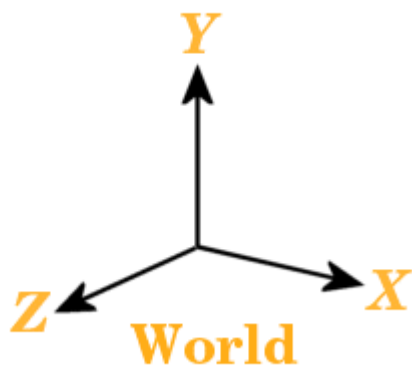


### Opening the SimMechanics Block Library

Following one of the ways described earlier in the [Accessing the Libraries](#) section in this chapter, open the SimMechanics library. From there, open a new, empty Simulink model window.

### The World Coordinate System and Gravity

Before you configure a Ground block, you need to understand the internally defined fixed or "absolute" SimMechanics [coordinate system](#) (CS) called *World*. The World CS sits at rest in the inertial [reference frame](#) also called World. The World CS has an origin (0,0,0) and a triad of right-handed, orthogonal coordinate axes.



The default World coordinate axes are defined so that

- +x points right
- +y points up (gravity in -y direction)
- +z points out of the screen, in three dimensions

The vertical direction or up-and-down is determined by the gravity vector direction (acceleration  $\mathbf{g}$ ) relative to the World axes. Gravity is a background property of a model that you can reset before starting a simulation, but does not dynamically change during a simulation.

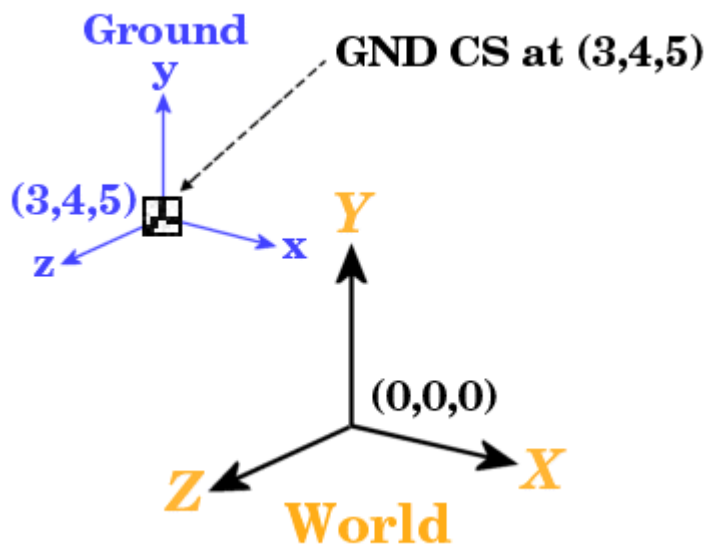
See [Configuring SimMechanics Models in Simulink](#) in [Running Mechanical Models](#) for displaying global mechanical properties of SimMechanics models.

### Configuring the Ground

World serves as the single absolute CS that defines all other CSs. But you can create additional [ground points](#) at rest in World, at positions other than the World origin, by using Ground blocks. Ground blocks, representing ground points, play a dynamical role in mechanical models. They function as immobile bodies and also serve to implement a machine's mechanical environment.

**Minimum Ground Blocks** Every machine model must have *at least one* Ground block. Exactly one Ground block in every machine must be connected to a Machine Environment block.

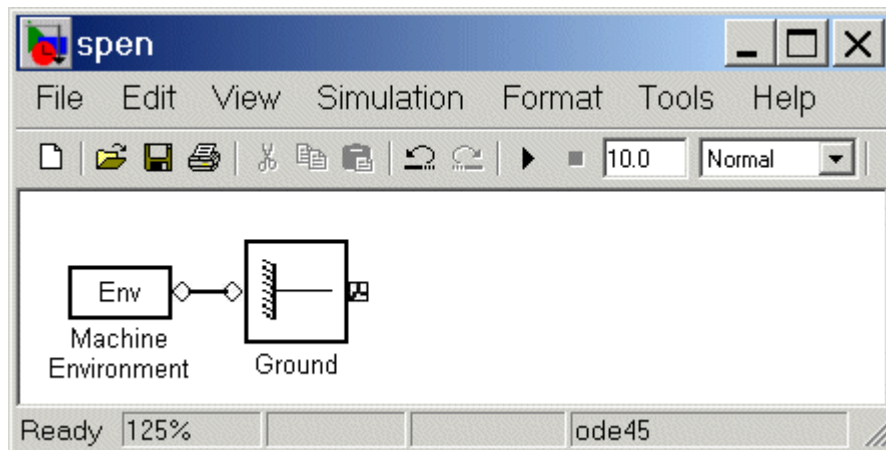
### A Ground Point Relative to World



### Steps to Configuring the Ground Block

Now place a fixed ground point at position (3,4,5) in the World CS:

1. In the SimMechanics library, open the Bodies library.
2. Drag and drop a Ground and a Machine Environment block from the Bodies library into the model window.
3. Open the Ground block dialog box. Into the **Location [x y z]** field, enter the vector [ 3 4 5 ]. Select the **Show Machine Environment port** check box. Click **OK** to close the dialog. Connect the environment block.



### Properties of Grounds

At every ground point, a *Grounded CS* is automatically created:

- The origin of each Grounded CS is the ground point itself.
- The Grounded CS axes are always fixed to be parallel to the World CS axes, as shown in the figure [A Ground Point Relative to World](#).

### Configuring the Body

While you need one Machine Environment and at least one Ground block to make a mechanical model, a real machine consists of one or more rigid bodies. So you need to translate the components of a real machine into block representations. This section explains how you use a Body block to represent each rigid body in your system:

- [Characteristics of a Body Block](#)
- [Properties of the Simple Pendulum Body](#)
- [Configuring the Body Dialog](#)

Although the [body](#) is the most complicated component of a machine, the Body block does not use the full geometric shape and mass distribution of the body. A SimMechanics model uses only certain mass properties and simplified geometric information about the body's center of gravity, its orientation, and the coordinate systems attached to the body. The [Representing Motion](#) chapter explains in detail how to orient bodies and their coordinate systems.

Setting these properties sets the body's initial conditions of motion, if you do nothing else to the Body block or its connected Joints before simulating.

### Characteristics of a Body Block

The main characteristics of a Body block are its *mass properties*, its *position* and *orientation* in space, and its attached *Body coordinate systems* (CSs).

The mass properties include the [mass](#) and [inertia tensor](#). The mass is a real, positive scalar. The inertia tensor is a real, symmetric 3-by-3 matrix. It does not have to be diagonal.

The position of the body's [center of gravity](#) (CG) and *orientation* relative to some coordinate system axes indicate where the body is and how it is rotated. These are the body's initial conditions during construction of the model and remain so when you start the simulation, unless you change them before starting.

The attached [Body CSs](#) (their origins and coordinate axes) are fixed rigidly in the body and move with it. The minimum CS set is one, the CS at the CG (the CG CS), with its CS origin at the center of gravity of the body. The default CS set is three, the CG CS and two additional CSs called CS1 and CS2 for connecting to Joints on either side. See the next section, [Configuring the Joint](#).

Beyond the minimum CS at the CG, you can attach as many Body CSs on one Body as you want. You need a separate CS for each connected Joint, Constraint, or Driver and for each attached [Actuator](#) and [Sensor](#).

The inertia tensor components are interpreted in the CG CS, setting the orientation of the body relative to the CG CS axes. The orientation of the CG CS axes relative to the World axes then determines the absolute initial orientation of the body. Since the CG CS axes remain rigidly fixed in the body during the simulation, this relative orientation of the CG CS axes and the body does not change during motion. The inertia tensor components in the CG CS also do not change. As the body rotates in inertial space, however, the CG CS axes rotate with it, measured with respect to the absolute World axes.

### Properties of the Simple Pendulum Body

The simple pendulum is a uniform, cylindrical steel rod of length 1 meter and diameter 2 cm. The initial condition is the rod lying parallel to the negative *x*-axis, horizontal in the gravity field. One end of the rod, the fixed pivot for the rod to swing, is located at the ground point (3,4,5). Its coordinate system is called CS1. The center of gravity and the origin of the CG CS is the geometric center of the rod. Take the CG CS axes to be parallel to the World axes as you set up the pendulum.

Uniform steel has density  $\rho = 7.93 \text{ gm/cc}$  (grams per cubic centimeter). In the CG CS here, the inertia tensor  $I$  is diagonal, and  $I_{zz}$  controls the swinging about the *z*-axis, in the *x*-*y* plane. The inertia tensor is always evaluated with the origin of coordinates at the CG. For a rod of length  $L = 1 \text{ m}$  and radius  $r = 1 \text{ cm}$ , the [mass](#)  $m = \rho \pi r^2 L = 2490 \text{ gm}$  (grams), and the [inertia tensor](#)  $I$  reads

$$\begin{pmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{pmatrix} = \begin{pmatrix} \frac{mr^2}{2} & 0 & 0 \\ 0 & \frac{mL^2}{12} & 0 \\ 0 & 0 & \frac{mL^2}{12} \end{pmatrix} = \begin{pmatrix} 1250 & 0 & 0 \\ 0 & 2.08 \times 10^6 & 0 \\ 0 & 0 & 2.08 \times 10^6 \end{pmatrix}$$

in  $\text{gm-cm}^2$  (gram-centimeters<sup>2</sup>). The *x*-axis is the cylinder's symmetry axis. Thus  $I_{yy} = I_{zz}$ .

The mass and geometric properties of the body are summarized in the following table and depicted in the figure [Equivalent Ellipsoid of Simple Pendulum with Body Coordinate Systems](#).

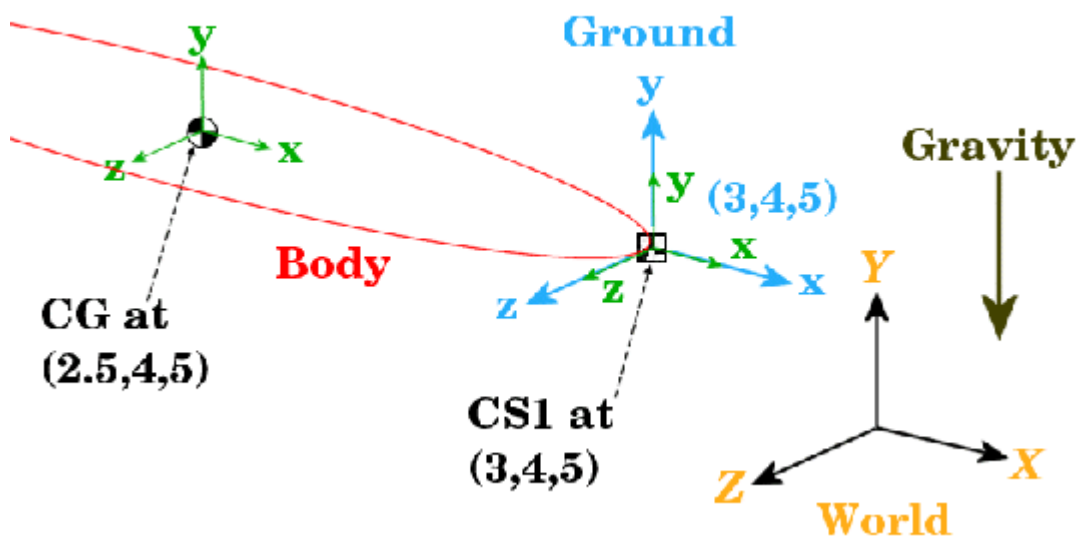
### Body Data for the Simple Pendulum

Property	Value
Mass (gm)	2490
Inertia tensor (kg-m <sup>2</sup> )	[1.25e-4 0 0; 0 0.208 0; 0 0 0.208]
CG Position/Origin (m)	[ 2.5 4 5 ]
CS1 Origin (m)	[ 3 4 5 ]

