

# Simulink<sup>®</sup> Control Design

**For Use with Simulink<sup>®</sup>**

- Modeling
- Simulation
- Implementation

Getting Started

*Version 1*



## How to Contact The MathWorks:



www.mathworks.com      Web  
comp.soft-sys.matlab      Newsgroup



support@mathworks.com      Technical Support  
suggest@mathworks.com      Product enhancement suggestions  
bugs@mathworks.com      Bug reports  
doc@mathworks.com      Documentation error reports  
service@mathworks.com      Order status, license renewals, passcodes  
info@mathworks.com      Sales, pricing, and general information



508-647-7000      Phone



508-647-7001      Fax



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

For contact information about worldwide offices, see the MathWorks Web site.

### *Getting Started with Simulink Control Design*

© COPYRIGHT 2004–2005 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, Simulink, Stateflow, Handle Graphics, Real-Time Workshop, and xPC TargetBox are registered trademarks of The MathWorks, Inc.

Other product or brand names are trademarks or registered trademarks of their respective holders.

### **Patents**

The 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**

June 2004	Online only	New for Version 1.0 (Release 14)
October 2004	Online only	Revised for Version 1.1 (Release 14SP1)
March 2005	Online only	Revised for Version 1.2 (Release 14SP2)
September 2005	Online only	Revised for Version 1.3 (Release 14SP3)



## Introduction

**1**

<b>What Is Simulink Control Design?</b> .....	<b>1-2</b>
<b>Purpose of Linearization</b> .....	<b>1-3</b>
<b>Using the GUI vs. Command Line Functions</b> .....	<b>1-4</b>
<b>Using the Documentation</b> .....	<b>1-5</b>
Expected Background .....	<b>1-5</b>
How to Use This Guide .....	<b>1-5</b>
Online Documentation .....	<b>1-5</b>
Using Examples and Demos .....	<b>1-5</b>
Related Products .....	<b>1-6</b>
<b>Quick Start</b> .....	<b>1-7</b>

## What Is Linearization?

**2**

<b>Understanding Linearization</b> .....	<b>2-2</b>
Operating Points .....	<b>2-2</b>
Linearization of Nonlinear Models .....	<b>2-4</b>
Linearization of Discrete-Time Models .....	<b>2-6</b>
Linearization of Multi-Rate Models .....	<b>2-8</b>
Linearization of Simulink Models .....	<b>2-8</b>
<b>Understanding Open Loop Analysis</b> .....	<b>2-10</b>

<b>Overview</b> .....	<b>3-3</b>
<b>Example: The Magnetic Ball</b> .....	<b>3-4</b>
Magnetic Ball System .....	3-4
Model Equations .....	3-4
<b>Creating or Opening a Simulink Model</b> .....	<b>3-6</b>
<b>Beginning a New Linearization Project</b> .....	<b>3-9</b>
<b>Configuring Inputs and Outputs for the Linearized Model</b> .....	<b>3-12</b>
Inserting Linearization Points .....	3-13
Removing Linearization Points .....	3-16
Performing Open Loop Analysis .....	3-16
Inspecting Analysis I/Os .....	3-18
<b>Specifying Operating Points</b> .....	<b>3-20</b>
Computing Operating Points from Specifications .....	3-21
Specifying Completely Known Operating Points .....	3-26
Extracting Operating Points from Simulation .....	3-28
Importing Operating Points .....	3-31
Copying Operating Points .....	3-32
Exporting Operating Points .....	3-33
<b>Linearizing the Model</b> .....	<b>3-35</b>
Using the LTI Viewer .....	3-37
Linearizing Discrete-Time and Multi-Rate Models .....	3-38
<b>Linearizing a Block</b> .....	<b>3-39</b>
<b>Analyzing the Results</b> .....	<b>3-41</b>
Validating the Linearization .....	3-41
Displaying the Results .....	3-46
<b>Exporting and Saving Your Work</b> .....	<b>3-50</b>

Exporting Results .....	3-50
Exporting and Restoring Linearization I/O Settings .....	3-51
Saving Control and Estimation Tools Manager Projects ..	3-51
Opening Control and Estimation Tools Manager Projects .....	3-52

## Linearizing Models Using Functions

# 4

<b>Overview</b> .....	4-2
<b>Example: Water-Tank System</b> .....	4-3
The Water-Tank System .....	4-3
Model Equations .....	4-3
<b>Creating or Opening a Simulink Model</b> .....	4-5
<b>Configuring the Model for Linearization</b> .....	4-7
Choosing and Storing Linearization Points .....	4-7
Extracting Linearization Points from a Model .....	4-9
Editing an I/O Object .....	4-10
Open Loop Analysis Using Functions .....	4-12
<b>Specifying Operating Points Using Functions</b> .....	4-13
Computing Operating Points from Specifications .....	4-13
Specifying Completely Known Operating Points .....	4-19
Extracting Values from Simulation .....	4-21
Using Structures and Vectors of Operating Point Values ..	4-21
<b>Linearizing the Model</b> .....	4-24
Linearizing Discrete-Time and Multi-Rate Models .....	4-25
<b>Analyzing the Results</b> .....	4-27
Using the LTI Viewer .....	4-27
Saving Your Work .....	4-29
Restoring Linearization I/O Settings .....	4-29





# Introduction

---

What Is Simulink Control Design?  
(p. 1-2)

Introduction to the key features of  
Simulink® Control Design

Purpose of Linearization (p. 1-3)

Simplify system design and analysis  
using linearized models

Using the GUI vs. Command Line  
Functions (p. 1-4)

Compare features of the graphical  
and command-line interfaces

Using the Documentation (p. 1-5)

Expected background, how to use  
this guide, and other documentation  
sources

Quick Start (p. 1-7)

Brief outline of the linearization  
process using Simulink Control  
Design

## What Is Simulink Control Design?

Simulink® Control Design provides tools for linearization of control systems and physical models in Simulink. Linearized Simulink models often simplify system analysis and compensator design. This is useful in many industries and applications, including

- Aerospace: flight control, guidance, navigation
- Automotive: cruise control, emissions control, transmission
- Equipment manufacturing: motors, disk drives, servos

Simulink Control Design builds on features of the Control System Toolbox, such as LTI Viewer and SISO Design Tool, as well as the Simulink linearization engine. Use Simulink Control Design to

- Specify input and output points without inserting blocks in the model (as was the case in previous versions of the Control System Toolbox)
- Perform open loop analysis without deleting feedback loops
- Extract operating points from a simulation
- Compute operating points from implicitly defined state and output values (also known as trimming)
- Inspect and debug results of the linearization
- Export results to the MATLAB® workspace
- Manage linearization projects
- Use new linearization and trim functions

## Purpose of Linearization

Many common control system analysis and design methodologies require linear, time-invariant models. However, control systems and physical models created with Simulink are often nonlinear and time-varying.

Linearization is the approximation of a nonlinear system as a linear system, based on the assumption that the system is almost linear within a certain range of operation. With a linearized model you can

- Use the Control System Toolbox LTI Viewer to display and analyze dynamic behaviors of a model
- Tune feedback gains and design compensators for a system
- Express a model as a transfer function, state space model, or zero-pole-gain model
- Determine the response of a model to arbitrary input signals

A linearized model can provide a good approximation to a nonlinear system when created and used carefully. Factors affecting the accuracy of the approximation addressed in the next chapter are

- Choice of operating points. See “Operating Points” on page 2-2.
- Understanding the equations for the linearized model. See “Linearization of Nonlinear Models” on page 2-4.
- Controlling the effect of feedback loops. See “Understanding Open Loop Analysis” on page 2-10.

## Using the GUI vs. Command Line Functions

The Simulink Control Design GUI provides a graphical environment for control system linearization and design. Using it, you can easily inspect and analyze operating points and results of linearization. In addition, you can save and restore settings as well as export results to the MATLAB workspace.

You can also linearize models using the command line functions. With the functions, you can automate many of your linearization projects and perform *batch linearization*, such as linearization of a system at several different values of a parameter.

# Using the Documentation

## Expected Background

Users of this guide should be familiar with control systems design and analysis, and have experience creating Simulink models. Familiarity with the Control System Toolbox is also desirable.

## How to Use This Guide

**To get started quickly with Simulink Control Design**, read “Quick Start” on page 1-7 for a brief overview of the linearization process.

**If you are new to linearization**, begin with Chapter 2, “What Is Linearization?” This introduces linearization concepts that are important for accurate creation and use of linearized models.

**All users** should read Chapter 3, “Linearizing Models” which describes features and use of the Simulink Control Design GUI.

**To automate the linearization process**, or perform batch linearization, continue with Chapter 4, “Linearizing Models Using Functions”

## Online Documentation

Further documentation is available online or in the Help browser, including the following chapters: “Understanding and Controlling Results”, “Functions — Alphabetical List”, and “Blocks — Alphabetical List”.

## Using Examples and Demos

The Simulink Control Design documentation makes use of several examples. You can access these examples by typing

```
demo
```

at the MATLAB prompt and selecting **Simulink Control Design** under the **Simulink** node. You can also access the examples by clicking the **Demos** tab in the Help browser.

## **Related Products**

The MathWorks provides several products that are especially relevant to the kinds of tasks you can perform with Simulink Control Design. For more information about these products, visit the MathWorks Web site, at <http://www.mathworks.com/products/simcontrol/>.

## Quick Start

To starting working with Simulink Control Design quickly, follow these steps. The documentation gives additional information for each step in the process and describes the advanced functionality of the product.

- 1** Create or open a Simulink model and decide which portion of the model you need to linearize. See “Creating or Opening a Simulink Model” on page 3-6 for more information.
- 2** Open a linearization project for this model in the Control and Estimation Tools Manager by selecting **Tools > Control Design > Linear Analysis** from the model window. See “Beginning a New Linearization Project” on page 3-9 for more information.
- 3** Select inputs and outputs for the linearized model by inserting linearization points on the model diagram. To do this, right-click on a signal line and select **Linearization Points**, and then select either **Input Point** or **Output Point** depending on whether you want the signal to be an input or output for the linearized model. To remove the effect of feedback loops from the linearized model, you can artificially open a loop by right-clicking on the signal line and selecting **Linearization Points > Open Loop**. See “Configuring Inputs and Outputs for the Linearized Model” on page 3-12 for more information.
- 4** Specify the operating point for the linearized model in the **Operating Points** node of the Controls and Estimation Tools Manager. To compute a steady-state operating point from state specifications, select the **Create Operating Points** panel and enter the specifications into the table, then click **Compute Operating Point**. See “Specifying Operating Points” on page 3-20 for more information and for methods of specifying alternative operating points.
- 5** Linearize the model by first selecting the **Operating Points** panel within the **Linearization Task** node and then selecting the appropriate operating point from the list. To compute the linearized model, click the **Linearize Model** button. See “Linearizing the Model” on page 3-35 for more information.
- 6** Analyze the results with the **Linearization Summary** panel and the **Linearization Inspector** panel within the **Model** node. Create plots of

the results with the **Custom Views** node. See “Analyzing the Results” on page 3-41 for more information.

- 7 Export the linearized model to the MATLAB workspace by right-clicking on the **Model** node and selecting **Export**. See “Exporting and Saving Your Work” on page 3-50 for more information.



# What Is Linearization?

---

Engineers often use linearization in the design and analysis of control systems and physical models. Successful linearization depends on an understanding of important linearization concepts and factors affecting linearization, discussed here.

Understanding Linearization (p. 2-2)	Factors affecting linearization of Simulink models
Understanding Open Loop Analysis (p. 2-10)	Effects of feedback loops in linearization

## Understanding Linearization

Understanding the process of linearization with Simulink Control Design depends on first understanding several concepts such as operating points (define the point at which the linearization takes place), the relationship between linearization theory and linearization of Simulink models, and the effects of feedback loops. The remainder of this chapter discusses these concepts in the following sections.

- “Operating Points” on page 2-2
- “Linearization of Nonlinear Models” on page 2-4
- “Linearization of Discrete-Time Models” on page 2-6
- “Linearization of Multi-Rate Models” on page 2-8
- “Linearization of Simulink Models” on page 2-8
- “Understanding Open Loop Analysis” on page 2-10

Subsequent chapters give information on linearization using Simulink Control Design.

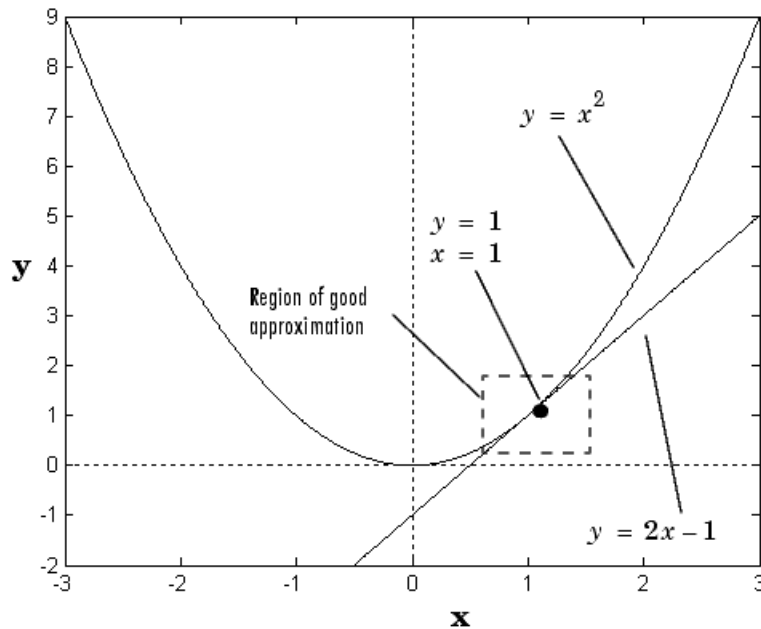
### Operating Points

The operating point of a dynamic system defines its overall *state* at a given time. For example, in a model of a car engine, variables such as engine speed, throttle angle, engine temperature, and surrounding atmospheric conditions typically describe the operating point. The level of the operating point affects the system’s behavior. For example, the behavior of a car engine can vary greatly when it operates at high or low elevations.

A linearized model is an approximation that is valid in a small region around the operating point of the system. Near the operating point the approximation will be good, while far away it will be poor. A linearized model of a car being operated at 3000 ft. will be very accurate at elevations close to 3000 ft. but less accurate as the car travels higher or lower.

The following figure shows a nonlinear function,  $y = x^2$ , and a linear function,  $y = 2x - 1$ . The linear function is an approximation to the nonlinear

function about the operating point  $x=1, y=1$ . Near this operating point, the approximation is good. Away from this operating point, the approximation is poor. The precise boundaries of this region are often somewhat arbitrary. The following figure shows a possible region of good approximation for the linearization of  $y = x^2$ .



### Linear Approximation

When creating a linearized model from a Simulink model, the choice of operating point is important as it will determine the accuracy of the approximation. Choose an operating point that is very close to the expected operating values of the system. One way to do this is with an equilibrium operating point, described in the next section.

### Equilibrium Operating Points

The equilibrium operating point remains steady and constant with time; all states in the model are at equilibrium. It is also known as a steady state or

trimmed operating point. For example, a car operating on cruise control on a flat road maintains a constant speed. Its operating point is steady, or at equilibrium, although a controller stabilizes the system.

A hanging pendulum provides an example of a stable equilibrium operating point. When the pendulum hangs straight down its position does not change with time since it is at an equilibrium position. When its position deviates slightly from this position, it always returns to the equilibrium; small changes in the operating point do not cause the system to leave the region of good approximation around the equilibrium value.

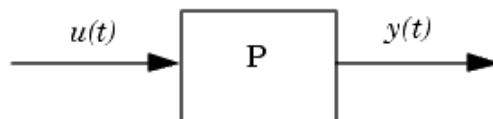
An example of an unstable equilibrium operating point is a pendulum that points upwards. As long as the pendulum points *exactly* upwards, it is steady at this equilibrium state. However, when the pendulum deviates slightly from this state, it swings downwards and the operating point leaves the region around the equilibrium value.

A model linearized about a stable equilibrium operating point is likely to remain within the region around the equilibrium value. This model will give a good approximation to the nonlinear model. A model linearized about an unstable equilibrium operating point is not likely to remain within the region around the equilibrium value. This model gives a poor approximation to the nonlinear model once it deviates from this equilibrium point, unless a controller is designed to stabilize the system. The magnetic ball system in gives an example of an unstable equilibrium that is stabilized by a controller.

Simulink Control Design provides several methods for specifying the operating point of a model. “Specifying Operating Points” on page 3-20, and “Specifying Operating Points Using Functions” on page 4-13 describe these methods.

### **Linearization of Nonlinear Models**

The following figure shows a block diagram consisting of an external input signal,  $u(t)$ , a measured output signal,  $y(t)$ , and a nonlinear system that describes the system’s states and its dynamic behavior,  $P$ .



### Block Diagram

The block diagram is a visual representation of a nonlinear system. You can also express the system in terms of the state space equations

$$\dot{x}(t) = f(x(t), u(t), t)$$

$$y(t) = g(x(t), u(t), t)$$

where  $x(t)$  represents the system's states,  $u(t)$  represents the inputs, and  $y(t)$  represents the outputs. In these equations, the variables vary continuously with time. Discrete-time and multi-rate models are discussed in the section "Linearization of Discrete-Time Models" on page 2-6 and "Linearization of Multi-Rate Models" on page 2-8.

A linear time-invariant approximation to this system is valid in a region around the operating point at  $t=t_0$ ,  $x(t_0)=x_0$ , and  $u(t_0)=u_0$ . If the values of the system's states,  $x(t)$  and inputs,  $u(t)$  are close enough to the operating point, the system will behave approximately linearly.

To describe the linearized model, it helps to first define a new set of variables centered about the operating point of the states, inputs, and outputs:

$$\delta x(t) = x(t) - x_0$$

$$\delta u(t) = u(t) - u_0$$

$$\delta y(t) = y(t) - y_0$$

The value of the outputs at the operating point is given by  $y(t_0)=g(x_0, u_0, t_0)=y_0$ .

---

**Note** When comparing a linearized model with the original model, remember that the convention used in this book is to write the linearized model in terms of  $\delta x$ ,  $\delta u$ , and  $\delta y$ . The value of each of these variables at the operating point is zero.

---

The linearized state space equations written in terms of  $\delta x(t)$ ,  $\delta u(t)$ , and  $\delta y(t)$  are

$$\begin{aligned}\delta \dot{x}(t) &= A\delta x(t) + B\delta u(t) \\ \delta y(t) &= C\delta x(t) + D\delta u(t)\end{aligned}$$

where  $A$ ,  $B$ ,  $C$ , and  $D$  are constant coefficient matrices. These matrices are defined as the Jacobians of the system, evaluated at the operating point

$$\begin{aligned}A &= \left. \frac{\partial f}{\partial x} \right|_{t_0, x_0, u_0} & B &= \left. \frac{\partial f}{\partial u} \right|_{t_0, x_0, u_0} \\ C &= \left. \frac{\partial g}{\partial x} \right|_{t_0, x_0, u_0} & D &= \left. \frac{\partial g}{\partial u} \right|_{t_0, x_0, u_0}\end{aligned}$$

The transfer function of the linearized model can be used in place of the system,  $P$ , in the previous figure. To find the transfer function, divide the Laplace transform of  $\delta y(t)$  by the Laplace transform of  $\delta u(t)$ :

$$P_{lin}(s) = \frac{\delta Y(s)}{\delta U(s)}$$

## Linearization of Discrete-Time Models

Discrete-time models are similar to continuous models, discussed in the previous section, with the exception that the values of system variables change at discrete times,  $t_k$ , where  $k$  is an integer value. The state-space equations for a nonlinear, discrete-time system are

$$x_{k+1} = f(x_k, u_k, t_k)$$

$$y_k = g(x_k, u_k, t_k)$$

A linear time-invariant approximation to this system is valid in a region around the operating point

$$t_k = t_{k_0}, x_k = x_{k_0}, u_k = u_{k_0}, \text{ and } y_k = g(x_{k_0}, u_{k_0}, t_{k_0}) = y_{k_0}$$

If the values of the system's states,  $x_k$ , inputs,  $u_k$ , and outputs,  $y_k$ , are close enough to the operating point, the system will behave approximately linearly. As with continuous time systems it is helpful to define variables centered about the operating point values

$$\delta x_k = x_k - x_{k_0}$$

$$\delta u_k = u_k - u_{k_0}$$

$$\delta y_k = y_k - y_{k_0}$$

where the value of the outputs at the operating point are defined as:

$$y_{k_0} = g(x_{k_0}, u_{k_0}, t_{k_0})$$

The linearized state-space equations can then be written in terms of these new variables

$$\delta x_{k+1} \approx A \delta x_k + B \delta u_k$$

$$\delta y_k \approx C \delta x_k + D \delta u_k$$

where  $A$ ,  $B$ ,  $C$ , and  $D$  are given by

$$A = \left. \frac{\partial f}{\partial x_k} \right|_{t_0, x_0, u_0} \quad B = \left. \frac{\partial f}{\partial u_k} \right|_{t_0, x_0, u_0}$$

$$C = \left. \frac{\partial g}{\partial x_k} \right|_{t_0, x_0, u_0} \quad D = \left. \frac{\partial g}{\partial u_k} \right|_{t_0, x_0, u_0}$$

## Linearization of Multi-Rate Models

Multi-rate models involve states with various sampling rates. This means that the state variables change values at different times and with different frequencies, with some variables possibly changing continuously. The general state-space equations for a nonlinear, multi-rate system are

$$\begin{aligned} \dot{x}(t) &= f(x(t), x_1(k_1), \dots, x_m(k_m), u(t), t) \\ x_1(k_1 + 1) &= f_1(x(t), x_1(k_1), \dots, x_m(k_m), u(t), t) \\ &\vdots \\ x_m(k_m + 1) &= f_m(x(t), x_1(k_1), \dots, x_m(k_m), u(t), t) \\ y(t) &= g(x(t), x_1(k_1), \dots, x_m(k_m), u(t), t) \end{aligned}$$

where  $k_1, \dots, k_m$  are integer values and  $t_{k_1}, \dots, t_{k_m}$  are discrete times. The linearized equations will approximate this system as a single-rate discrete model:

$$\begin{aligned} \delta x_{k+1} &\approx A \delta x_k + B \delta u_k \\ \delta y_k &\approx C \delta x_k + D \delta u_k \end{aligned}$$

## Linearization of Simulink Models

You apply the linearization concepts when linearizing Simulink models with Simulink Control design. Simulink uses a series of connected *blocks* to model physical systems and control systems. Input and output signals connect the blocks, which represent mathematical operations. The nonlinear system, P, in the previous figure, now represents a series of connected Simulink blocks.

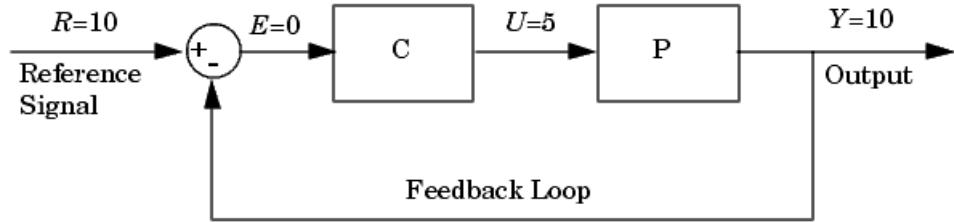
Simulink linearizes both continuous and discrete-time nonlinear systems by computing the state-space matrices,  $A$ ,  $B$ ,  $C$ , and  $D$ , using one of the linearization algorithms described in “Understanding and Controlling Results” in the online documentation. A common method is to perturb the inputs and states by a small amount about their operating point values. The response to these perturbations is measured in the outputs. Both the perturbations and the response are used to compute the matrices. Refer to “Understanding and Controlling Results” in the online documentation for more information.



To linearize models with Simulink Control Design, you must insert linearization input and output points in the model diagram. A linearization input point defines an input to the linearized model while a linearization output point defines an output of the linearized model. Additionally, when the linearized models are computed using numerical perturbation, an input point is the point on the diagram where the small perturbation to the input signal is introduced and an output point is the point on the diagram where the response to this perturbation is measured. “Inserting Linearization Points” on page 3-13 gives methods for selecting linearization input and output points with Simulink Control Design.

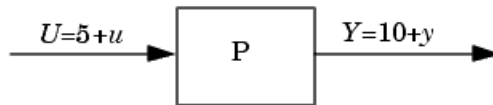
## Understanding Open Loop Analysis

Many control systems contain feedback loops. An example of such a system is shown in this figure.



### Control System with Feedback Loop

The model represented in this figure is at equilibrium. Consider linearizing the plant,  $P$ , about this equilibrium operating point by changing the input signal,  $U$ , by a small amount,  $u$ , and measuring the change in the output signal,  $y$ . The portion of the system that you want to linearize is shown in the following figure.

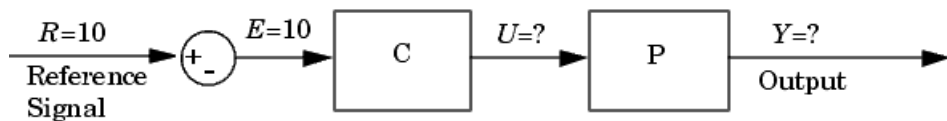


### Portion of System for Linearization

However, due to the presence of the feedback loop, the change in the output signal will feed back into the controller,  $C$ , and then into the plant. This affects the behavior of the system you are linearizing. In fact, if  $C$  and  $P$  were linear,

the linearized model between  $U$  and  $Y$  would be  $\frac{P(s)}{1+C(s)P(s)}$  rather than  $P(s)$ .