### Configuring the Body Dialog

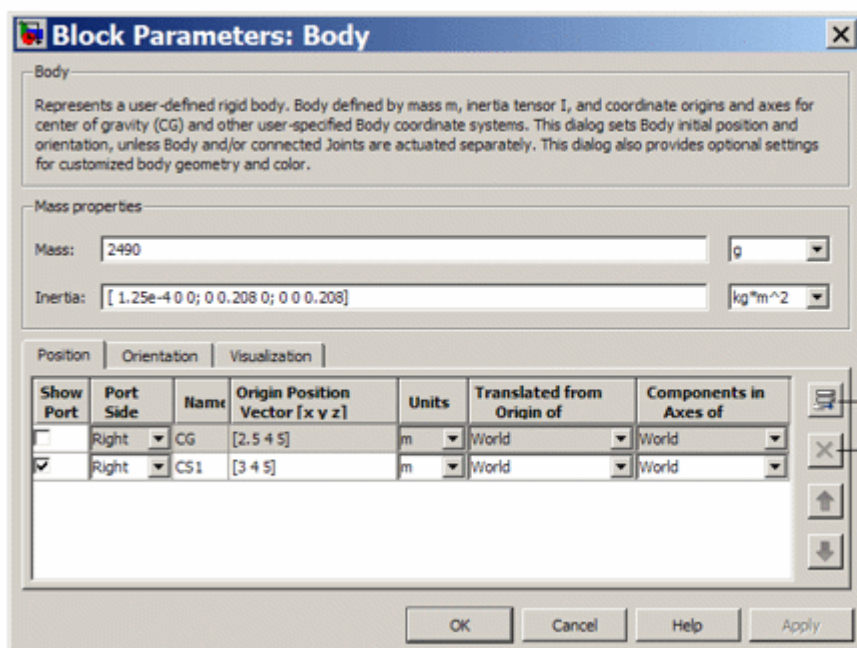
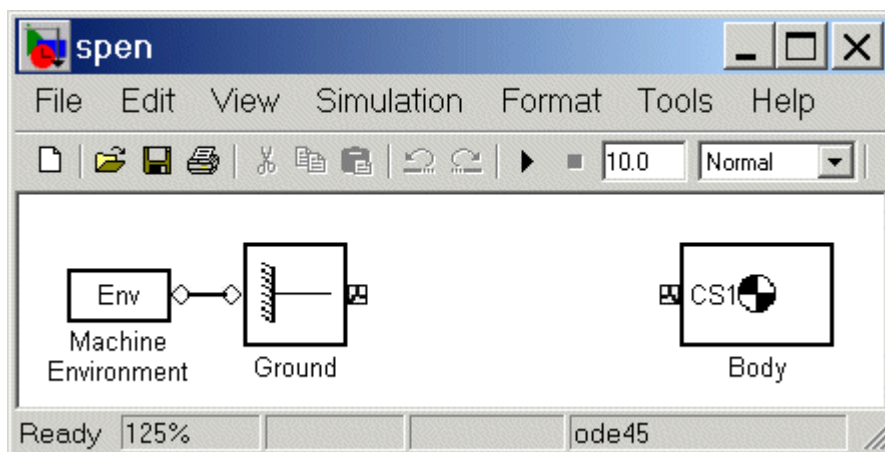
Take the steps to configuring a Body block in several stages.

### Equivalent Ellipsoid of Simple Pendulum with Body Coordinate Systems



**Adding the Body Block.** To start working with the Body block:

1. Open the Bodies library in the SimMechanics library.
2. Drag and drop a Body block into your model window.
3. Open the Body block dialog box. Note the two main areas you need to configure:
  - **Mass properties** — These are the mass and inertia tensor.
  - **Body coordinate systems** — These are the details about the position and orientation of the Body CSs.



**Specifying the Body's Mass Properties.** Now enter the body's mass and inertia tensor:

1. Use the data from the table [Body Data for the Simple Pendulum](#).  
In the **Mass** field, enter 2490 and change the units to g (grams).
2. In the **Inertia** field, enter [1.25e-4 0 0; 0 0.208 0; 0 0 0.208] and leave the default units as kg-m<sup>2</sup>.

**Specifying Body Coordinate Systems (Position).** Enter the translational position of the body and its Body CS origins in space:

1. Use the data from the table [Body Data for the Simple Pendulum](#), and work on the **Position** tab. Vectors are assumed translated from the World origin and oriented to the World axes.
2. Note the three default CSs in the Body dialog box. The CS at the CG is necessary for any Body, and you will connect CS1 to the Ground with a Joint shortly.

Delete CS2 by selecting its line in the Body CS list and clicking the **Delete** button in the Body CS controls.

You have two already existing CSs not on this Body that you can use to specify the positions of the Body CS origins that are on this Body:

- Preexisting World origin at [0 0 0]

- The **Adjoining CS** on the neighboring body, in this case the Grounded CS origin at [ 3 4 5 ]
3. Specify the CG and CS1 origins relative to World:
    - a. In the pull-down menu under **Translated from Origin of**, choose `World` for both coordinate systems, CG and CS1.
    - b. Under **Origin Position Vector**, specify the position of the origin of each CS, translated from the World origin:
      - [ 3 4 5 ] for CS1
      - [ 2.5 4 5 ] for CG
  4. Select a CS relative to whose coordinate axes the components of the vectors in the last step are measured. You choose these CS axes in the **Components in Axes of** menu. Select `World` for both CSs. Leave the units as `m` (meters).

**Specifying Body Coordinate Systems (Orientation).** Enter the rotational orientation of the body and its Body CS axes in space:

1. Work on the **Orientation** tab. The default orientation for all CS axes is parallel to World. The sign of all rotations is determined by the *right-hand rule*.  
  
In the figure [Equivalent Ellipsoid of Simple Pendulum with Body Coordinate Systems](#), the CS1 and CG axes are oriented parallel to the World axes, so the CS1 and CG axes need no rotation.
2. For both CSs, set the **Relative to CS** menu to `World`.
3. For CG and CS1, leave the **Orientation Vector** at default [ 0 0 0 ] and the **Specified Using Convention** at default `Euler X-Y-Z`. Close the Body dialog.

Show Port	Port Side	Name	Orientation Vector	Units	Relative CS	Specified Using Convention
<input type="checkbox"/>	Right	CG	[0 0 0]	deg	World	Euler X-Y-Z
<input checked="" type="checkbox"/>	Right	CS1	[0 0 0]	deg	World	Euler X-Y-Z

## Configuring the Joint

A mechanical system is made up of Bodies with geometric and mass information. But Bodies carry no information of how they move. The possible directions of motion that a Body can take are called its *degrees of freedom* (DoFs), and this section explains how you represent these DoFs by Joint blocks:

- [How to Connect a Joint Between Two Bodies](#)
- [Choosing a Revolute Joint for the Simple Pendulum](#)

**DoFs Are Relative** SimMechanics DoFs and the Joints that represent them are *relative* DoFs. That is, DoFs represent the possible motions between one body and another. So a DoF is defined by a pair of bodies, and you must connect every Joint to two and only two Bodies.

One (but not both) of the members of such a pair of Bodies can be a Ground. The other member Body of such a pair then has its motion defined relative to a fixed ground point. This fixed ground point does not have to be the same as the World origin. A system can have many such Ground-Body pairs and *must have at least one*.

## How to Connect a Joint Between Two Bodies

You represent relative motion of bodies with respect to one another by connecting their Body blocks with Joints. You can connect a Body to one or more Joints.



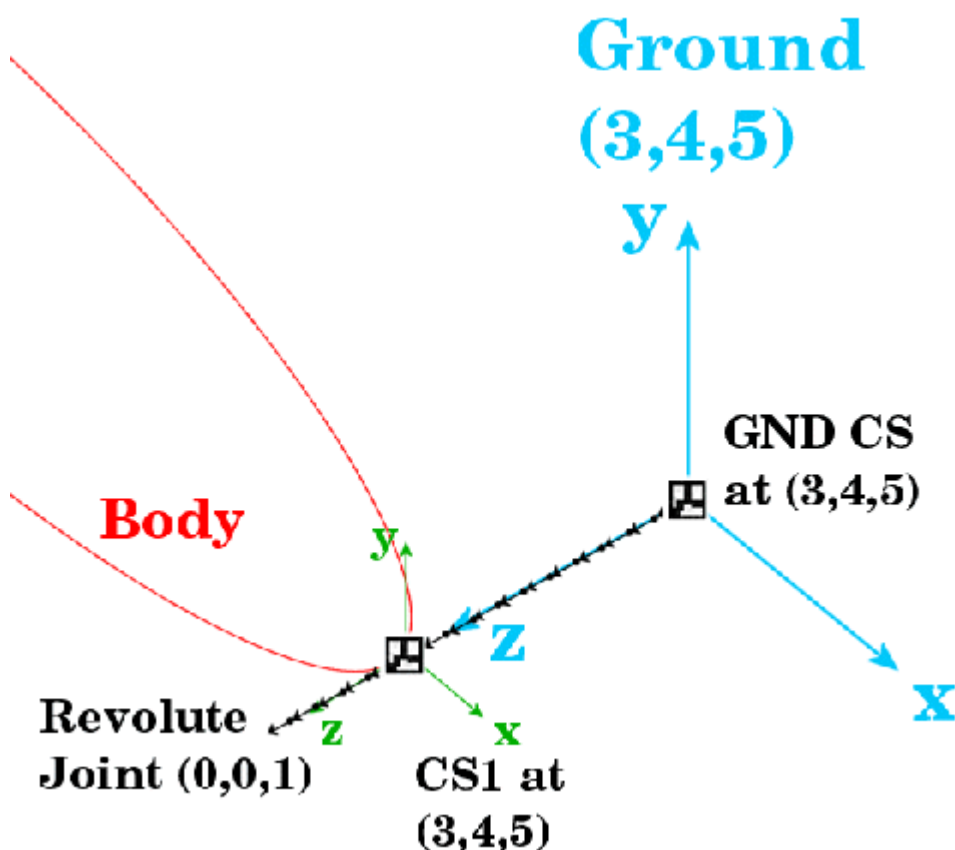
A Joint block is always connected to a specific point on the Body on either side of the Joint. The specific point for anchoring a Joint on a Body is the origin of a Body CS, and a Joint is therefore connected on one side to one Body at a Body CS origin, and on the other side to the other Body at a Body CS origin.

Usually a Body is connected to a Joint on either side, so the default [you saw earlier in this tutorial](#) for Body CSs in the Body dialog box is three Body CSs: the CS at the center of gravity (CG) and two other CSs (CS1 and CS2). But a Body at the end of a Body — Joint — ... — Body chain is connected to only one Joint.

### Choosing a Revolute Joint for the Simple Pendulum

In spite of the complexity of the concepts implicit in a Joint, the actual configuration of a Joint block is fairly simple. Here you insert and configure one revolute Joint block between the Ground and Body blocks you've already put into the model window.

#### A Simple Pendulum Connected to Ground by a Revolute



**Configuring the Revolute Joint Block.** The geometry of the Joint connection is shown in the figure preceding. The ground point at (3,4,5) and the CS1 at (3,4,5) are actually the same point in space, but have been separated in the figure for clarity. The revolute rotation axis is along the +z direction:

1. Open the Joints library in the block library.
2. Drag and drop a Revolute block into your model window.
3. Rotate the Revolute block so that you can connect the base (B) side of the Joint to the Ground block and the follower (F) side of the Joint to the Body block. Make the two connections.
4. Open the Revolute dialog box. In the **Parameters** area, on the **Axes** tab, configure the rotation axis to the World z-axis:
  - a. Enter [0 0 1] under **Axis of Action [x y z]**.
  - b. Leave the **Reference CS** at World.
  - c. Ignore the **Advanced** tab.

Note several important things:

- Under **Connection parameters**, the **Current base** is located at GND@Ground, which is the Grounded CS associated with the Ground block located at (3,4,5) in World.
  - Under **Connection parameters**, the **Current follower** is located at CS1@Body, which is the CS1 on Body1 located at (3,4,5) in World.
  - This Joint's **directionality** runs from Ground to Body along the +z-axis.
5. Close the Revolute dialog box.

**Block Parameters: Revolute**

**Revolute**  
Represents one rotational degree of freedom. The follower (F) Body rotates relative to the base (B) Body about a single rotational axis going through collocated Body coordinate system origins. Sensor and actuator ports can be added. Base-follower sequence and axis direction determine sign of forward motion by the right-hand rule.

**Connection parameters**

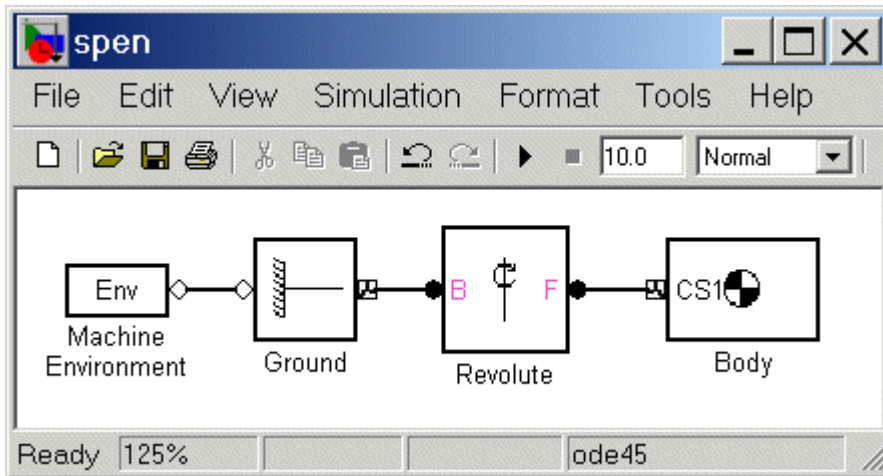
Current base: GND@Ground  
Current follower: CS1@Body  
Number of sensor / actuator ports: 0

**Parameters**

Name	Primitive	Axis of Action [x y z]	Reference CS
R1	revolute	[0 0 1]	World

OK Cancel Help Apply

Congratulations — you have now finished the simplest possible model of a machine: a connected block diagram of Ground–Joint–Body. Your model window should look like this.