You could manually remove the feedback signal from the model in an attempt to resolve this issue. However, as shown in the following figure, this changes the operating point of the system since the error signal,  $E$ , is now equal to the reference signal,  $R$ . Linearizing about this new operating point would change the linearization results. Of course, this only makes a difference for nonlinear models. When the model is already linear, it has the same form regardless of the operating point.



### Control System with Feedback Signal Removed

When linearizing Simulink models, use Simulink Control Design's capability to label an input or output point as open loop. Doing so ensures that the output signal is not fed back into the model but keeps the operating point the same. In other words, in the linear case, you would compute  $P(s)$  rather

than  $\frac{P(s)}{1+C(s)P(s)}$ . "Performing Open Loop Analysis" on page 3-16 and "Open Loop Analysis Using Functions" on page 4-12 give methods for assigning open loop points in Simulink models using the Simulink Control Design GUI and functions respectively.



# Linearizing Models

---

Use the Simulink Control Design Graphical User Interface (GUI) to linearize Simulink models. With this tool you can insert, inspect, and change input and output points for linearization without adding blocks to the model diagram, perform open loop analysis without manually breaking signal lines, and specify or compute operating points for linearization. Additionally, use the GUI to inspect operating points and results of the linearization (block-by-block), export them to the MATLAB workspace, and display them in the LTI Viewer. You can also save, reload, and compare linearization projects using alternative operating points and parameter values.

Overview (p. 3-3)	An introduction to the linearization process
Example: The Magnetic Ball (p. 3-4)	Derivation of a nonlinear system that illustrates features of Simulink Control Design
Creating or Opening a Simulink Model (p. 3-6)	Opening the Simulink model of the magnetic ball example
Beginning a New Linearization Project (p. 3-9)	Introduction to the features of the Control and Estimation Tool Manager GUI
Configuring Inputs and Outputs for the Linearized Model (p. 3-12)	Selecting and inspecting linearization points, and performing open loop analysis
Specifying Operating Points (p. 3-20)	Specifying the operating point for linearization using completely known values, operating point specifications, or values from a simulation

Linearizing the Model (p. 3-35)

Linearizing a Block (p. 3-39)

Analyzing the Results (p. 3-41)

Exporting and Saving Your Work  
(p. 3-50)

Running the linearization

Running the linearization of a single  
block or subsystem

Inspecting the linearization results  
and displaying the results in the LTI  
Viewer.

Exporting the results to the  
MATLAB workspace and saving  
them for later use

## Overview

The main steps to linearize a model using the Simulink Control Design GUI are

- 1** Creating or opening a model
- 2** Opening a linearization project in the Control and Estimation Tools Manager
- 3** Configuring the model
- 4** Specifying the operating point
- 5** Linearizing the model or block
- 6** Analyzing the results
- 7** Exporting and saving your work

The following section introduces an example containing a nonlinear system (a levitating magnetic ball). Subsequent sections of this chapter use the example for a detailed discussion of each step.

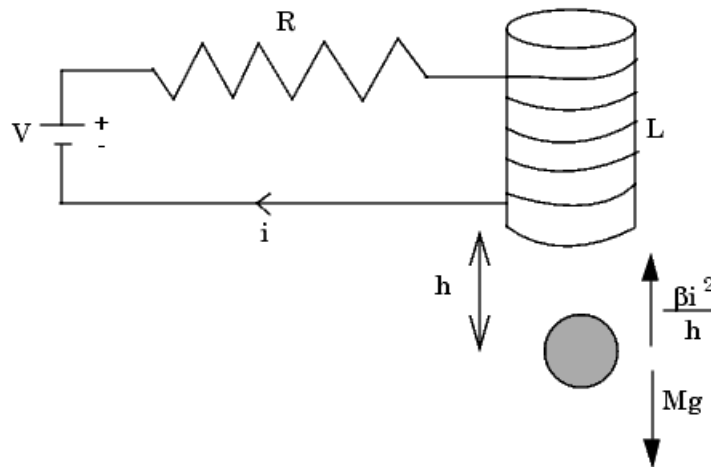
You can also use the Simulink Control Design command line functions to linearize a model. Chapter 4, “Linearizing Models Using Functions” gives detailed information on these functions. For a discussion of the advantages and disadvantages of the GUI versus the command line interface, please refer to “Using the GUI vs. Command Line Functions” on page 1-4.

## Example: The Magnetic Ball

This section introduces an example that continues throughout the remaining sections of this chapter. By following this example, you will learn the process of linearizing a model with Simulink Control Design.

### Magnetic Ball System

The electronic circuit in the following figure consists of a voltage source, a resistor, and an inductor in the form of a tightly wound coil. An iron ball beneath the inductor experiences a gravitational force as well as an induced magnetic force (from the inductor) that opposes the gravitational force.



### Schematic Diagram of Magnetic Ball System

### Model Equations

This section describes the model equations for the example started in the previous section .



A differential equation for the force balance on the ball is given by

$$M \frac{d^2h}{dt^2} = Mg - \frac{\beta i^2}{h}$$

where  $M$  is the mass of the ball,  $h$  is the height (position) of the ball,  $g$  is the acceleration due to gravity,  $i$  is the current, and  $\beta$  is a constant related to the magnetic force experienced by the ball. This equation describes the height,  $h$ , of the ball due to the unbalanced forces acting upon it.

The current in the circuit also varies with time and is given by the following differential equation

$$L \frac{di}{dt} = V - iR$$

where  $L$  is the inductance of the coil,  $V$  is the voltage in the circuit, and  $R$  is the resistance of the circuit.

The system of equations has three states

$$h, \frac{dh}{dt}, i$$

The system also has one input ( $V$ ), and one output ( $h$ ). It is a nonlinear system due to the term in the equation involving the square of  $i$  and the inverse of  $h$ . Due to its nonlinearity, you cannot analyze this system using methods for linear-time-invariant (LTI) systems such as step response plots, bode diagrams, and root-locus plots. However, linearizing using Simulink Control Design approximates the nonlinear system as an LTI system. This linearized system can then use the LTI Viewer for display and analysis. Refer to “Purpose of Linearization” on page 1-3 for a discussion of the uses of linearized models and “Linearization of Nonlinear Models” on page 2-4 for a discussion of the linearization process.

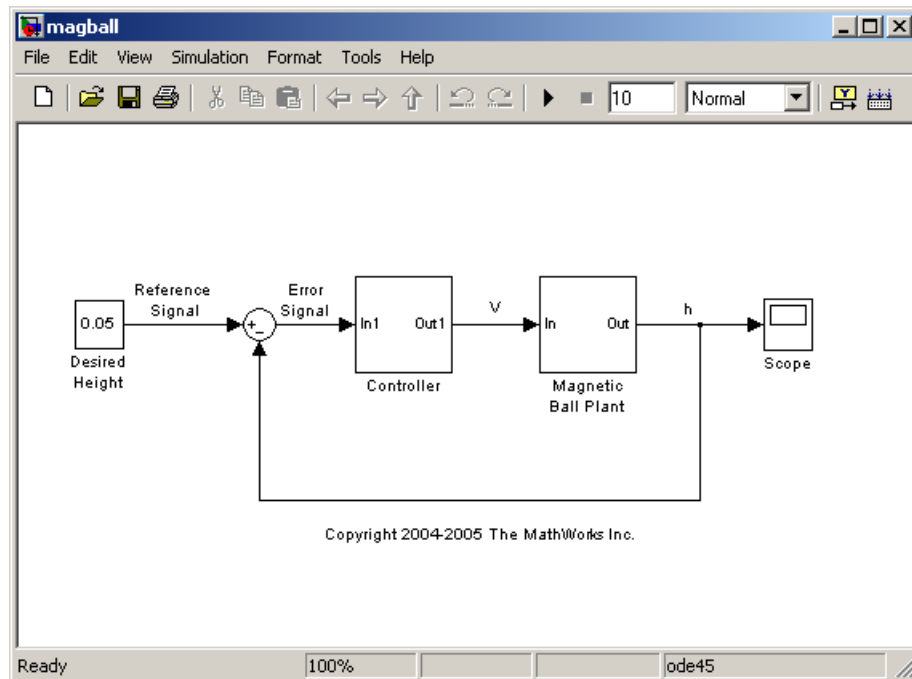
## Creating or Opening a Simulink Model

The first step in the linearization process is to create or open a Simulink model of your system. The model can have any number of inputs and outputs (including none), and any number of states. The model can include user-defined blocks or S-functions. Your model can involve a plant only, a plant with a feedback loop and controller, or any number of subsystems.

To open the model for the magnetic ball example introduced in “Example: The Magnetic Ball” on page 3-4, type

```
magball
```

at the MATLAB prompt. The magnetic ball system opens in Simulink as shown in this figure.

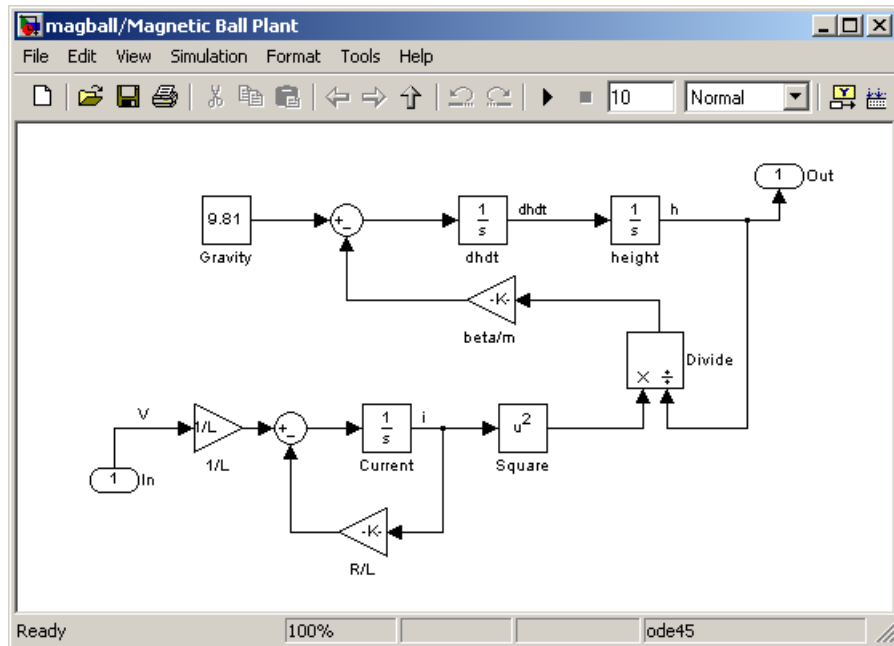


### Simulink Model of Magnetic Ball System

The magball model consists of

- The magnetic ball system itself, within the subsystem labeled Magnetic Ball Plant.
- A Controller subsystem that controls the height of the ball by balancing the forces acting on it.
- A reference signal that sets the desired height of the ball.
- A Scope block that displays the height of the ball as a function of time.

Double-click a block to view its contents. The Controller block contains a zero-pole-gain model. The Magnetic Ball Plant block is shown in this figure.



**Plant Subsystem of the Magnetic Ball Model**

The input to the Magnetic Ball Plant system, which is also the output of the Controller subsystem, is the voltage,  $V$ . The output is the height of the ball,  $h$ . The system contains three states within the three integrators: height,  $dhd t$ , and Current.

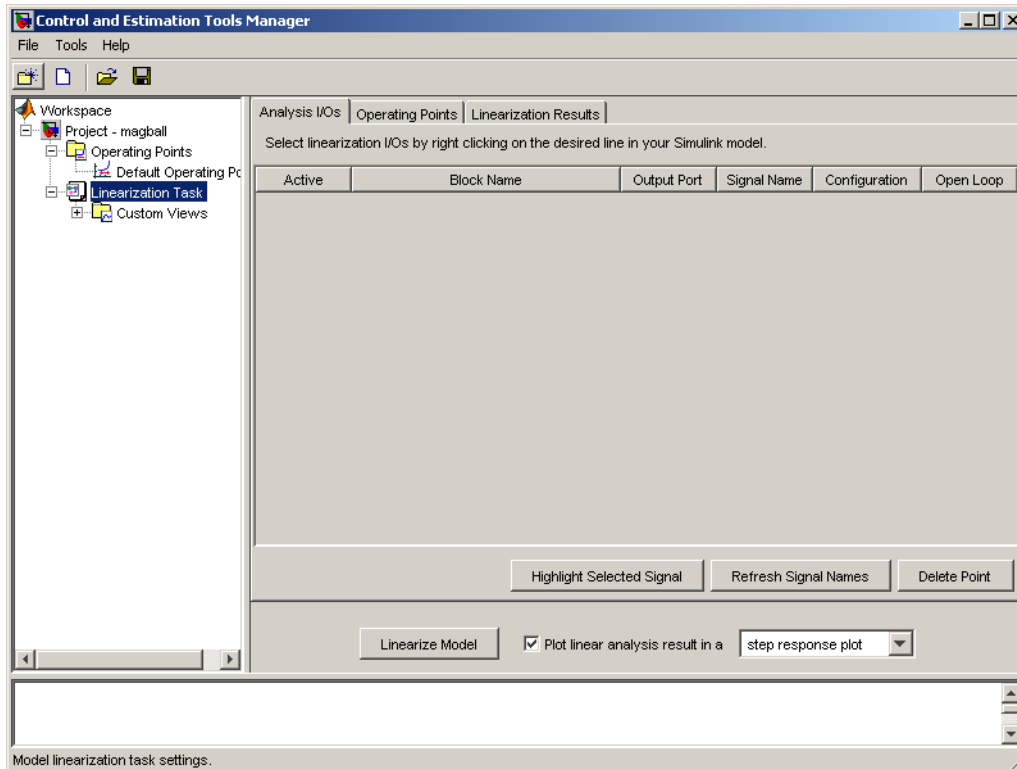
Values of the parameters are given as:  $M=0.1$  kg,  $g=9.81$  m/s<sup>2</sup>,  $R=2$  Ohm,  $L=0.02$  H,  $\beta=0.001$ .