### Adding a Sensor and Starting the Simulation

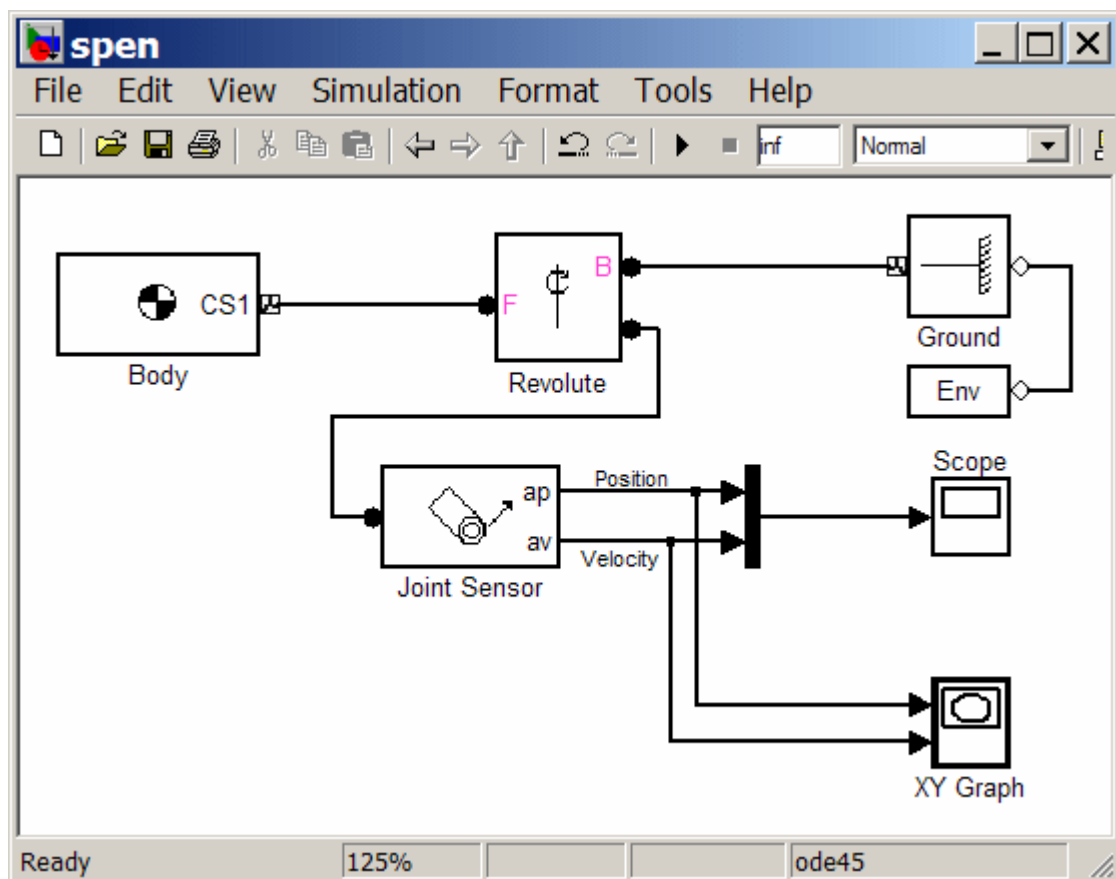
To measure the motion of the pendulum as it swings, you need to connect one or more Simulink Scope blocks to your model. The SimMechanics library of Actuators and Sensor blocks gives you the means to input and output Simulink signals to and from SimMechanics models. Sensors allow you to watch the mechanical motion once you start the simulation, as the following explain:

- [Connecting and Configuring the Pendulum Sensor](#)
- [Configuring the Machine Environment and Configuration Parameters](#)
- [Starting and Interpreting the Motion](#)

### Connecting and Configuring the Pendulum Sensor

In this example, you measure the angular motion of the revolute joint:

1. In the block library, open the Sensors & Actuators library. Drag and drop a Joint Sensor block into your model window.
2. Open the Revolute block. Change **Number of sensor/actuator ports** from 0 to 1 using the spinner menu. An open connector port  appears on the side of Revolute. Close Revolute.
3. Connect this connector port to the connector port on the Joint Sensor block. The open connector port changes to solid .
4. Open Joint Sensor. Select the **Angle** and the **Angular velocity** check boxes. Unselect the **Output selected parameters as one signal** check box.  
Leave the other defaults. Close the Joint Sensor block.
5. Open the Simulink Library Browser. From the Sinks library, drag and drop a Scope block and an XY Graph block into your model window. From the Signal Routing library, drag and drop a Mux block as well. Connect the Simulink outputs > on the Joint Sensor block to the Scope and XY Graph blocks as shown.



The lines from the outputs > to the Scope and XY Graph blocks are normal Simulink signal lines and can be branched. You cannot branch the lines connecting SimMechanics blocks to each other at the round connector ports ●.

6. Save your model for future reference as `spen.mdl`.

You now need to configure the global parameters of your model before you can run it.

### Configuring the Machine Environment and Configuration Parameters

The Configuration Parameters dialog box is a standard feature of Simulink. Reset its entries for this model to obtain more accurate simulation results.

1. In the Simulink menu bar, open the **Simulation** menu and click **Configuration Parameters** to open the Configuration Parameters dialog.
2. Select the **Solver** node of the dialog. Under **Solver options**, change **Relative tolerance** to  $1e-6$  and **Absolute tolerance** to  $1e-4$ . Change **Max step size** to  $0.2$ .

If you want the simulation to continue running without stopping, change **Stop time** to `inf`. The pendulum period is approximately 1.6 seconds.

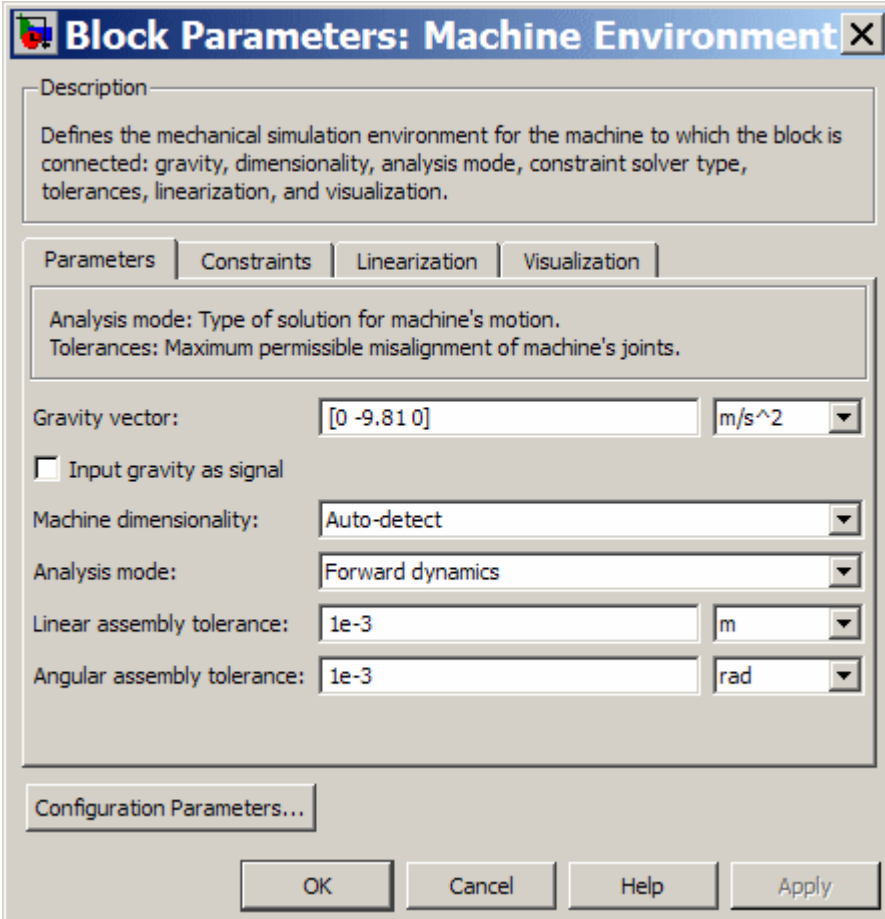
3. Close the Configuration Parameters dialog box.

A special feature of SimMechanics models is the Machine Environment block.

1. Open your block diagram's Machine Environment block dialog.

Note the default **Gravity vector**,  $[0 \ -9.81 \ 0] \text{ m/s}^2$ , which points in the  $-y$  direction, as shown in the figure [Equivalent Ellipsoid of Simple Pendulum with Body Coordinate Systems](#). The gravitational acceleration  $g = 9.81 \text{ m/s}^2$ .

2. Close the Machine Environment dialog.



**Block Parameters: Machine Environment**

Description

Defines the mechanical simulation environment for the machine to which the block is connected: gravity, dimensionality, analysis mode, constraint solver type, tolerances, linearization, and visualization.

Parameters | Constraints | Linearization | Visualization

Analysis mode: Type of solution for machine's motion.  
Tolerances: Maximum permissible misalignment of machine's joints.

Gravity vector: [0 -9.81 0] m/s<sup>2</sup>

☐ Input gravity as signal

Machine dimensionality: Auto-detect

Analysis mode: Forward dynamics

Linear assembly tolerance: 1e-3 m

Angular assembly tolerance: 1e-3 rad

Configuration Parameters...

OK Cancel Help Apply

### Starting and Interpreting the Motion

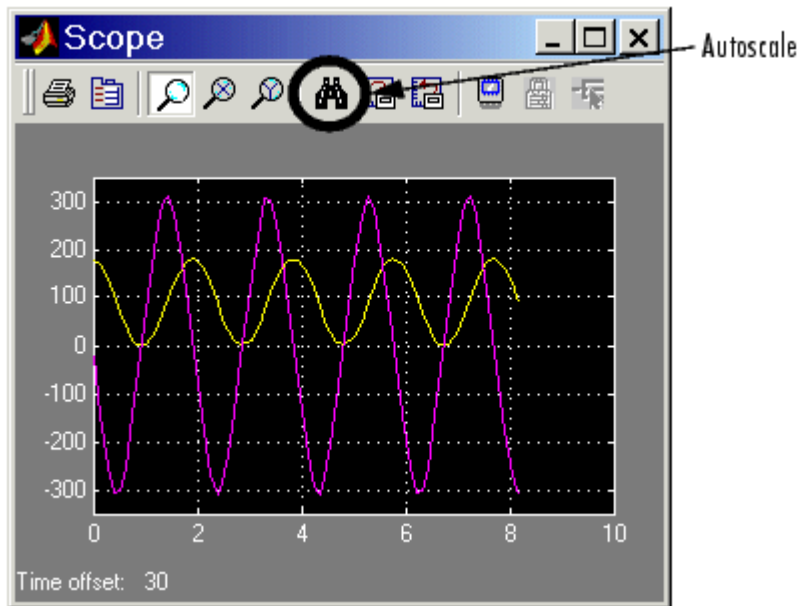
You can now start your simulation and watch the pendulum motion via the Scope and XY Graph blocks:

1. Open the XY Graph block dialog box. Set the following parameters.

Parameter	Value
x-min	0
x-max	200
y-min	-500
y-max	500

Leave **Sample time** at default and close the dialog.

2. Open the Scope block and start the model. The XY Graph opens automatically when you start the simulation.
3. View the full motion of both angle and angular velocity (in degrees and degrees per second, respectively) as functions of time in Scope. Click Autoscale if the motion is not fully visible.



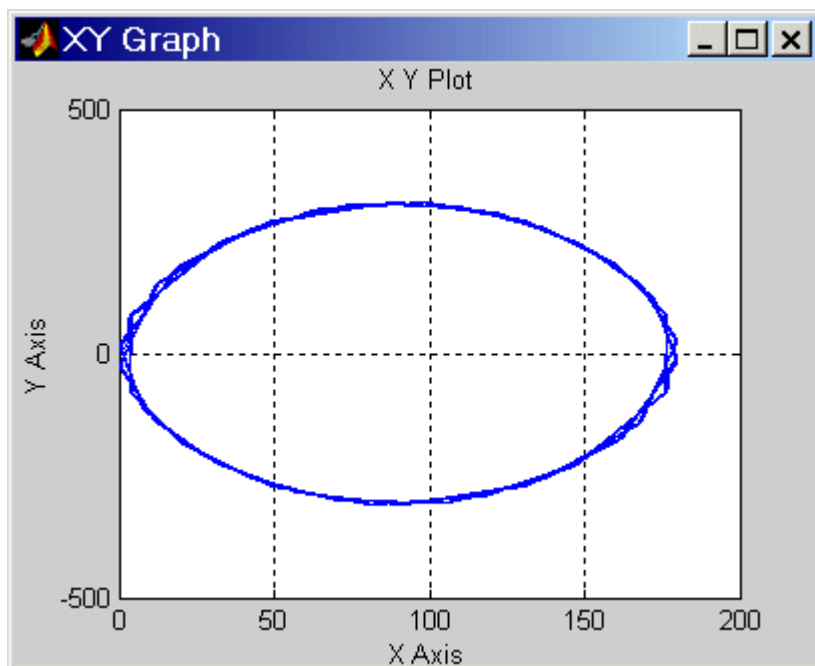
The motion is periodic but not simple harmonic (sinusoidal), because the amplitude of the swing is so large (180 degrees from one turning point to the other). Note that the zero of angle is the initial horizontal angle, not the vertical. The zeros of motion are always the initial conditions.

The XY Graph shows the angle versus angular velocity, with no explicit time axis. These two variables trace out a figure similar to an ellipse, because of the conservation of total energy  $E$ :

$$\frac{1}{2}J\left(\frac{d\theta}{dt}\right)^2 + mgh*(1 - \sin\theta) = E = \text{constant}$$

where  $J = I_{zz} + mL^2/4$  is the inertial moment of the rod about its pivot point (not the center of gravity). The two terms on the left side of this equation are the kinetic and potential energies, respectively. The coordinate-velocity space is the *phase space* of the system.

#### Phase Space Plot of Simple Pendulum Motion: Angular Velocity Versus Angle



The directionality of the Revolute Joint assumes that the rotation axis lies in the +z direction. Looking at the pendulum from the front, follow the figures

- [A Ground Point Relative to World](#)
- [Equivalent Ellipsoid of Simple Pendulum with Body Coordinate Systems](#)
- [A Simple Pendulum Connected to Ground by a Revolute](#)

Positive angular motion from this perspective is counterclockwise, following the [right-hand rule](#).

The next tutorial walks you through visualizing and animating this same simple pendulum model.

## Modeling and Simulating a Closed-Loop Machine

### On this page...

[Modeling the Four Bar Mechanism](#)  
[Viewing a Mechanical Drawing of the Four Bar Mechanism](#)  
[Counting the Degrees of Freedom](#)  
[Configuring the Mechanical Environment](#)  
[Setting Up the Block Diagram](#)  
[Configuring the Ground and Joints](#)  
[Configuring the Bodies](#)  
[Sensing Motion and Running the Model](#)  
[For More About the Four Bar Mechanism](#)

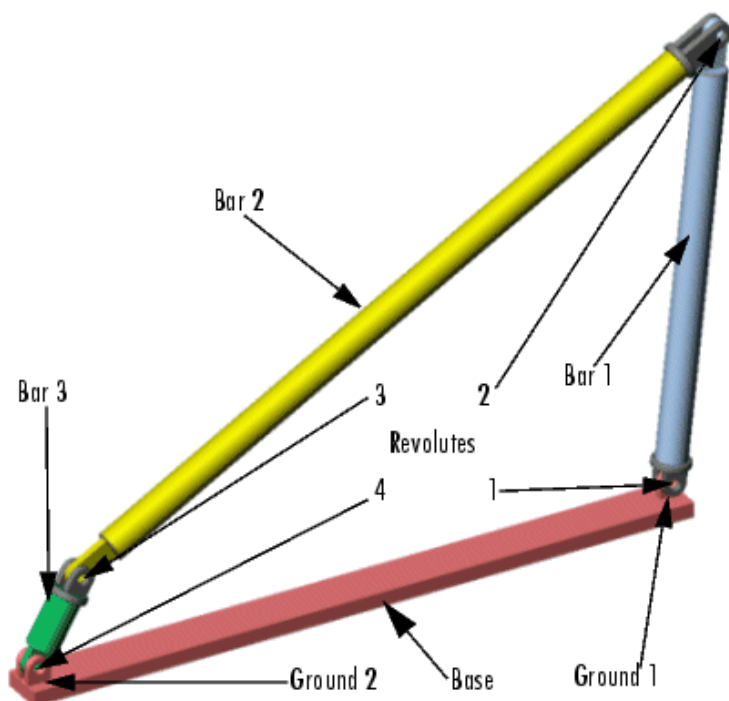
### Modeling the Four Bar Mechanism

In this tutorial, you build a model of a planar, four bar mechanism and practice using some of the important SimMechanics features.

You are urged to work through [Modeling and Simulating a Simple Machine](#) and [Visualizing a Simple Machine](#) in this chapter before proceeding with this section. Learn more about how to position and orient bodies in the [Representing Motion](#) chapter.

The system consists of three moving bars of homogeneous steel, two connected at one end each to ground points and a third crossbar connecting the first two. The base acts as an immobile fourth bar, with a Ground at each end. The mechanism forms a single closed loop, and its motion is confined to two dimensions.

#### A Four Bar Mechanism



The elementary parts of the mechanism are the bodies, while the revolute joints are the idealized rotational [degrees of freedom](#) (DoFs) at each body-to-body contact point. The bodies and the joints expressing the bodies' relative motions must be translated into corresponding SimMechanics blocks. If you want, you can add elaborations such as Constraints, Drivers, Sensors, and Actuators to this essential block diagram.

#### Viewing a Mechanical Drawing of the Four Bar Mechanism

Click [here](#) to open a detailed mechanical drawing of the four bar system.

#### Counting the Degrees of Freedom

The three moving bars are constrained to move in a plane. So each bar has two translational and one rotational DoFs, and the total number of mechanical DoFs, before counting constraints, is  $3 \cdot (2+1) = 9$ .



Because the motion of the bars is constrained, however, not all of these nine DoFs are *independent*:

- In two dimensions, each connection of a body with another body or with a ground point imposes two restrictions (one for each coordinate direction).

Such a restriction effectively eliminates one of the two body ends as independently moving points, because its motion is determined by the next body's end.

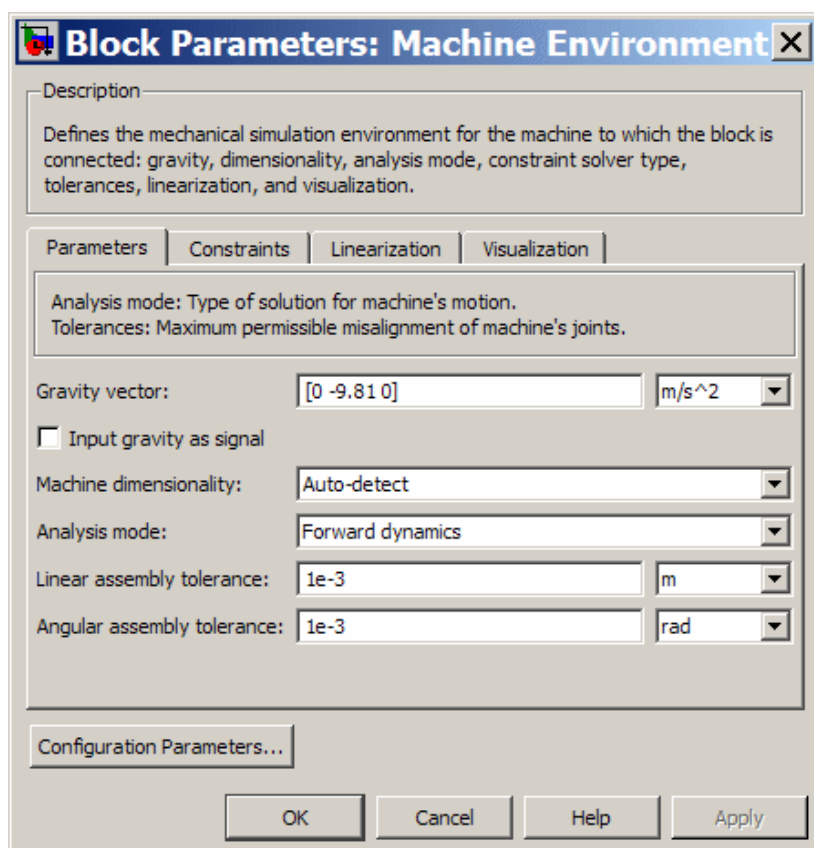
- There are four such body-body or body-ground connections and therefore eight restrictions implicit in the machine's geometry.

The eight restrictions on the nine apparent DoFs reduce the DoFs to one,  $9 - 8 = 1$ . There are four rotational DoFs between bars or between bars and grounds. But three of these are dependent. Specifying the state of one rotational DoF fully specifies the other three.

### Configuring the Mechanical Environment

Open a new blank model window from the SimMechanics library. From the Bodies library, drag in and drop a Machine Environment block and a Ground block. Enable the Ground's Machine Environment port and connect the environment block to the Ground.

First you need to configure the machine's mechanical settings. Open the Machine Environment block. The block dialog box appears.



### The Machine Environment Dialog Box Tabs

Click the four tabs in succession to display each pane.

Tab	Function
Parameters	Controls general settings for mechanical simulations
Constraints	Sets constraint tolerances and how constraints are interpreted
Linearization	Controls how SimMechanics models are linearized with Simulink
Visualization	Chooses whether or not to visualize the machine

Note some important features of this dialog box:

- The **Gravity vector** field specifies the magnitude and direction of gravitational acceleration and sets the vertical or up-

down direction.

- The **Linear** and **Angular assembly tolerance** fields are also set here. Change **Angular assembly tolerance** to  $1e-3$  deg (degrees). (See [Controlling Machine Assembly](#) in the [Running Mechanical Models](#) chapter.)
- Leave the other defaults.

Close the dialog by clicking **OK**.

### Starting Visualization

**Tip** If possible, open the visualization window before building a model. With it, you can keep track of your model components and how they are connected, as well as correct mistakes.

To visualize the bodies as you build the model, go to the **SimMechanics** node of the Configuration Parameters dialog:

1. Select the **Display machines after updating diagram** check box. If you want to animate the simulation later when you run the model, select the **Show animation during simulation** check box as well. Click **OK** or **Apply**.

Then select **Update Diagram** from the **Edit** menu or enter **Ctrl+D** at the keyboard. The visualization window opens.

2. In the **Model** menu, select **Body Geometries**, then **Ellipsoids**.

As you add and change bodies in your model, you can update the display in your window at any time by updating your diagram.

### Setting Up the Block Diagram

In this set of steps, you create Bodies, position them, connect them with Joints, then configure the Body and Joint properties. The Body dialog boxes give you many ways to represent the same system in the same physical state. This section explains one way.

Alternative, equivalent ways of configuring Bodies are discussed in [Body Coordinate Systems](#).

#### MAT-File Data Entry

The geometric and mass properties you need to specify for the Grounds and Bodies in this model are listed in the tables of the following two sections, [Configuring the Ground and Joints](#) and [Configuring the Bodies](#).

Instead of typing the numerical values of these properties into the dialog boxes, you can load the variable set you need into the workspace by entering

```
load fourbar_data
```

at the MATLAB command line. The variable name for each property is given in the tables. Just enter the appropriate variable names in the appropriate fields as you come to them in the dialog boxes.

If you are working with online Help, you can click here to load the [fourbar\\_data.mat](#) file into the workspace.

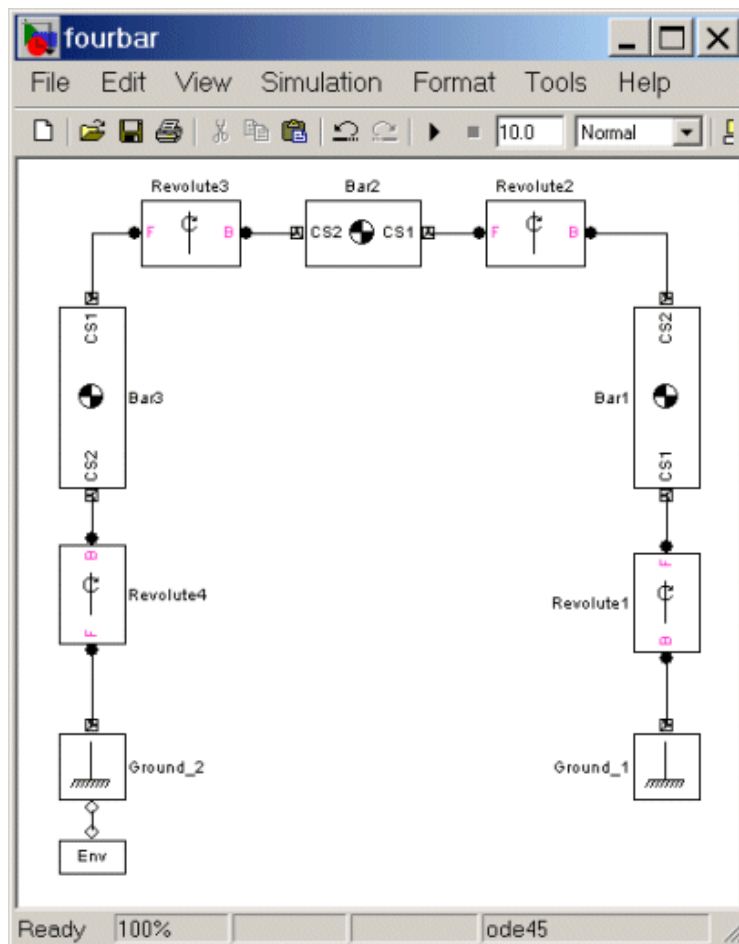
#### Block Diagram Setup

Your model already has one environment block and one ground block. Assemble the full model with these steps:


1. In the block library, open the Bodies library. Drag and drop another Ground block and three Body blocks into the new model window. Close the Bodies library.
2. From the Joints library, drag and drop four Revolute blocks into the model window.
3. Rotate and connect the blocks in the pattern shown in the following figure or with an equivalent block diagram [topology](#).