## Beginning a New Linearization Project

The Control and Estimation Tools Manager provides a graphical environment for performing, and managing, control and estimation tasks such as linearization. Within this environment you can create projects which can include several tasks, including tasks from Simulink Parameter Estimation and the Model Predictive Control Toolbox, when they are installed.

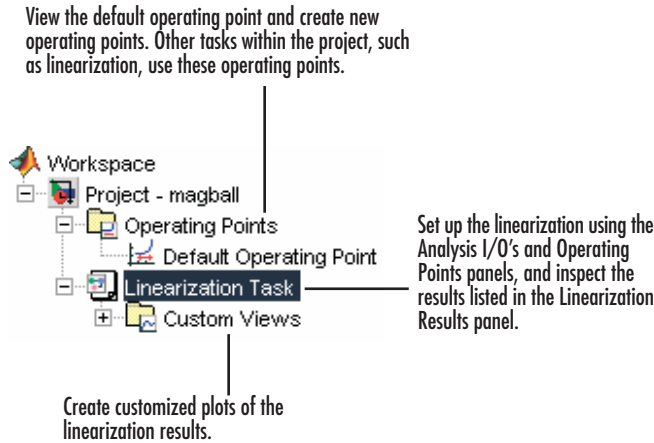
To open a new project for linearization within the Controls and Estimation Tools Manager, choose **Tools > Control Design > Linear Analysis** within the Simulink model window.

To create a new project for linearization of the magball model, continuing the example from “Example: The Magnetic Ball” on page 3-4, choose **Tools > Control Design > Linear Analysis** from the magball window. The Control and Estimation Tools Manager opens and creates a new linearization project, as shown in the following figure.



#### Control and Estimation Tools Manager Window

The Control and Estimation Tools Manager creates and manages projects that use tools such as Simulink Control Design. The left panel is the project tree, which contains all your current projects. At this stage you should have just one project, **Project - magball**. Select a node within the tree to display its contents in the panel on the right.



### Project Tree

- For information on the **Operating Points** node or the **Operating Points** panel within the **Linearization Task** node, refer to “Specifying Operating Points” on page 3-20.
- For information on the **Analysis I/Os** panel within the **Linearization Task** node, refer to “Configuring Inputs and Outputs for the Linearized Model” on page 3-12.
- For information on the **Linearization Results** panel within the **Linearization Task** node and inspecting linearization results, refer to “Analyzing the Results” on page 3-41.
- For information on **Custom Views**, refer to “Analyzing the Results” on page 3-41.

## Configuring Inputs and Outputs for the Linearized Model

Before linearizing the model, you must first configure the model diagram. This involves selecting input and output linearization points (also called analysis I/O's), and then, if necessary, inserting open loop points for performing open loop analysis. Inspect the selected linearization points using the **Analysis I/Os** panel within the **Linearization Task** node of the Control and Estimation Tools Manager.

A linearization input point defines an input to the linearized model while a linearization output point defines an output of the linearized model. Additionally, when the linearized models are computed using numerical perturbation, an input point is the point on the diagram where the small perturbation to the input signal is introduced and an output point is the point on the diagram where the small perturbation to the output signal is measured. Linearization input and output points are not the same as *operating points* which define the state of the model at the point of linearization.

The region between the input and output points defines the portion of the model that you want to linearize, unless a feedback loop feeds the output signal back into the input signal. When such a feedback loop is present, you might want to remove the effect of the loop by inserting an open loop point. Instructions for inserting open loop points are given in “Performing Open Loop Analysis” on page 3-16.

You can use Simulink Control Design to linearize the whole model, or any blocks or subsystems within the model. Define the system you are linearizing by placing linearization points before and after it in the model diagram.

When you want to linearize a model that contains references to other models using the Model block, you must use the numerical perturbation linearization algorithm. The numerical perturbation algorithm numerically perturbs the whole model and does not use linearization input and output points. See “Numerical-Perturbation Linearization” for more information.



---

**Note** Linearization input and output points should not be confused with Simulink Inport and Outport blocks. Input and output points define linear analysis inputs and outputs, while Inport and Outport blocks define the operating point of the system.

---

## Inserting Linearization Points

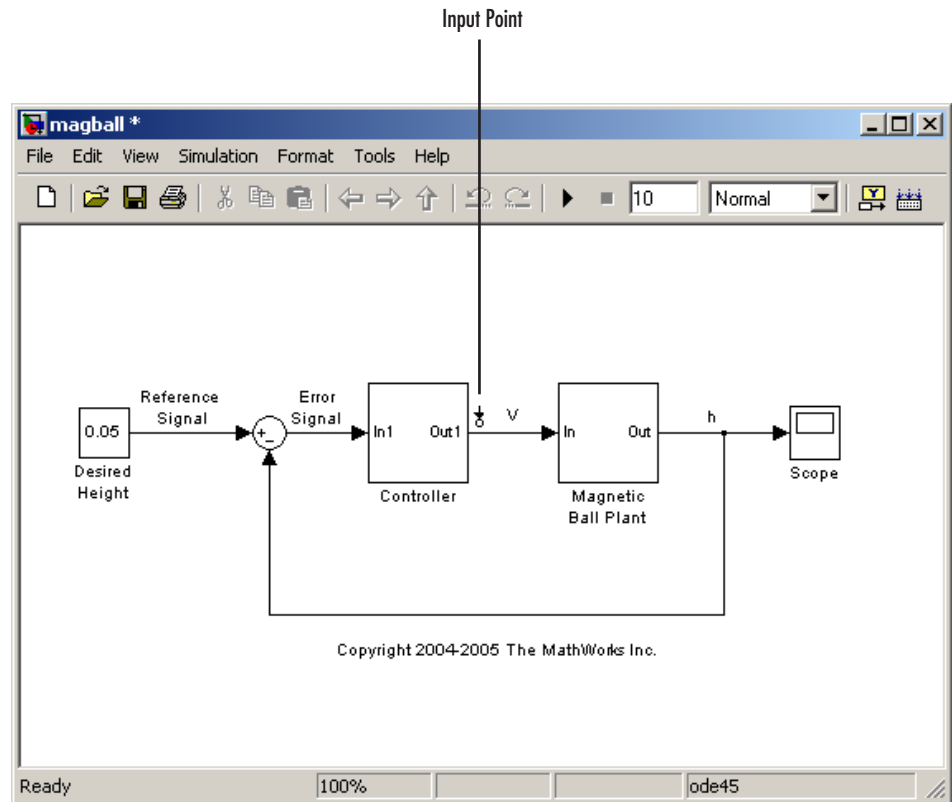
This section continues the example from “Example: The Magnetic Ball” on page 3-4. At this stage in the example, a linearization project has already been created for the model. See “Beginning a New Linearization Project” on page 3-9 for more information on creating a project for linearization.

For a definition of linearization point, also known as analysis I/O’s, see “Configuring Inputs and Outputs for the Linearized Model” on page 3-12.

In the magnetic ball model, the nonlinearities are all contained within the Magnetic Ball Plant. To linearize this subsystem,

- 1** Select the **Linearization Task** node within the Control and Estimation Tools Manager.
- 2** On the model diagram position the mouse on the signal line between the Magnetic Ball Plant and the Controller. Right-click and select **Linearization Points > Input Point** from the menu.

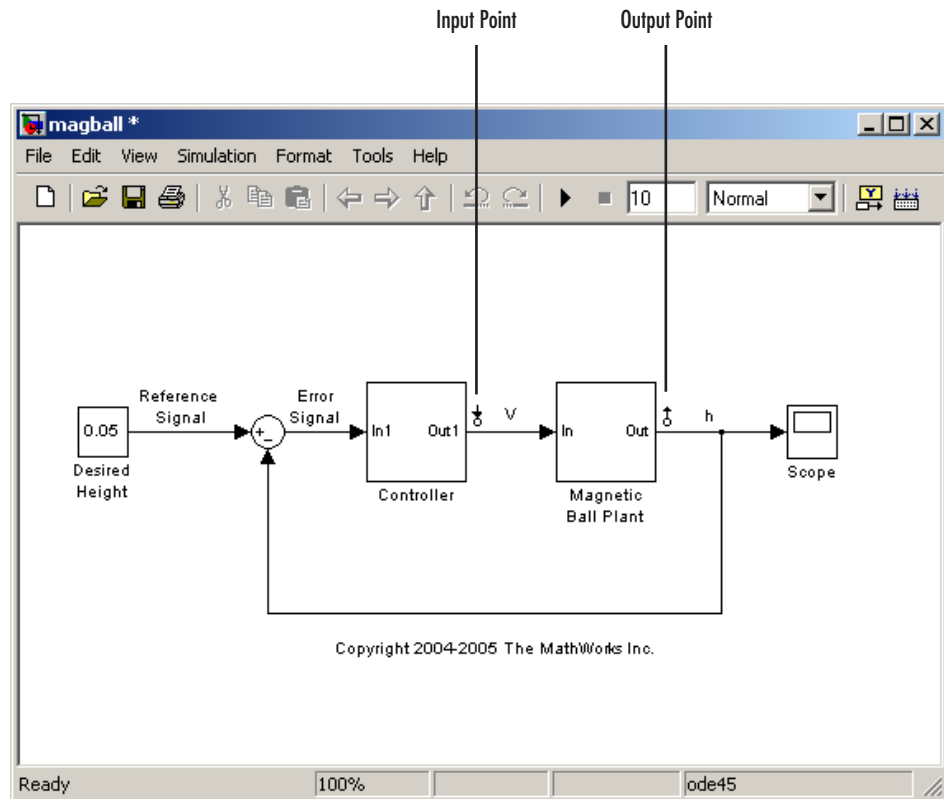
The model diagram now contains a small arrow pointing towards a circle just above the signal line between the Controller and the Magnetic Ball Plant, as in the following figure. This symbol indicates an input point for linearization has been placed there.



#### Magnetic Ball Model with Input Point Selected

- 3 Right-click the signal line after the Magnetic Ball Plant and select **Linearization Points > Output Point** from the menu.

A small arrow pointing away from a circle on the signal line appears after the Magnetic Ball Plant indicating an output point for linearization has been placed there. The diagram should now look like that in the following figure.



### Magnetic Ball Model with Input and Output Points Selected

**Note** All linearization points are referenced to the output port of the block the signal line originates from. For example, placing a linearization point anywhere on the feedback loop in the figure above will result in a linearization point at the output of the Magnetic Ball Plant block.

To remove a linearization point from the model, see “Removing Linearization Points” on page 3-16. To inspect the linearization points in the Control and Estimation Tools Manager, see “Inspecting Analysis I/Os” on page 3-18.

### Removing Linearization Points

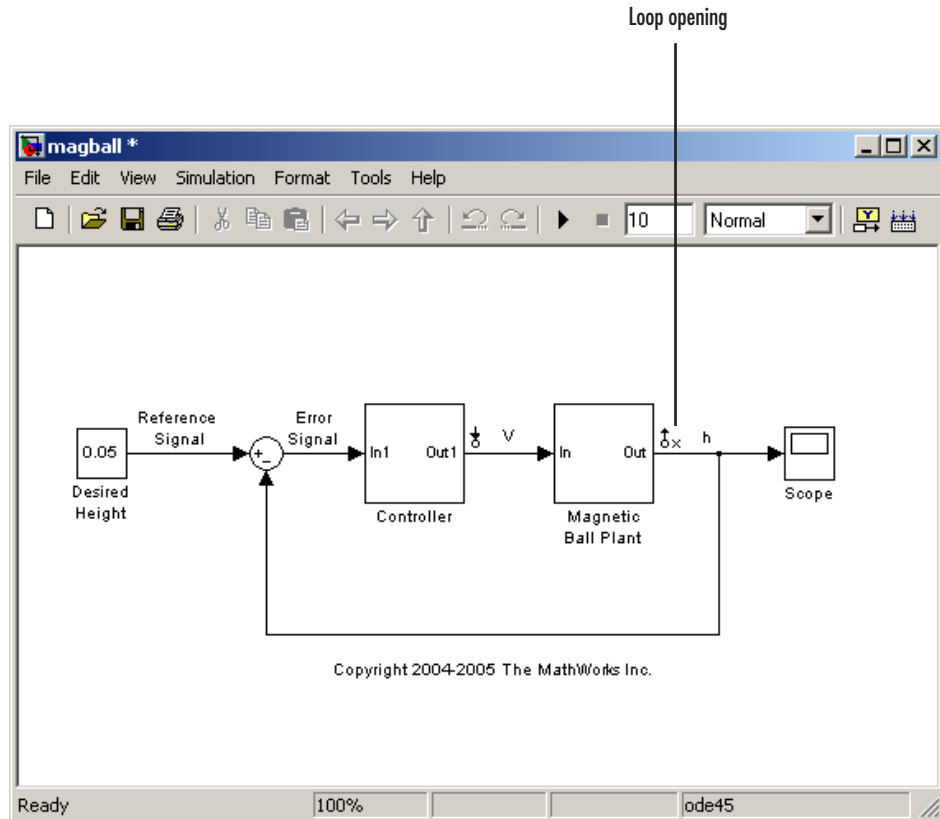
To remove a linearization point from a signal line in your model, repeat the same process as for inserting a linearization point. For example, to remove an input point, right-click on the signal line containing the input point and select **Linearization Points > Input Point** from the menu. The input point disappears from the diagram.

### Performing Open Loop Analysis

Due to the presence of feedback loops in a model, the input and output points might not completely define the portion of the model you want to linearize. In these cases, to remove the effect of signals feeding back into the portion of the model you are linearizing, you might choose to insert an open loop point. For more information on this topic refer to “Understanding Open Loop Analysis” on page 2-10.

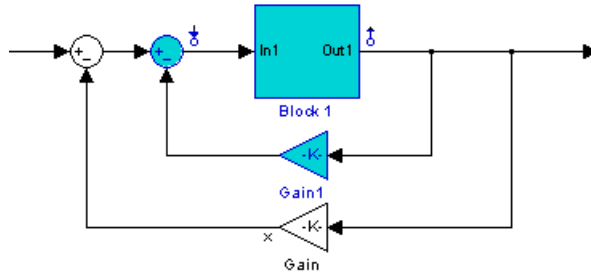
This section continues the example from “Example: The Magnetic Ball” on page 3-4. At this stage in the example, a linearization project has already been created for the model and linearization input and output points have been inserted. See “Inserting Linearization Points” on page 3-13 for more information on inserting linearization points.

To linearize only the Magnetic Ball Plant, right-click the signal line containing the output point and choose **Linearization Points > Open Loop**. This inserts a small x next to the output point in the diagram, representing a loop opening.



### Magnetic Ball Model with Loop Opening

**Note** You do not need to place a loop opening in the same place as an input or output point. For example, in the following figure the highlighted blocks are included in the linearization. The loop opening is placed after the gain on the outer feedback loop, which removes the effect of this loop from the linearization. Placing a loop opening at the same place as the output point would have removed the effect of the inner loop from the linearization as well.

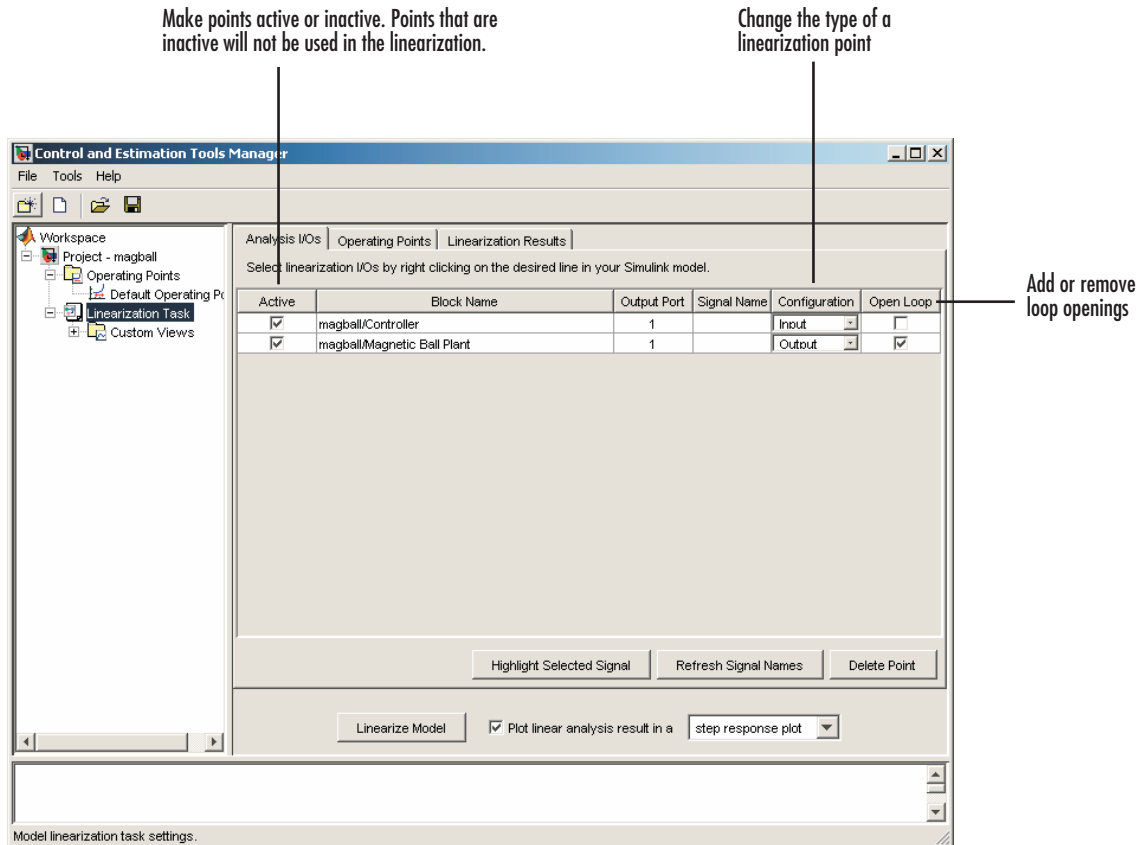


### Open Loop Example

### Inspecting Analysis I/Os

This section continues the example from “Example: The Magnetic Ball” on page 3-4. At this stage in the example, a linearization project has already been created for the model, linearization input and output points have been inserted, and loop openings have been created. See “Performing Open Loop Analysis” on page 3-16 for more information on creating loop openings.

To view the linearization points, select the **Analysis I/Os** panel of the **Linearization Task** node in the Control and Estimation Tools Manager, as shown in the following figure. Use this panel to inspect and make changes to your linearization points.



## Analysis I/Os Panel

**Note** You cannot make changes to the **Analysis I/Os** or perform open loop analysis when you select Numerical Perturbation as the **Linearization Algorithm** in the Linearization Task Options window (accessed by selecting **Tools > Options** from the Control and Estimation Tools Manager window). See for information.

## Specifying Operating Points

Before linearizing the model, you must choose an operating point about which to linearize the system. This is often a steady state value. Refer to “Operating Points” on page 2-2 for more information on the role of operating points in linearization.

There are five methods available for specifying the operating point:

- Specify target values or constraints on a subset of the model’s inputs, outputs, and states (“Computing Operating Points from Specifications” on page 3-21). Simulink Control Design uses numerical methods to determine the full operating point based on this partial specification.

For example, when you know that the height of the ball in `magball` should be 0.05, the rate of change of the height is small, the current should be positive, and your initial guess for the states in the Controller is 0, compute an operating point that closely matches the specifications.

- Completely specify all inputs and states in the operating point (“Specifying Completely Known Operating Points” on page 3-26).

For example, when you know that the height of the ball in `magball` should be 0.05, the rate of change of the height should be 0, the current should be 7.0036, and you also know the values of the states in the Controller, this information completely specifies the operating point.

- Extract an operating point from a simulation of the model ().

For example, you run a simulation of a model and use the values of the states and inputs at time 10 as the operating point values. This is especially useful when the simulation has reached a steady state.

- Accept the default operating point. The initial values of the states, inputs, and outputs, define this operating point. Only use the default operating point when the initial values are very close to the operating point of interest.
- Import a previously saved operating point from another project, the MATLAB workspace, or a file (“Importing Operating Points” on page 3-31).



The following sections give instructions for each of these methods, with the exception of accepting the default values, which does not require any further action.

---

**Note** The operating point consists of values for *all* the states in the model although only those states between the linearization points will be linearized. This is because the whole model contributes to the operating point values of the states/inputs/outputs of the portion of the model you are linearizing.

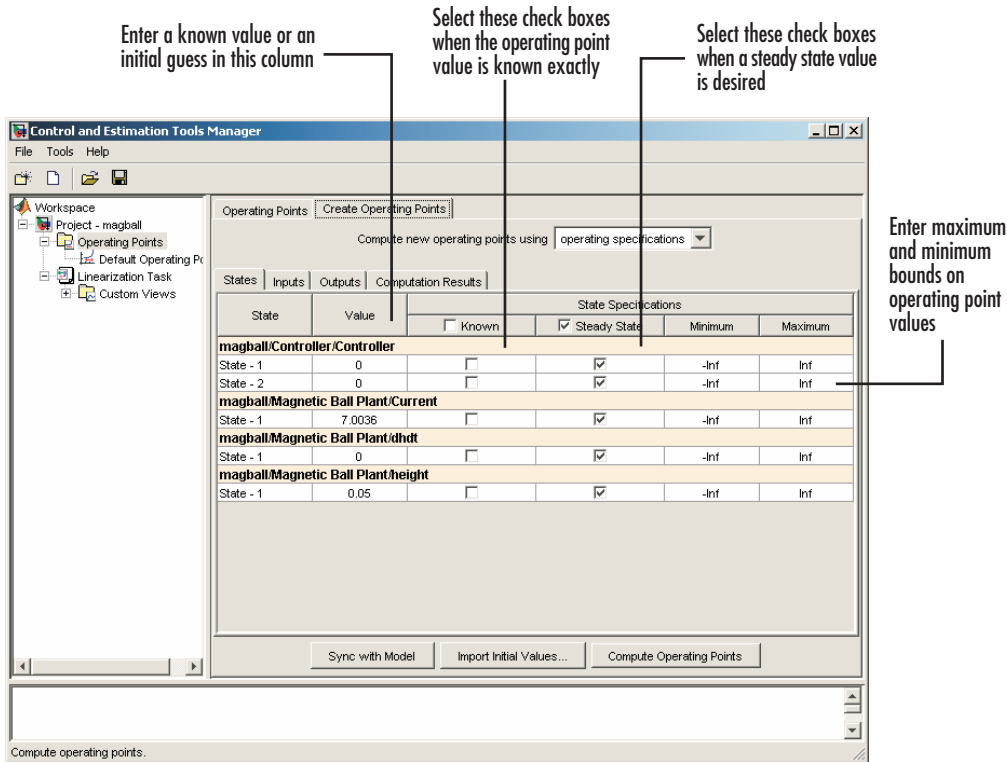
---

## Computing Operating Points from Specifications

This section continues the example from “Example: The Magnetic Ball” on page 3-4. At this stage in the example, a linearization project has already been created for the model, and linearization points have been inserted and inspected. See “Inspecting Analysis I/Os” on page 3-18 for more information on inspecting linearization points.

Use Simulink Control Design to compute the operating point from specifications when you only know partial or implicit information:

- 1 Create a new operating point by either
  - Selecting the **Operating Points** node and then selecting the **Compute Operating Points** tab on the right,
  - or
  - Clicking the **New Operating Point** button within the **Operating Points** panel of the **Linearization Task** node.
- 2 From **Compute new operating points using**, select operating specifications. The Control and Estimation Tools Manager window should now look like that in the following figure.



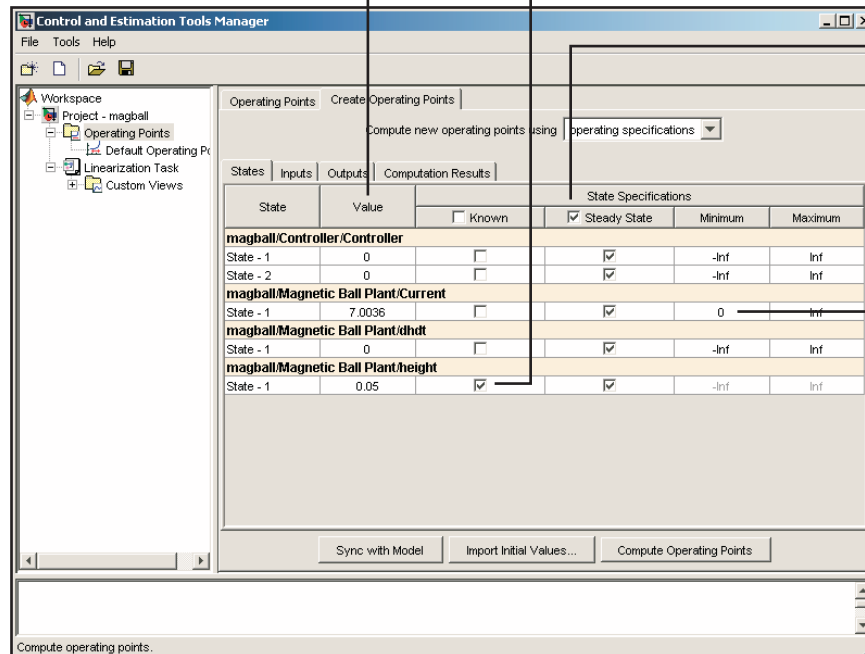
#### Compute Operating Points Panel with Operating Specifications Selected

- Enter operating point specifications in the table, such as any known values or constraints on signal values. Switch between states, inputs, and outputs using the tabs on the left.