Use the block names shown in this figure for later consistency.

**Connected Environment, Ground, Body, and Joint Blocks for the Four Bar**



**Block Diagram Topology.** The [topology](#) of the block diagram is the connectivity of its elements. The elements are the Bodies and Grounds, connected by the Joints. Unlike the model of [Modeling and Simulating a Simple Machine](#), the four bar mechanism is a closed-loop mechanism. The two Ground blocks represent points on the same absolute, immobile body, and they close the loop of blocks. The simple pendulum has only one ground and does not close its block connections.

To maintain consistent Body motion direction, make sure the Body coordinate system (CS) port  pairs on each Body follow the sequence CS1-CS2, CS1-CS2, etc., for each bar, moving from Ground\_1 to Ground\_2, from right to left, as shown. To make the Joints consistent with the Body motion, the base-follower pairs B-F, B-F, etc., should follow the same right-to-left sequence.

### Configuring the Ground and Joints

Now configure the Ground blocks with the data from the following table. Grounded coordinate systems (CSs) are automatically created.

#### Geometry of the Four Bar Base

This table summarizes the geometry of ground points.

#### Geometric Properties of the Four Bar Grounds

Property	Value	MAT-File Variable
Ground_1 point (m)	[ 0.433 0.04 0 ]	gpoint_1
Ground_2 point (m)	[ -0.434 0.04 0 ]	gpoint_2

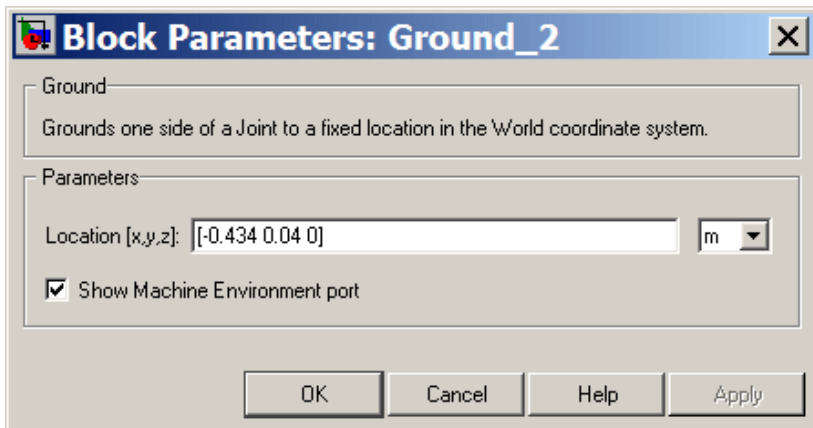
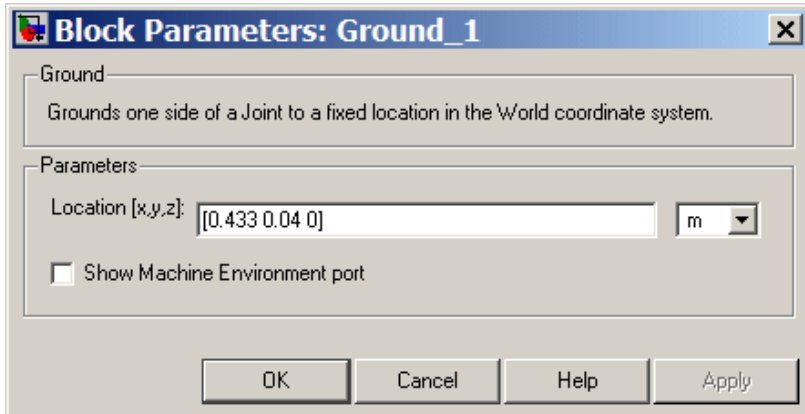
The base of the mechanism has these measurements:

- The base is horizontal, with length 86.7 cm.
- Ground\_1 represents the ground point 43.3 cm to the right of the World CS origin.
- Ground\_2 represents the ground point 43.4 cm to the left of the World CS origin.
- The bottom revolutes are 4 cm above the origin (x-z) plane.

## Setting Up the Grounds

To represent ground points on the immobile base, you need to configure the Ground blocks. Use the variable names if you've loaded `fourbar_data.mat` into your workspace:

1. Open Ground\_1 and enter [ 0.433 0.04 0 ] or `gpoint_1` in the **Location** field.
2. Open Ground\_2 and enter [-0.434 0.04 0 ] or `gpoint_2` in the **Location** field.
3. Leave both pull-down menus for units at default m (meters).



## Configuring the Revolute Joints

The three nongrounded bars move in the plane of your screen (x-y plane), so you need to make all the Revolute axes the z-axis (out of the screen):

1. Open each Revolute's dialog box in turn. In its **Parameters** area, note on the **Axes** tab that the z-axis is the default: **Axis of Action** is set to [0 0 1] in each, relative to **Reference CS** World. Leave these defaults.

A Revolute block contains only one primitive joint, a single revolute DoF. So the **Primitive** is automatically revolute. Its name within the block is R1.

2. Leave these Revolute joint block defaults and ignore the **Advanced** tab.

The Body CS and base-follower joint directionality should be set up as shown in the block diagram of the figure [Connected Environment, Ground, Body, and Joint Blocks for the Four Bar](#). In the **Connection parameters** area, the default Joint directionality for each Revolute automatically follows the right-to-left sequence of Grounded and Body CSs:

- Revolute1: Base to follower: GND@Gound\_1 to CS1@Bar1
- Revolute2: Base to follower: CS2@Bar1 to CS1@Bar2
- Revolute3: Base to follower: CS2@Bar2 to CS1@Bar3
- Revolute4: Base to follower: CS2@Bar3 to GND@Ground\_2

In this Joint directionality convention,

- At each Joint, the leftward Body moves relative to the rightward Body.
- The rotation axis points in the +z direction (out of the screen).

- Looking at the mechanism from the front in the figure, [A Four Bar Mechanism](#), the positive rotational sense is counterclockwise. All Joint Sensor and Actuator data are interpreted in this sense.

**Block Parameters: Revolute1**

**Revolute**  
Represents one rotational degree of freedom. The follower (F) Body rotates relative to the base (B) Body about a single rotational axis going through collocated Body coordinate system origins. Sensor and actuator ports can be added. Base-follower sequence and axis direction determine sign of forward motion by the right-hand rule.

**Connection parameters**

Current base: GND@Ground\_1  
Current follower: CS1@Bar1  
Number of sensor / actuator ports: 0

**Parameters**

Axis | Advanced

Name	Primitive	Axis of Action [x y z]	Reference CS
R1	revolute	[0 0 1]	World

OK Cancel Help Apply

**Block Parameters: Revolute2**

**Revolute**  
Represents one rotational degree of freedom. The follower (F) Body rotates relative to the base (B) Body about a single rotational axis going through collocated Body coordinate system origins. Sensor and actuator ports can be added. Base-follower sequence and axis direction determine sign of forward motion by the right-hand rule.

**Connection parameters**

Current base: CS2@Bar1  
Current follower: CS1@Bar2  
Number of sensor / actuator ports: 0

**Parameters**

Axis | Advanced

Name	Primitive	Axis of Action [x y z]	Reference CS
R1	revolute	[0 0 1]	World

OK Cancel Help Apply

**Block Parameters: Revolute3**

**Revolute**  
Represents one rotational degree of freedom. The follower (F) Body rotates relative to the base (B) Body about a single rotational axis going through collocated Body coordinate system origins. Sensor and actuator ports can be added. Base-follower sequence and axis direction determine sign of forward motion by the right-hand rule.

**Connection parameters**  
 Current base: CS2@Bar2  
 Current follower: CS1@Bar3  
 Number of sensor / actuator ports: 0

**Parameters**  
 Axes | Advanced

Name	Primitive	Axis of Action [x y z]	Reference CS
R1	revolute	[0 0 1]	World

OK Cancel Help Apply

**Block Parameters: Revolute4**

**Revolute**  
Represents one rotational degree of freedom. The follower (F) Body rotates relative to the base (B) Body about a single rotational axis going through collocated Body coordinate system origins. Sensor and actuator ports can be added. Base-follower sequence and axis direction determine sign of forward motion by the right-hand rule.

**Connection parameters**  
 Current base: CS2@Bar3  
 Current follower: GND@Ground\_2  
 Number of sensor / actuator ports: 0

**Parameters**  
 Axes | Advanced

Name	Primitive	Axis of Action [x y z]	Reference CS
R1	revolute	[0 0 1]	World

OK Cancel Help Apply

### Configuring the Bodies

Setting the Body properties is similar for each bar, but with different parameter values entered into each dialog box:

- Mass properties

- Lengths and orientations
- Center of gravity (CG) positions
- Body coordinate systems (CSs)

In contrast to the first tutorial, where you specify Body CS properties with respect to the absolute World CS, in this tutorial, you specify Body CS origins on the bars in relative coordinates, displacing Bar1's CS1 relative to Ground\_1, Bar2's CS1 relative to Bar1, and so on, around the loop. You can refer the definition of a Body CS to three types of coordinate systems:

- To [World](#)
- To the other Body CSs on the same Body
- To the [Adjoining CS](#) (the coordinate system on a neighboring body or ground directly connected by a Joint to the selected Body CS).

The components of the displacement vectors for each Body CS origin continue to be oriented with respect to the World axes. The rotation of each Body's CG CS axes is also with respect to the World axes, in the Euler X-Y-Z convention.

The following three tables summarize the body properties for the three bars.

#### Bar1 Mass and Body CS Data (MKS Units)

Property	Value	Variable Name
Mass	5.357	m_1
Inertia tensor	[1.07e-3 0 0; 0 0.143 0; 0 0 0.143]	inertia_1
CG Origin	[0.03 0.282 0] from CS1 in axes of World	cg_1
CS1 Origin	[0 0 0] from Adjoining in axes of World	cs1_1
CS2 Origin	[0.063 0.597 0] from CS1 in axes of World	cs2_1
CG Orientation	[0 0 83.1] from World in convention Euler X-Y-Z	orientcg_1

#### Bar2 Mass and Body CS Data (MKS Units)

Property	Value	Variable Name
Mass	9.028	m_2
Inertia tensor	[1.8e-3 0 0; 0 0.678 0; 0 0 0.678]	inertia_2
CG Origin	[-0.427 -0.242 0] from CS1 in axes of World	cg_2
CS1 Origin	[0 0 0] from Adjoining in axes of World	cs1_2
CS2 Origin	[-0.87 -0.493 0] from CS1 in axes of World	cs2_2
CG Orientation	[0 0 29.5] from World in convention Euler X-Y-Z	orientcg_2

#### Bar3 Mass and Body CS Data (MKS Units)

Property	Value	Variable Name
Mass	0.991	m_3
Inertia tensor	[2.06e-4 0 0; 0 1.1e-3 0; 0 0 1.1e-3]	inertia_3
CG Origin	[-0.027 -0.048 0] from CS1 in axes of World	cg_3
CS1 Origin	[0 0 0] from Adjoining in axes of World	cs1_3
CS2 Origin	[0 0 0] from Adjoining in axes of World	cs2_3
CG Orientation	[0 0 60] from World in convention Euler X-Y-Z	orientcg_3

## Configuring the Bodies

Here are the common steps for configuring the Body dialogs of all three bars. See the three preceding tables for Body dialog box mass property (mass and inertia tensor) entries. The units are MKS: lengths in meters (m), masses in kilograms (kg), and inertia tensors in kilogram-meters<sup>2</sup> (kg-m<sup>2</sup>).