A suitable set of specifications for the magball model is shown in the following figure. Note that in this case, the model determines the values of inputs and outputs.

The height of the ball is known to be 0.05 (the same as the reference signal value) and for a steady ball, use an initial guess of 0 for dhdt. For the remaining states choose arbitrary initial guesses.

The height of the ball is known exactly, because it is the same as the reference signal height.



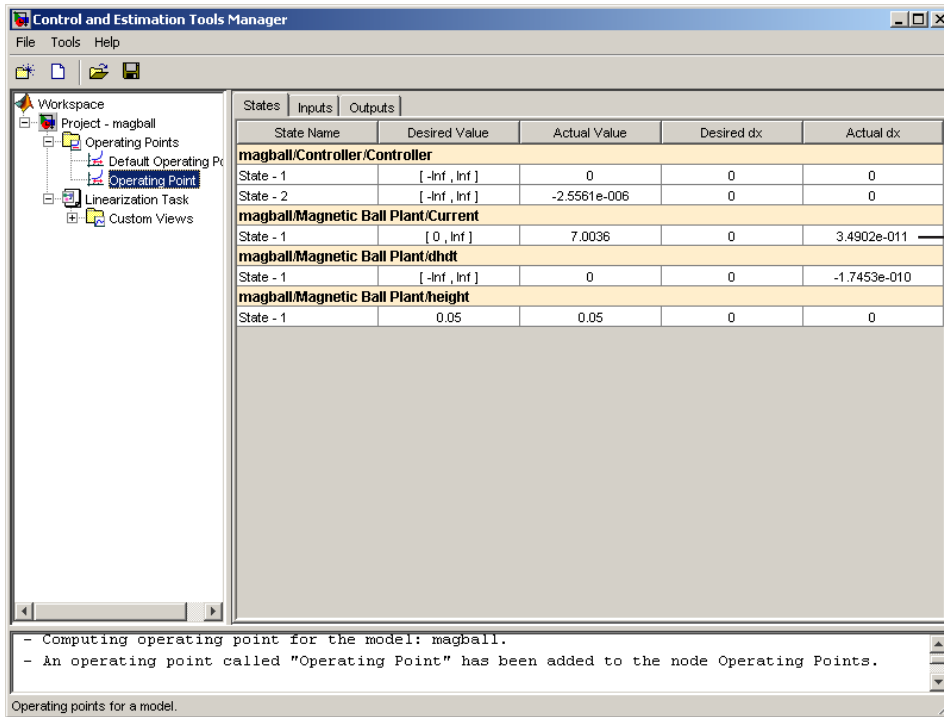
A steady state operating point is desired.

Use a minimum value of zero for the current, because by convention current is positive when flowing clockwise around the circuit.

## Operating Point Specifications

When you add states, inputs, or outputs to the model, or remove them from the model, click the **Sync with Model** button to update the operating point table to reflect these changes.

- 4 Click **Compute Operating Point**. Simulink Control Design finds an operating point that closely matches the specifications and adds the new operating point, labeled **Operating Point**, to the **Operating Points** node. Note that some specifications, even values specified as **Known**, may not be met exactly. Select the operating point in the project tree to view its contents and assess the results.



### Operating Point Results

For information on options that you can set when finding operating points from partial specifications, see in the online documentation.

---

**Note** Inputs and outputs for the operating point are not the same as the linearization input and output points, or analysis I/O's, used to define the inputs and outputs of the linearized model. Inputs and outputs for operating point are the inputs and outputs for the entire Simulink model, along with any additional, user-defined output constraints.

---

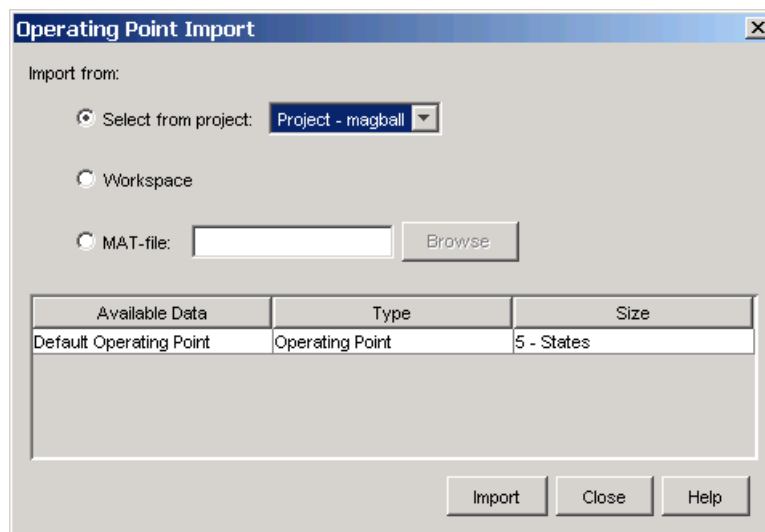
### Computing Equilibrium Operating Points

You can use Simulink Control Design to compute equilibrium operating points. Follow the basic instructions in “Computing Operating Points from Specifications” on page 3-21. When you enter specifications in the **States**

panel, select the **Steady State** check box at the top of the table. This causes the algorithm to look for an operating point in which all states are at equilibrium, or steady state.

## Importing Initial Values

When you want populate the **Value** column of the operating point specifications by importing initial or known values from another operating point, a Simulink states structure, or a vector of values, click the **Import Initial Values** button at the bottom of the window. The Operating Point Import dialog opens, as shown below.



## Importing Operating Point Initial Values

Within this window select where to import the initial values from (a project, the workspace, or a file), then select the operating point from the list of available operating points below (or in the case of MAT-files, browse for a file). Click **Import** to import the initial values from the selected operating point into the **Value** column of the operating point specifications.

## Constraining Outputs

Operating specifications often include constraints on the values of specific signals in the model. To constrain output signals when determining operating

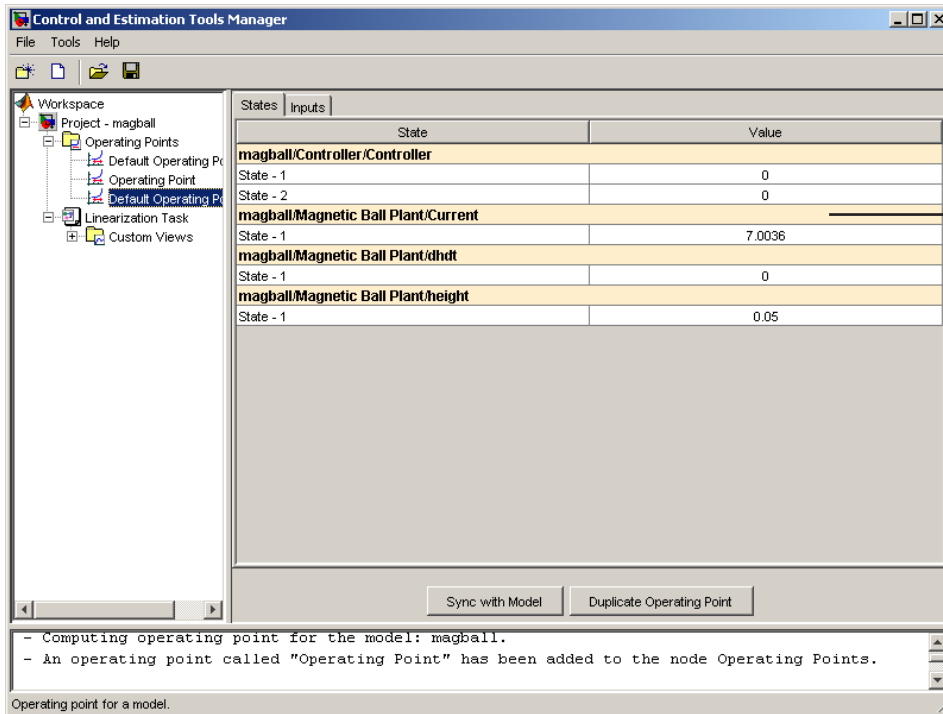
points from specifications, add an output constraint annotation to the model by right-clicking on the signal line and choosing **Output Constraint** from the menu. This adds a small T to the signal line. Then, within the **Outputs** panel of the **Compute Operating Points** panel, select the **Known** check box and enter desired values as well as minimum and maximum values for this signal.

### Specifying Completely Known Operating Points

This section continues the example from “Example: The Magnetic Ball” on page 3-4. At this stage in the example, a linearization project has already been created for the model, and linearization points have been inserted and a steady state operating point has been computed from specifications. See “Computing Operating Points from Specifications” on page 3-21 for more information on computing steady state operating points.

When you know the values of *all* states and inputs at the operating point, create a new operating point in the Control and Estimation Tools Manager and edit the operating point values:

- 1 Select the **Operating Points** node in the left panel and then select the **Operating Points** tab in the right panel.
- 2 Click the **New** button in the bottom right corner to create a new operating point under the **Operating Points** node. This new operating point is called **Default Operating Point (2)**.
- 3 View the details of **Default Operating Point (2)** by either
  - Selecting it under the **Operating Points** node in the project tree,
  - or
  - Selecting it in the **Operating Points** panel of the **Linearization Task** node, and then clicking **View**.



Enter known values of states and inputs at the operating point.

## Default Operating Point (2)

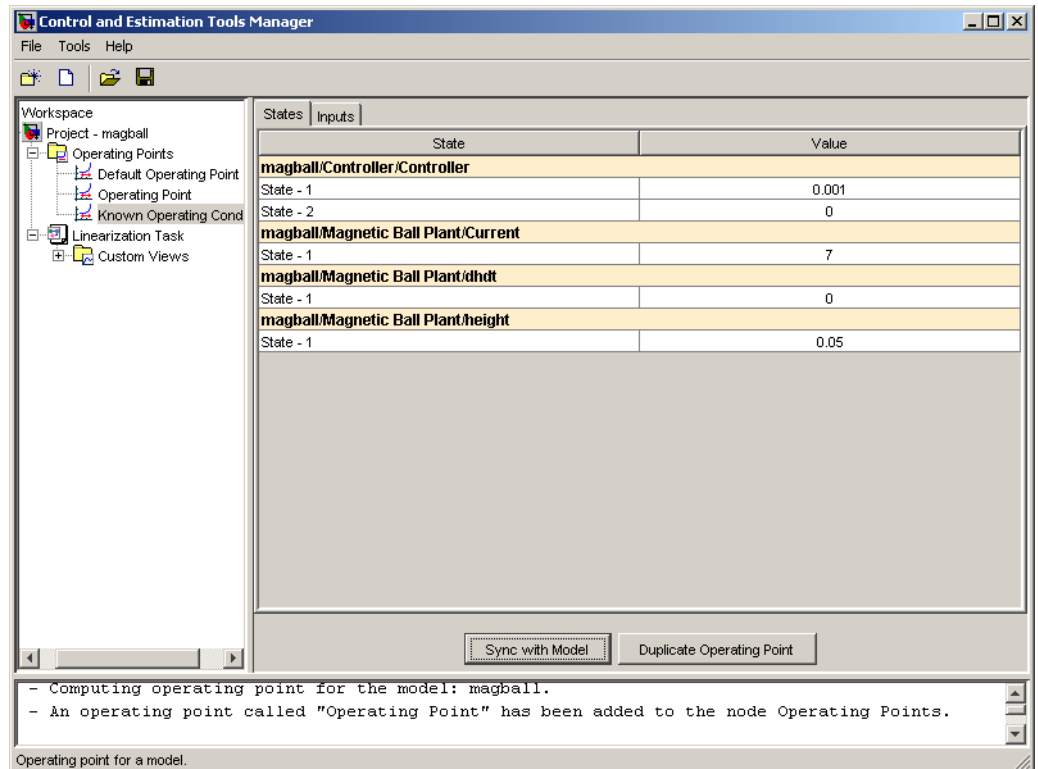
- 4 Edit the operating point by entering new values in the table. Switch between state and input values using the tabs on the left.

For example, change the value of **State-1** of **Controller** to 0.001 and the value of **Current** to 7. Note that for this model, input values cannot be entered because the model determines all inputs.

When you add states, inputs, or outputs to the model, or remove them from the model, click the **Sync with Model** button to update the operating point table to reflect these changes.

- 5 Rename the operating point by right-clicking **Default Operating Point (2)** under the **Operating Points** node, selecting **Rename**, and entering a new name in the dialog box.

For example, rename this operating condition to Known Operating Conditions. The panel should now look like that in the following figure.



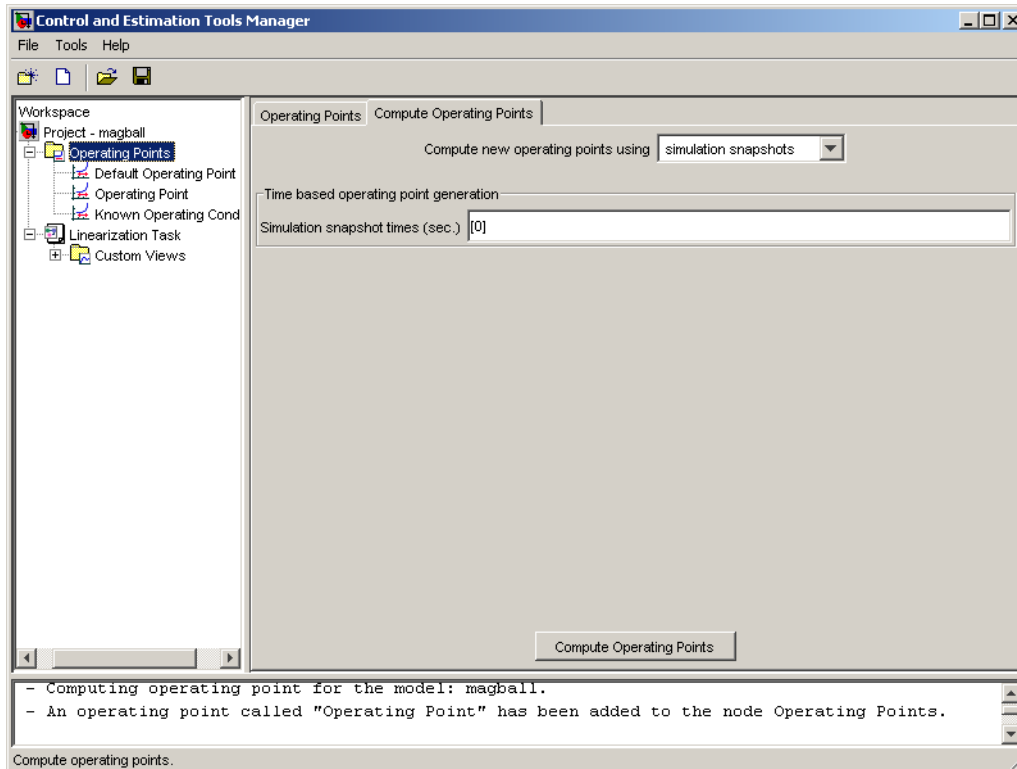
## Extracting Operating Points from Simulation

This section continues the example from “Example: The Magnetic Ball” on page 3-4. At this stage in the example, a linearization project has already been created for the model, and linearization points have been inserted, a steady state operating point has been computed from specifications, and a completely known operating point has been specified. See “Specifying Completely Known Operating Points” on page 3-26 for more information on specifying completely known operating points.



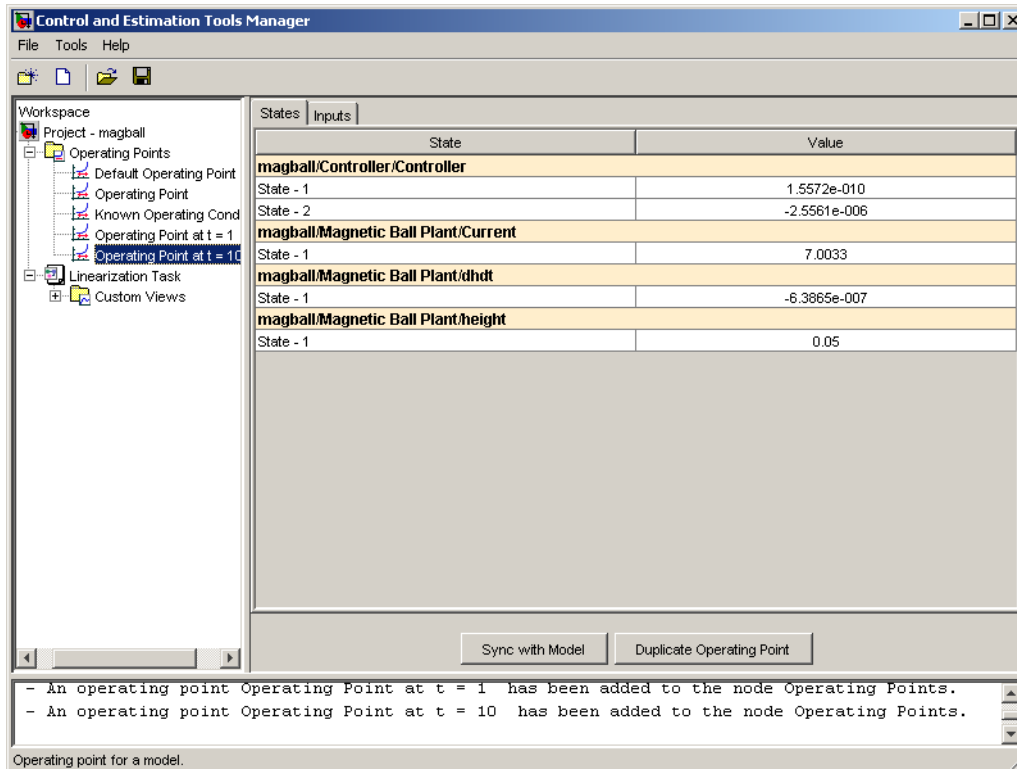
Use Simulink Control Design to extract operating points from a simulation of your model at specified times, such as when the simulation reaches a steady state solution:

- 1** Create a new operating point by either
  - Selecting the **Operating Points** node in the project tree and then selecting the **Compute Operating Points** tab on the right,
  - or
  - Clicking the **New Operating Point** button within the **Operating Points** panel of the **Linearization Task** node.
- 2** From **Compute new operating points using**, select simulation snapshots. The window should now look like that in this figure.



#### Simulation Snapshots Set-Up

- 3 Enter a vector of times in the **Simulation Snapshot Times** field. For example, to compute operating points at  $t=1$  and  $t=10$ , enter  $[1, 10]$ .
- 4 Click **Compute Operating Point**. Simulink Control Design simulates the model, extracts operating points, labeled **Operating Point at  $t=1$**  and **Operating Point at  $t=10$** , and adds them to the **Operating Points** node in the project tree. Select an operating point to view its contents and assess the results as shown in this figure.



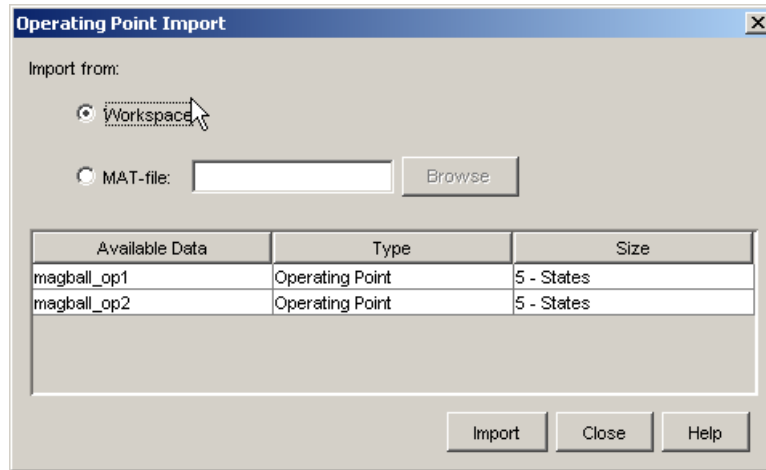
## Simulation Snapshot Operating Point Results

When you add states, inputs, or outputs to the model, or remove them from the model, click the **Sync with Model** button to update the operating point table to reflect these changes.

## Importing Operating Points

Use Simulink Control Design to import operating points from the MATLAB workspace or from a MAT-file.

- 1 To import a new operating point, select the **Operating Points** node in the project tree and then select the **Operating Points** tab on the right. Click the **Import** button at the bottom of the panel. This displays the Operating Point Import dialog.

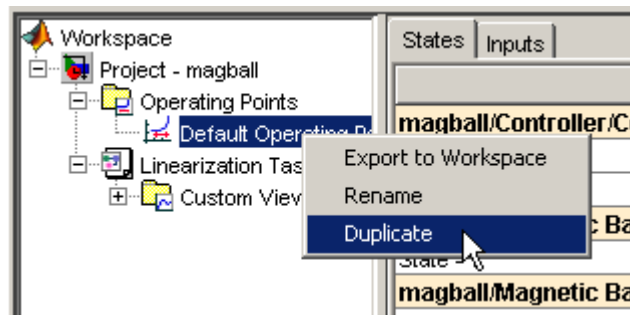


**Operating Point Import Dialog**

- 2 Within the Operating Point Import dialog choose **Workspace** or **MAT-file** as the location to import the operating point from, select an operating point from the list below, and then click **Import**. For this example, two operating points are loaded into the MATLAB workspace when you open the magball model.

### Copying Operating Points

In some situations you might want to create and edit a copy of an operating point. To create a copy of an operating point, right-click the operating point in the tree on the left, and select **Duplicate** from the right-click menu.

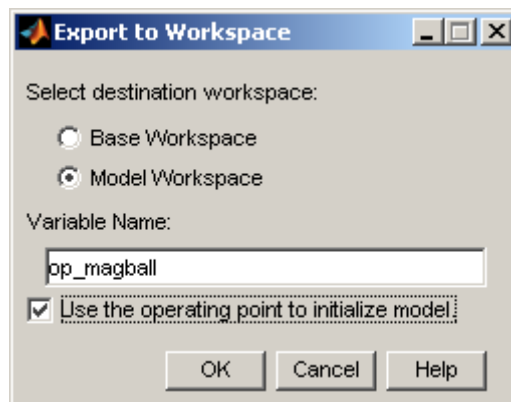


The new operating point appears beneath the original one in the tree. Click the new operating point to display its contents in the pane on the right. To change state or input values in the duplicated operating point, edit the values in the right pane. To change the name of the new operating point, right-click the operating point in the tree, select **Rename** from the right-click menu, and then enter a new name for the operating point.

Note that you cannot copy operating points that were computed from specifications. These operating points contain information related to the success of the optimization which would not be meaningful when the operating point values were changed.

## Exporting Operating Points

After creating operating points using Simulink Control Design, you can export them from the Control and Estimation Tools Manager to the MATLAB workspace or the model workspace. You can use an exported operating point to perform analysis at the MATLAB command line or to initialize a model for simulation. To export an operating point, right click on the operating point under **Operating Points** in the panel on the left and select **Export to Workspace**. This opens the Export to Workspace dialog, as shown below:



1 Under **Select destination workspace**, select either

- **Base workspace** to export the operating point to the MATLAB workspace where you can use it with Simulink Control Design command line functions

or

- **Model workspace** to export the operating point to the Model workspace where you can save it with the model for future use.

**2** Enter a name for the exported operating point.

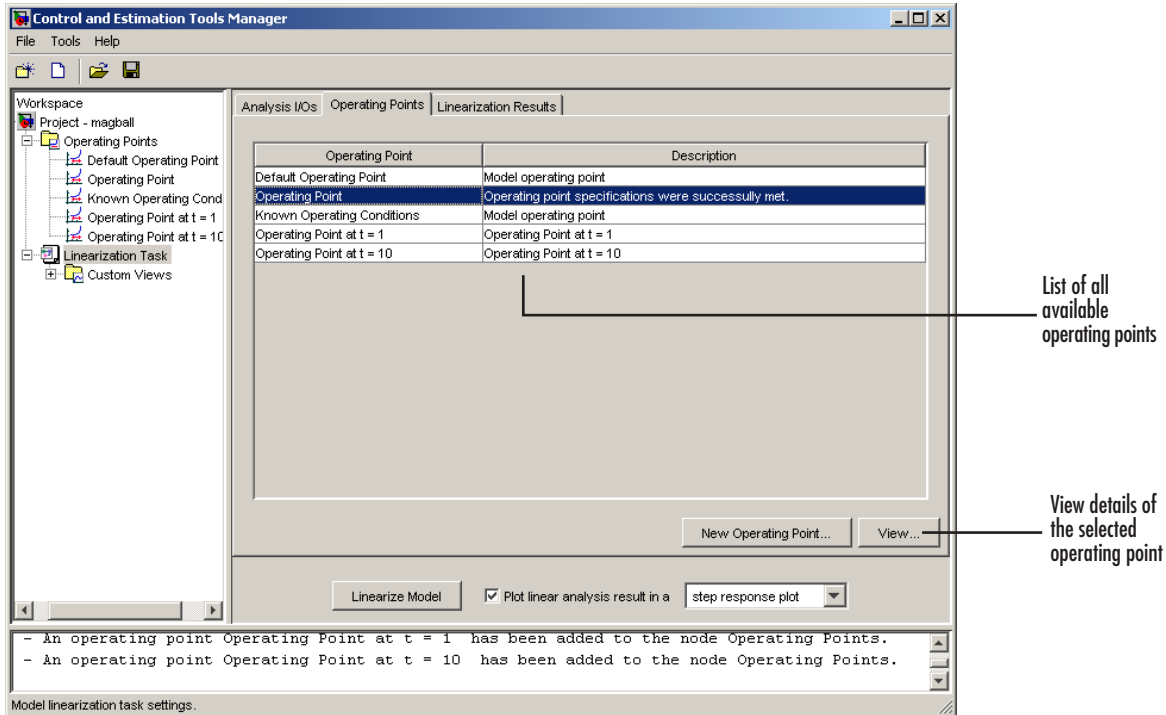
**3** Select **Use the operating point to initialize model** when you want to use the operating point values as initial conditions for the states and inputs in the model. The initial values are automatically set in the **Data Import/Export** pane of the Configuration Parameters dialog and Simulink uses these initial conditions when simulating the model.