1. Open all three Body dialogs for each bar. Enter the mass properties for each from the tables in the **Mass** and **Inertia** fields.
2. Now work in the Body coordinate systems area, the **Position** tab:
  - a. Set the **Components in Axes of** menu, for each Body CS on each bar, to World.
  - b. Leave units as default m (meters).
3. Set the Body CS properties for each Body CS on each bar from the data of the preceding tables:
  - a. Enter the Body CS origin position data for CG, CS1, and CS2 on each bar from the tables or from the corresponding MAT-file variables.
  - b. Set the **Translated from Origin of** menu entries for each Body CS on each bar according to the values in the tables.
4. Select the **Orientation** tab by clicking its tab:
  - a. Enter the **Orientation Vector** for the CG on each bar from the tables or from the corresponding MAT-file variables.
  - b. Choose World for **Relative CS** in each case.
  - c. Leave the other fields in their default values.

**Block Parameters: Bar1**

Body

Represents a user-defined rigid body. Body defined by mass  $m$ , inertia tensor  $I$ , and coordinate origins and axes for center of gravity (CG) and other user-specified Body coordinate systems. This dialog sets Body initial position and orientation, unless Body and/or connected Joints are actuated separately. This dialog also provides optional settings for customized body geometry and color.

Mass properties

Mass:  kg

Inertia:  kg\*m^2

Position | Orientation | Visualization

Show Port	Port Side	Name	Origin Position Vector [x y z]	Units	Translated from Origin of	Components in Axes of
<input type="checkbox"/>	Bottom	CG	[0.03 0.282 0]	m	CS1	World
<input checked="" type="checkbox"/>	Bottom	CS1	[0 0 0]	m	Adjoining	World
<input checked="" type="checkbox"/>	Top	CS2	[0.063 0.597 0]	m	CS1	World

OK Cancel Help Apply



**Block Parameters: Bar2**
✕

**Body**

Represents a user-defined rigid body. Body defined by mass  $m$ , inertia tensor  $I$ , and coordinate origins and axes for center of gravity (CG) and other user-specified Body coordinate systems. This dialog sets Body initial position and orientation, unless Body and/or connected Joints are actuated separately. This dialog also provides optional settings for customized body geometry and color.

**Mass properties**

Mass:  kg

Inertia:  kg\*m^2

Position   Orientation   Visualization

Show Port	Port Side	Name	Origin Position Vector [x y z]	Units	Translated from Origin of	Components in Axes of
<input type="checkbox"/>	Right	CG	[-0.427 -0.242 0]	m	CS1	World
<input checked="" type="checkbox"/>	Right	CS1	[0 0 0]	m	Adjoining	World
<input checked="" type="checkbox"/>	Left	CS2	[-0.87 -0.493 0]	m	CS1	World

OK

Cancel

Help

Apply

**Block Parameters: Bar3**
✕

**Body**

Represents a user-defined rigid body. Body defined by mass  $m$ , inertia tensor  $I$ , and coordinate origins and axes for center of gravity (CG) and other user-specified Body coordinate systems. This dialog sets Body initial position and orientation, unless Body and/or connected Joints are actuated separately. This dialog also provides optional settings for customized body geometry and color.

**Mass properties**

Mass:  kg

Inertia:  kg\*m^2

Position   Orientation   Visualization

Show Port	Port Side	Name	Origin Position Vector [x y z]	Units	Translated from Origin of	Components in Axes of
<input type="checkbox"/>	Top	CG	[-0.027 -0.048 0]	m	CS1	World
<input checked="" type="checkbox"/>	Top	CS1	[0 0 0]	m	Adjoining	World
<input checked="" type="checkbox"/>	Bottom	CS2	[0 0 0]	m	Adjoining	World

OK

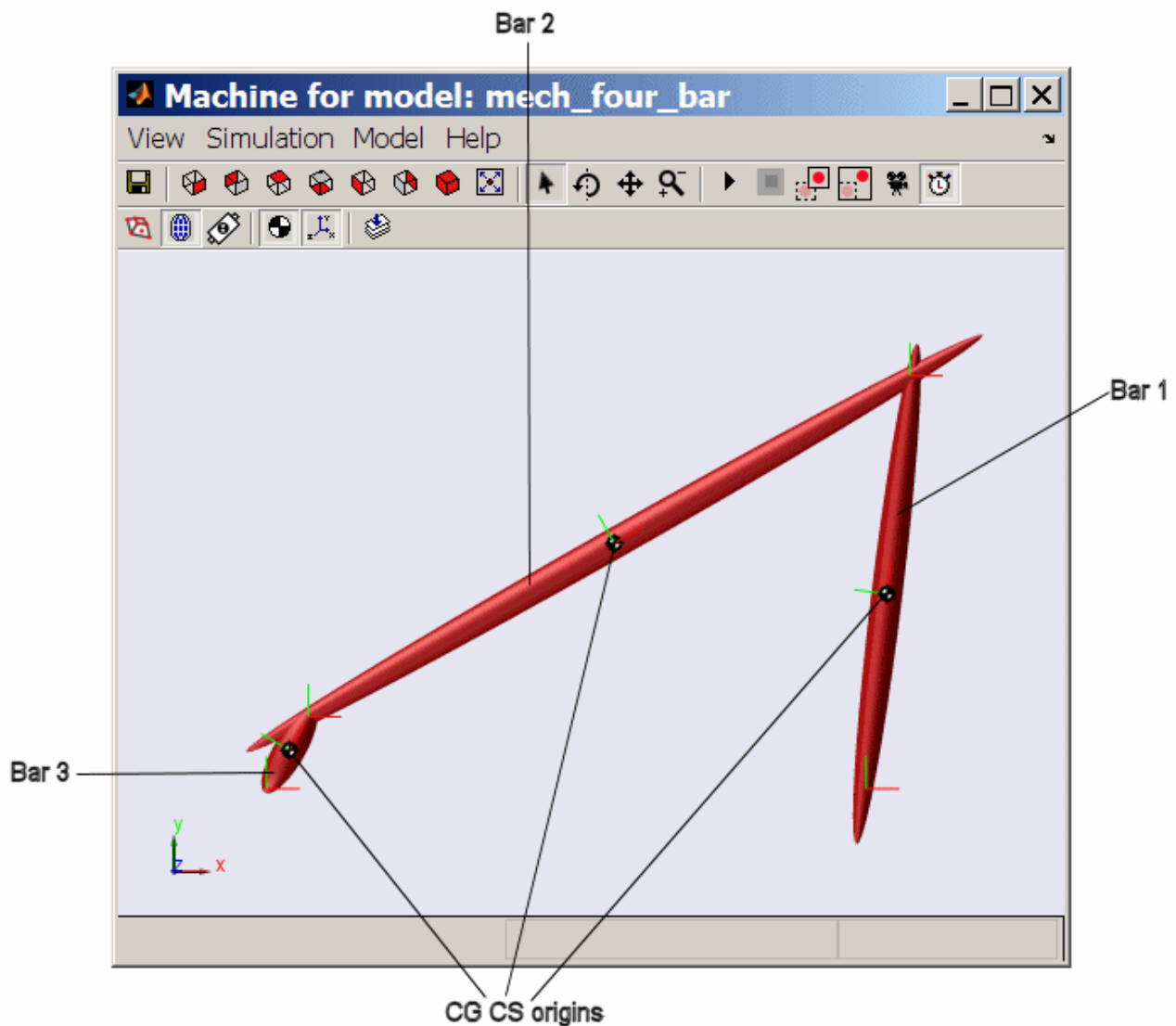
Cancel

Help

Apply

### Visualizing the Bodies

The front view of the four bar mechanism, with the bodies displayed as equivalent ellipsoids, looks like this:



### Sensing Motion and Running the Model

You finish building your model by setting initial conditions and inserting Sensors.


Before you start a simulation, you need to set its kinematic state or initial conditions. These include positions/angles and linear/angular velocities. This information, the machine's initial kinematic state, is discussed further in [Kinematics and the Machine's State of Motion](#) and [Applying Motions and Forces](#).


You can sense motion in any model in two basic ways: sensing bodies or sensing joints. Here you sense Joint motion, using Joint Sensor blocks and feeding their Simulink signal outputs to Scope blocks.


**Caution** Because they are immobile, ground points cannot be moved, nor do they have any motion to measure. Therefore, you cannot connect Ground blocks to Actuator or Sensor blocks.

### Connecting the Joint Sensors

To sense the motion of the Revolute2 and Revolute3 blocks,

1. From the Sensors & Actuators library, drag and drop two Joint Sensor blocks into the model window. Drag Joint Sensor next to Revolute2 and Joint Sensor1 next to Revolute3.
2. Before you can attach a Joint Sensor block to a Revolute block, you need to create a new open round connector port  on the Revolute. Open Revolute2's dialog box:
  - a. In the **Connection parameters** area in the middle, adjust the spinner menu **Number of sensor/actuator ports** to the value 1. Click **OK**.

A new connector port  appears on Revolute2.

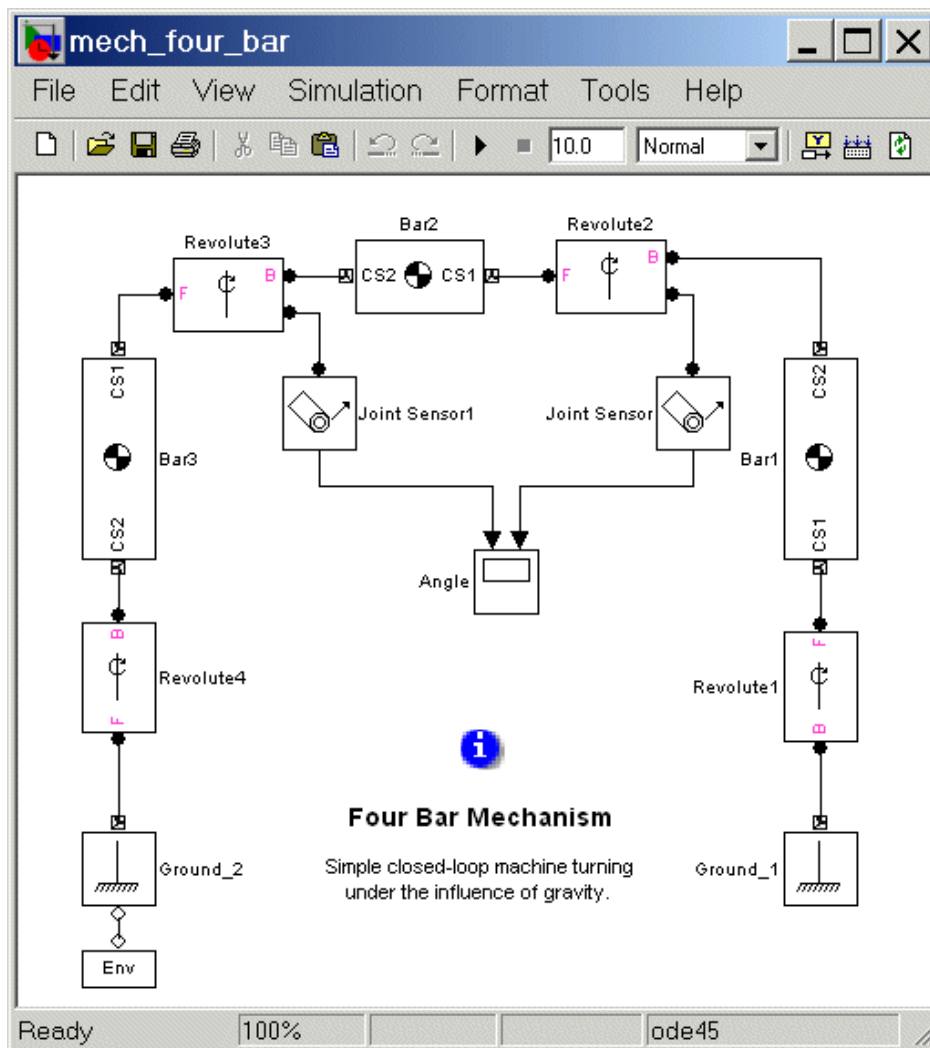
- b. Connect this connector port to the open round connector port on Joint Sensor.
3. Now repeat the same steps with Revolute3:
  - a. Create one new connector port  on Revolute 3.
  - b. Connect this port to Joint Sensor1.
4. Be sure to connect the outputs > of the Sensor blocks to a Simulink Sink block. These outputs are normal Simulink signals.

### Graphical Plot of Joint Motion with a Scope Block

Here you can view the Joint Sensor measurements of Revolute2 and Revolute3's motions using a Scope block from the Simulink Sinks library:

1. Open the Simulink Library Browser. From the Sinks library, drag and drop a Scope block into your model window in between Joint Sensor and Joint Sensor1 blocks. Rename the Scope block "Angle."
2. Open the Angle block. In this scope window's toolbar, open the **Parameters** box. Under **Axes**, reset **Number of axes** to 2. Click **OK**. A second inport > appears on the Angle block.
3. Expand the scope window for ease of viewing.
4. Connect the Joint Sensor and Joint Sensor1 block outputs > to the Angle block inports >.
5. Open Joint Sensor and Joint Sensor1:
  - a. In the **Measurements** area, **Connected to primitive** is set to R1 in both blocks, indicating the first and only primitive revolute inside Revolute2 and Revolute3 to which each Sensor can be connected.
  - b. Select the **Angle** check box to measure just the angle. Leave the units in default as deg (degrees). The Simulink line will contain one scalar.

Your completed model should look similar to the [mech\\_four\\_bar](#) demo model.



**Caution** Sensor and Actuator blocks are the *only* blocks that can connect SimMechanics blocks to normal Simulink blocks.

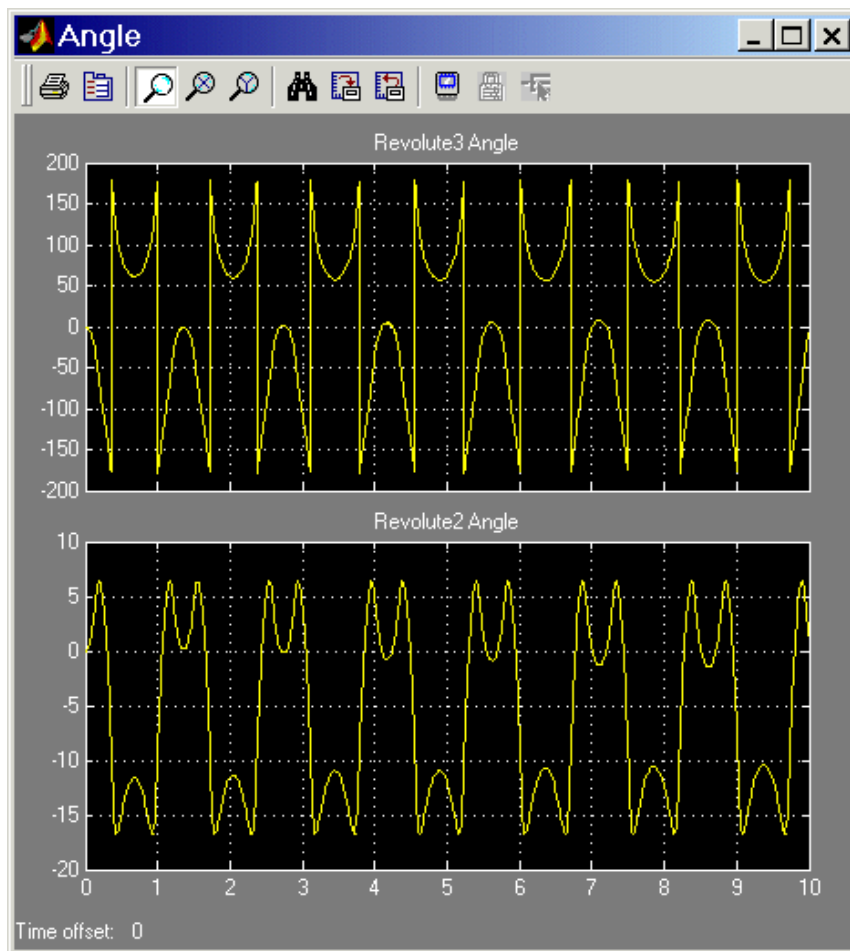
### Configuring and Running the Simulation

Now take the final steps to prepare and start the model:

1. In the model window **Simulation** menu, select **Configuration Parameters**:
  - a. In the **Solver** node, change **Absolute tolerance** to  $1e-6$ .
  - b. Leave the other defaults and click **OK**.
2. Now run the model by clicking **Start** in the Simulink toolbar. The four bar mechanism will fall under the influence of gravity.

Note some features of the simulation:

- In this example, the mechanism starts from rest, with the initial velocities at zero. Thus the initial state of the four bar system is just the geometric state that you set up in the immediately preceding section, [Setting Up the Block Diagram](#).
- The assembly at first falls over to the right, and the Revolute2 angle decreases.
- Bar3 turns all the way around, and Bar2 and Bar1 turn back to the left. The Revolute2 angle reverses direction. Revolute3 sweeps through a complete turn. Angles are mapped to the interval  $(-180^\circ, +180^\circ]$  and exhibit discontinuities.
- The motion repeats periodically, as there is no friction.



### Animation

If you leave your visualization window open at the time you start the simulation and select the **Animate machine during simulation** check box in the **SimMechanics** node of the Configuration Parameters dialog, the visualized machine moves in step with the simulation.

You can now compare the animated motion with the Scope plots of the Revolute2 and Revolute3 angles.

### Viewing a Four Bar Mechanism Animation

If you are connected to the Internet, have an AVI-compatible media streaming application installed on your system, and want to play a recorded animation of this system:

1. Click the following link. When the download dialog opens, choose **Save to file** and specify a file name and location on your system.
2. Click **OK** to save the AVI file to your system.
3. Once the downloading is complete, start the AVI animation on your system.

If you do not have an AVI-compatible application, consider using the MATLAB [VideoReader](#) class and its [read](#) method instead.

This is a compressed AVI recording, which requires that you have the Indeo 5 video codec installed to decompress and play.

[Download animation](#)

### For More About the Four Bar Mechanism

The four bar system is also discussed in the context of advanced SimMechanics features and methods: [Modeling Degrees of Freedom](#), [Validating Mechanical Models](#), [Finding Forces from Motions](#), [Trimming Mechanical Models](#), and [Linearizing Mechanical Models](#).

## Example — Modeling a DC Motor

### On this page...

[Overview of DC Motor Example](#)  
[Selecting Blocks to Represent System Components](#)  
[Building the Model](#)  
[Specifying Model Parameters](#)  
[Configuring the Solver Parameters](#)  
[Running the Simulation and Analyzing the Results](#)

### Overview of DC Motor Example

In this example, you model a DC motor driven by a constant input signal that approximates a pulse-width modulated signal and look at the current and rotational motion at the motor output.

To see the completed model, open the [Controlled DC Motor](#) demo.

### Selecting Blocks to Represent System Components

Select the blocks to represent the input signal, the DC motor, and the motor output displays.

The following table describes the role of the blocks that represent the system components.

Block	Description
<b>Solver Configuration</b>	Defines solver settings that apply to all physical modeling blocks.
<b>DC Voltage Source</b>	Generates a DC signal.
<b>Controlled PWM Voltage</b>	Generates the signal that approximates a pulse-width modulated motor input signal.
<b>H-Bridge</b>	Drives the DC motor.
<b>Current Sensor</b>	Converts the electrical current that drives the motor into a physical signal proportional to the current.
<b>Ideal Rotational Motion Sensor</b>	Converts the rotational motion of the motor into a physical signal proportional to the motion.
<b>DC Motor</b>	Converts input electrical signal into mechanical motion.
<b>PS-Simulink Converter</b>	Converts the input physical signal to a Simulink signal.
<b>Scope</b>	Displays motor current and rotational motion.
<b>Electrical Reference</b>	Provides the electrical ground.
<b>Mechanical Rotational Reference</b>	Provides the mechanical ground.

### Building the Model

Create a Simulink model, add blocks to the model, and connect the blocks.

1. Create a model.

If you are new to Simulink, see the [Creating a Simulink Model](#) example for information on how to create a model.

2. Add to the model the blocks listed in the following table. The Library column of the table specifies the hierarchical path to each block.

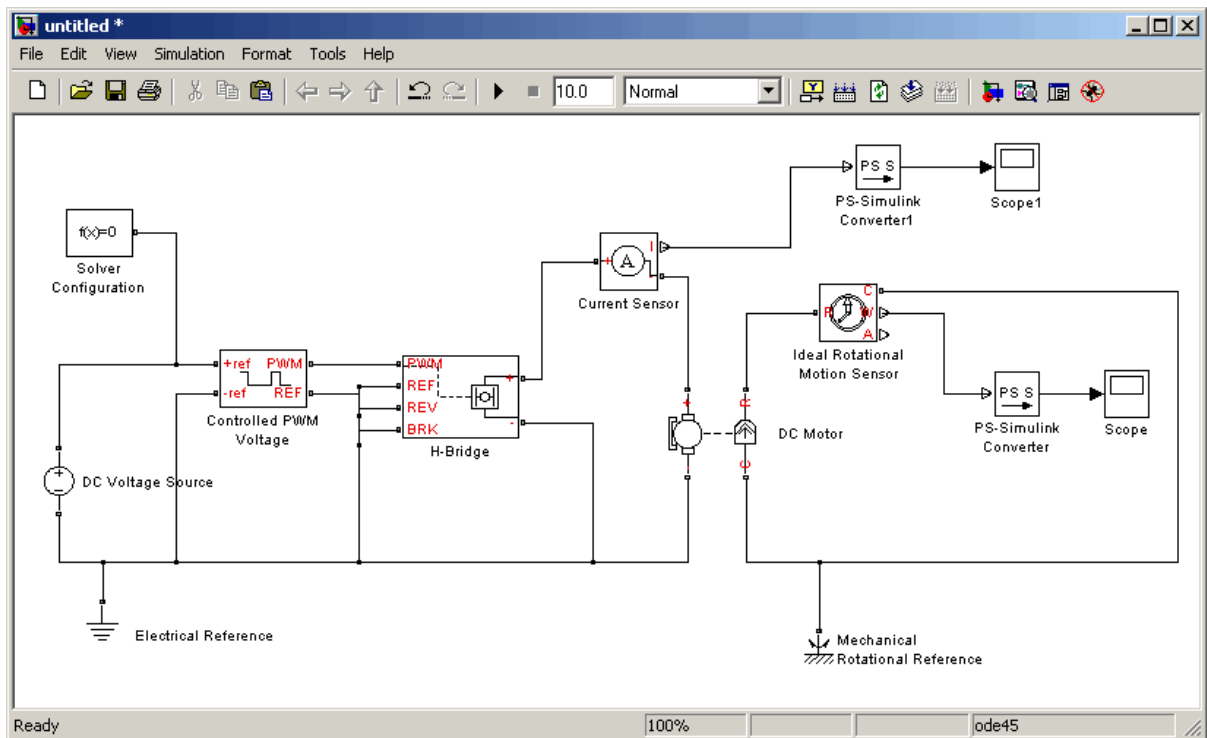
Block	Library Path	Quantity
<b>Solver Configuration</b>	<b>Simscape &gt; Utilities</b>	1
<b>DC Voltage Source</b>	<b>Simscape &gt; Foundation Library &gt; Electrical &gt; Electrical Sources</b>	1
<b>Controlled PWM Voltage</b>	<b>Simscape &gt; SimElectronics &gt; Actuators &amp; Drivers &gt; Drivers</b>	1
<b>H-Bridge</b>	<b>Simscape &gt; SimElectronics &gt; Actuators &amp; Drivers &gt; Drivers</b>	1
<b>Current Sensor</b>	<b>Simscape &gt; Foundation Library &gt; Electrical &gt; Electrical Sensors</b>	1
<b>Ideal Rotational Motion Sensor</b>	<b>Simscape &gt; Foundation Library &gt; Mechanical &gt; Mechanical Sensors</b>	1
<b>DC Motor</b>	<b>Simscape &gt; SimElectronics &gt; Actuators &amp; Drivers &gt; Rotational Actuators</b>	1
<b>PS-Simulink Converter</b>	<b>Simscape &gt; Utilities</b>	2
<b>Scope</b>	<b>Simulink &gt; Commonly Used Blocks</b>	2
<b>Electrical Reference</b>	<b>Simscape &gt; Foundation Library &gt; Electrical &gt; Electrical Elements</b>	1
<b>Mechanical Rotational Reference</b>	<b>Simscape &gt; Foundation Library &gt; Mechanical &gt; Rotational Elements</b>	1

**Note** You can use the Simscape function `ssc_new` with a domain type of `electrical` to create a Simscape model that contains the following blocks:

- Simulink-PS Converter
- PS-Simulink Converter
- Scope
- Solver Configuration
- Electrical Reference

This function also selects the Simulink ode15s solver.

3. Connect the blocks as shown in the following figure.



Now you are ready to specify block parameters.

### Specifying Model Parameters

Specify the following parameters to represent the behavior of the system components:

- [Model Setup Parameters](#)
- [Motor Input Signal Parameters](#)
- [Motor Parameters](#)
- [Current Display Parameters](#)
- [Torque Display Parameters](#)

### Model Setup Parameters

The following blocks specify model information that is not specific to a particular block:

- Solver Configuration
- Electrical Reference
- Mechanical Rotational Reference

As with Simscape models, you must include a Solver Configuration block in each topologically distinct physical network. This example has a single physical network, so use one Solver Configuration block with the default parameter values.

You must include an Electrical Reference block in each SimElectronics network. You must include a Mechanical Rotational Reference block in each network that includes electromechanical blocks. These blocks do not have any parameters.

For more information about using reference blocks, see [Grounding Rules](#) in the Simscape documentation.

### Motor Input Signal Parameters

You generate the motor input signal using three blocks:

- The DC Voltage Source block generates a constant signal.
- The Controlled PWM Voltage block generates a pulse-width modulated signal.
- The H-Bridge block drives the motor.

In this example, all input ports of the H-Bridge block except the PWM port are connected to ground. As a result, the H-Bridge block behaves as follows:

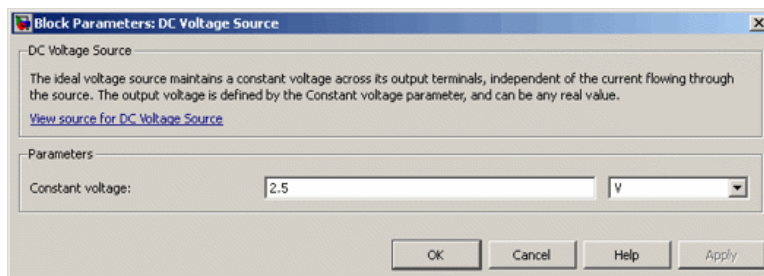
- When the motor is on, the H-Bridge block connects the motor terminals to the power supply.
- When the motor is off, the H-Bridge block acts as a freewheeling diode to maintain the motor current.

In this example, you simulate the motor with a constant current whose value is the average value of the PWM signal. By using this type of signal, you set

up a fast simulation that estimates the motor behavior.

1. Set the DC Voltage Source block parameters as follows:

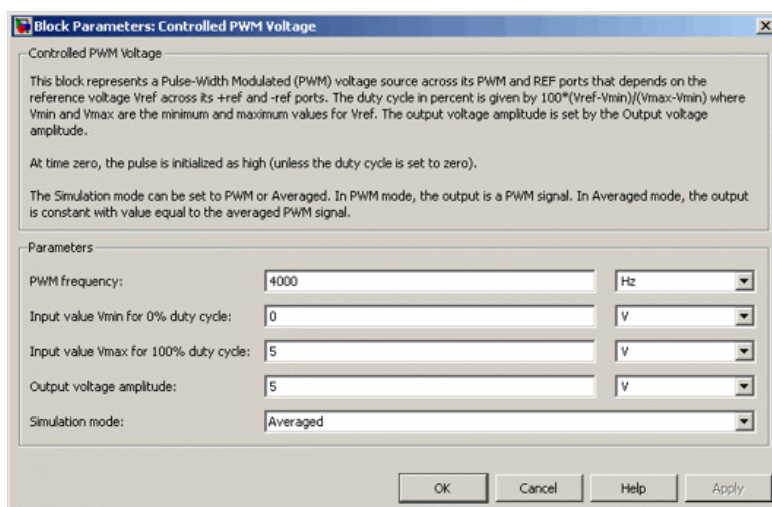
- **Constant voltage** = 2.5



2. Set the Controlled PWM Voltage block parameters as follows:

- **PWM frequency** = 4000
- **Simulation mode** = Averaged

This value tells the block to generate an output signal whose value is the average value of the PWM signal. Simulating the motor with an averaged signal estimates the motor behavior in the presence of a PWM signal. To validate this approximation, use value of PWM for this parameter.

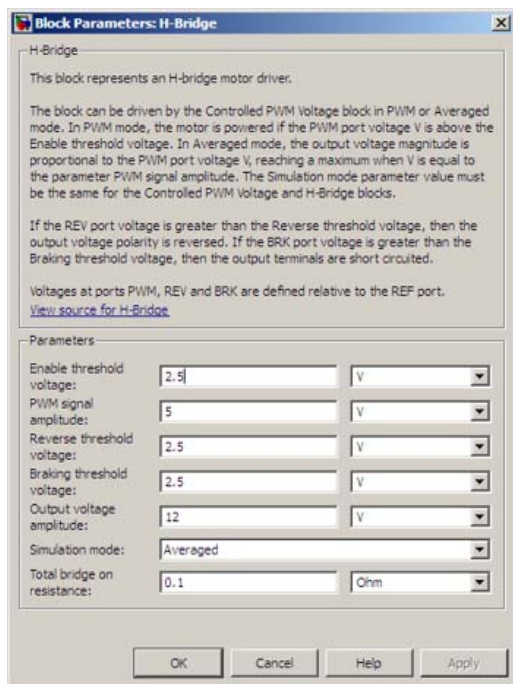


3. Set the H-Bridge block parameters as follows:

- **Simulation mode** = Averaged

This value tells the block to generate an output signal whose value is the average value of the PWM signal. Simulating the motor with an averaged signal estimates the motor behavior in the presence of a PWM signal. To validate this approximation, use value of PWM for this parameter to validate this approximation.





### Motor Parameters

Configure the block that models the motor.

Set the Motor block parameters as follows, leaving the unit settings at their default values where applicable:

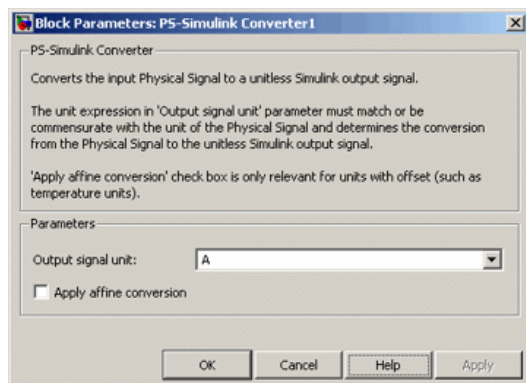
- **Electrical Torque** tab:
  - **Model parameterization** = By rated power, rated speed & no-load speed
  - **Armature inductance** = 0.01
  - **No-load speed** = 4000
  - **Rated speed (at rated load)** = 2500
  - **Rated load (mechanical power)** = 10
  - **Rated DC supply voltage** = 12
- **Mechanical** tab:
  - **Rotor inertia** = 2000
  - **Rotor damping** = 1e-06

### Current Display Parameters

Specify the parameters of the blocks that create the motor current display:

- Current Sensor block
- PS-Simulink Converter1 block
- Scope1 block

Of the three blocks, only the PS-Simulink Converter1 block has parameters. Set the PS-Simulink Converter1 block **Output signal unit** parameter to A to indicate that the block input signal has units of amperes.



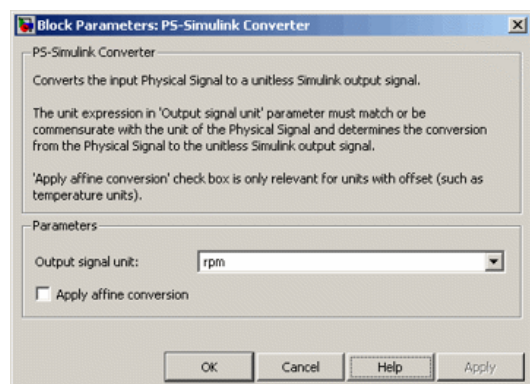
### Torque Display Parameters

Specify the parameters of the blocks that create the motor torque display:

- Ideal Rotational Motion Sensor block
- PS-Simulink Converter block
- Scope block

Of the three blocks, only the PS-Simulink Converter block has parameters you need to configure for this example. Set the PS-Simulink Converter block **Output signal unit** parameter to  $\text{rpm}$  to indicate that the block input signal has units of revolutions per minute.

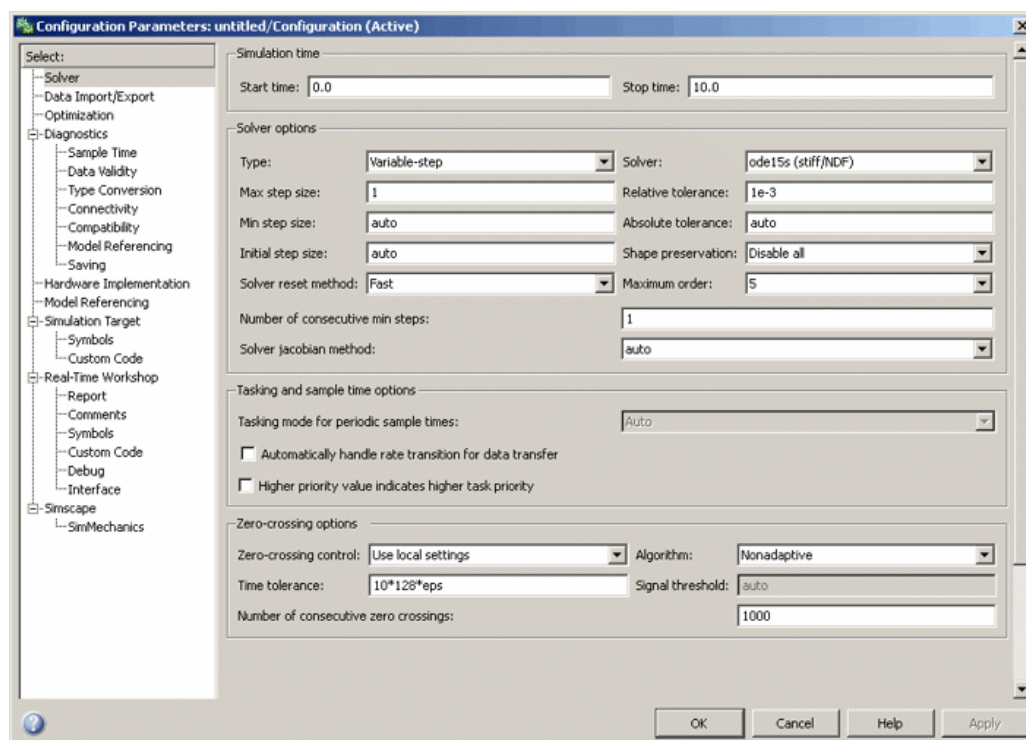
**Note** You must type this parameter value. It is not available in the drop-down list.



### Configuring the Solver Parameters

Configure the solver parameters to use a continuous-time solver because SimElectronics models only run with a continuous-time solver. Increase the maximum step size the solver can take so the simulation runs faster.

1. In the model window, select **Simulation** > **Configuration Parameters** to open the **Configuration Parameters** dialog box.
2. Select **ode15s** (Stiff/NDF) from the **Solver** list.
3. Enter 1 for the **Max step size** parameter value.
4. Click **OK**.



For more information about configuring solver parameters, see [Simulating an Electronic System](#).

### Running the Simulation and Analyzing the Results

In this part of the example, you run the simulation and plot the results.

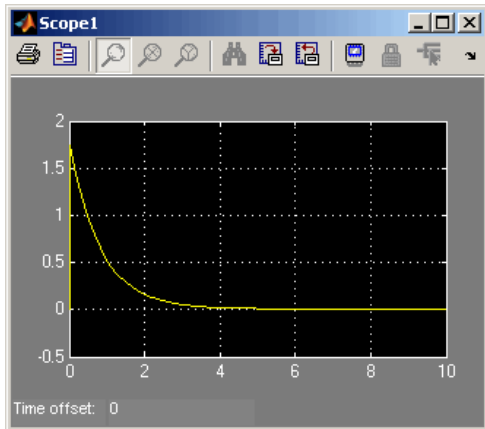
In the model window, select **Simulation** > **Start** to run the simulation.

To view the motor current and torque in the Scope windows, double-click the Scope blocks. You can do this before or after you run the simulation.

**Note** By default, the scope displays appear stacked on top of each other on the screen, so you can only see one of them. Click and drag the windows to reposition them.

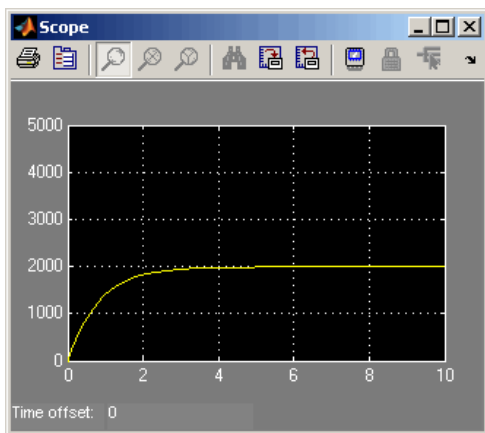
The following plot shows the motor current.

#### Motor Current



The next plot shows the motor rpm.

#### Motor RPM



As expected, the motor runs at about 2000 rpm when the applied DC voltage is 2.5 V.

The below mechanism is a Whitworth mechanism which is also known as quick return linkage. It consists of 6 part (5 body+ground). Try to model the mechanism in SimMechanism and run the model by applying torque on the joint A. The coordinates of hard points are provided in the below table. Mass of all parts is 1kg. Use the default values for inertia matrices.

Point	X (mm)	Y (mm)	Z (mm)
A	0	20	0
B	60.795	43	0
C	0	-65	0
D	-42.219	-140	0
E	172.781	-140	0
CG of part a	30.397	31.5	0
CG of part b	23.023	-24.101	0
CG of part c	65.281	-140	0