## Linearizing the Model

This section continues the example from “Example: The Magnetic Ball” on page 3-4. At this stage in the example, a linearization project has been created, linearization points have been inserted, and operating points have been created. See “Specifying Operating Points” on page 3-20 for more information on creating operating points.

After configuring the model and setting up the operating points, you are ready to linearize the model.

Within the **Linearization Task** node, select the **Operating Points** tab. A list of all available operating points for this project is shown. Select the operating point(s) you want to use for the linearization. To linearize the magnetic ball model around the operating point computed from partial specifications, select that line in the list, as shown in this figure.

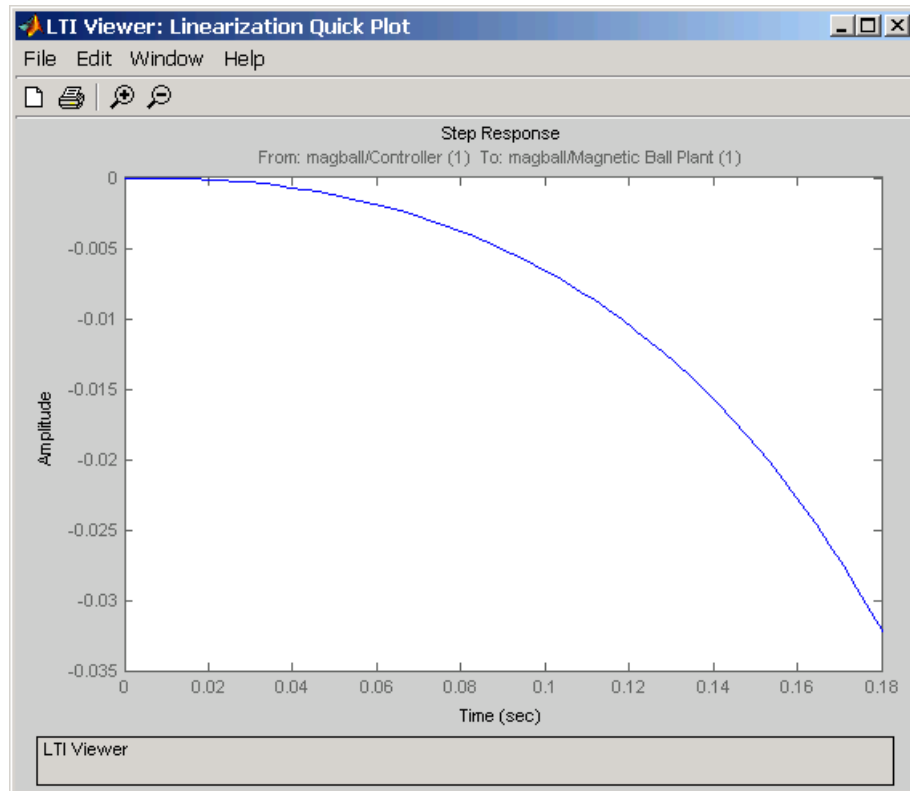


#### Select Operating Point for Linearization

To compute the linearized model, click the **Linearize Model** button at the bottom of the panel. The results display, by default, as a step response of the new linearized system in the LTI Viewer window, as shown in the following figure.

Select from one of seven options in the menu to the right of the **Linearize Model** button to display the results in a different format. Clear the check box next to this menu and the results will not display in the LTI Viewer.





### Linearization Results in LTI Viewer

For information on setting options for linearization, refer to the online documentation.

### Using the LTI Viewer

To add characteristics such as settling time or peak response to your plot, right-click anywhere in the plot area and choose an option from the menu. Add data markers by clicking the point you want to mark. In addition, you can display up to six plots at one time. To change the number of plots select **Edit > Plot Configurations**. Export the linearized model to the workspace by selecting **File > Export**. For more information on the LTI Viewer, please refer to the Control System Toolbox documentation.

### Linearizing Discrete-Time and Multi-Rate Models

The linearization method is the same for models containing discrete-time states or several different sample times. However, you can choose to adjust the **Linearization sample time** in the **Linearization** options panel. By default, this parameter is set to -1, in which case Simulink Control Design linearizes at the slowest sample rate in the model. To create a linearized model with a different sample time, enter a new value in the dialog window. A value of 0 gives a continuous-time model.

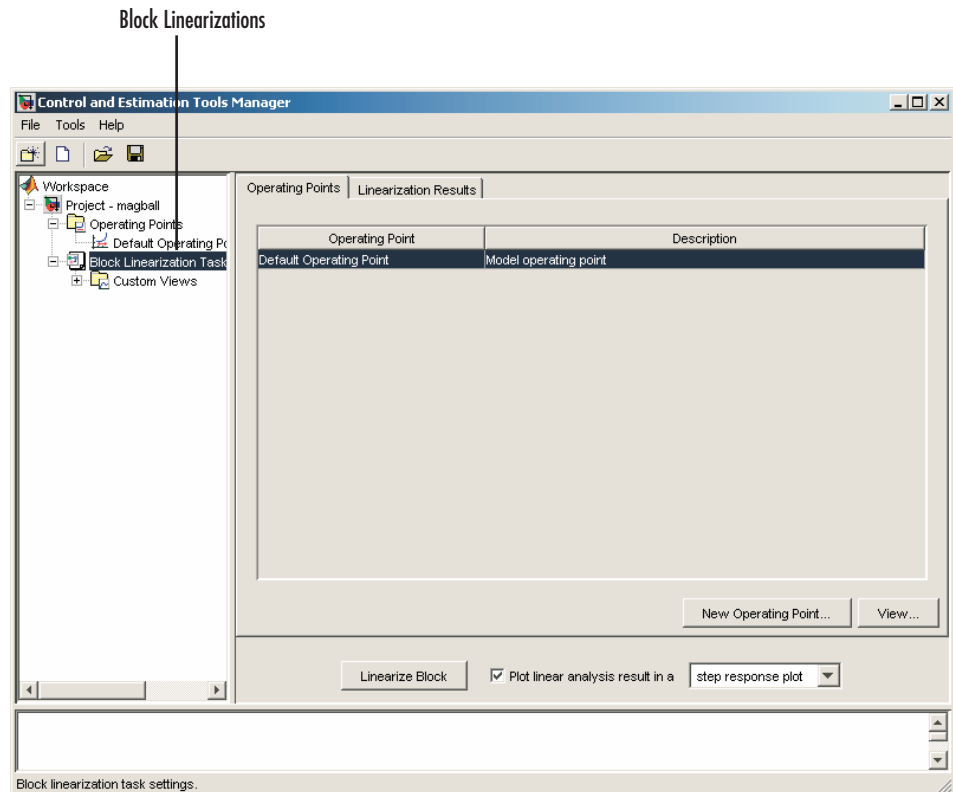
To change the method that Simulink Control Design uses for converting a multi-rate model to a single-rate model, change the **Rate conversion method** within the Options dialog box.

For more information, and examples, on methods and algorithms for rate conversions and linearization of multi-rate models, see the “Linearization of Multi-Rate Models” and “Rate Conversion Method Selection for Linearization” demos listed under the Simulink Control Design Demos in the demos browser.

## Linearizing a Block

With Simulink Control Design you can also linearize a single block in a Simulink model. To do this, right-click the block and choose **Linearize Block** from the context menu.

This adds a **Block Linearization Task** node to the project tree as shown in the following figure



### Control and Estimation Tools Manager with Block Linearization Task

To complete the linearization, specify an operating point in the same way as when linearizing models, and then click the **Linearize Block** button within the **Operating Points** panel of the **Block Linearization Task** node. You do not need to choose linearization input and output points because the inports

and outports of the block are used. If you do have linearization input and output points in your model, they will be ignored. You cannot linearize an individual block using numerical-perturbation linearization (when Numerical perturbation is selected as the **Linearization Algorithm** parameter).

---

**Note** To linearize an individual block, it must contain at least one data inport and output. Since SimMechanics and SimPowerSystems blocks have connection ports instead of inports and outports, they cannot be individually linearized.

---

## Analyzing the Results

To display all the linearization tasks of the current project, select the **Linearization Results** tab within the **Linearization Task** node. At this stage you should have just one, **Model**, which also appears as its own node in the project tree. To remove a linearization from the **Linearization Results** list, select it and then use the **Delete** button at the bottom of the panel.

To inspect and validate the results, refer to “Validating the Linearization” on page 3-41. To display and configure plots of the results, refer to “Displaying the Results” on page 3-46. For methods on simulating the linearized model for comparison with the original model, refer to . For methods on simulating the linearized model for comparison with the original model, refer to “Comparing the Linearized and Original Models” in the online documentation.

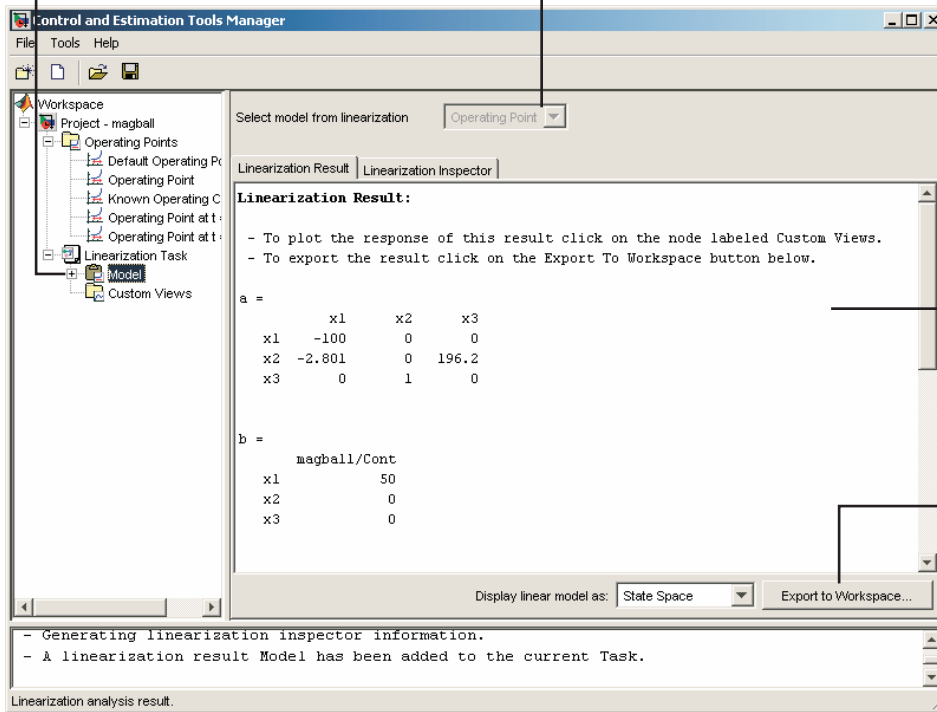
### Validating the Linearization

This section continues the example from “Example: The Magnetic Ball” on page 3-4. At this stage in the example, a linearization project has been created, linearization points have been inserted, operating points have been created, and a linearized model has been computed. See “Linearizing the Model” on page 3-35 for more information on computing the linearized model.

To display the **Linearization Summary**, as shown in the following figure, click the **Model** node in the project tree. To delete the linearization result, right-click **Model** and select **Delete** from the menu. The **Linearization Summary** panel displays the linearized model. By default it appears as a state space system and it displays the state space matrices. However, within the panel you can choose to view the model as a zero-pole-gain system or as a transfer function. You can also export the model to the MATLAB workspace.

View the operating points used in the analysis by clicking the + to the left of the Model node

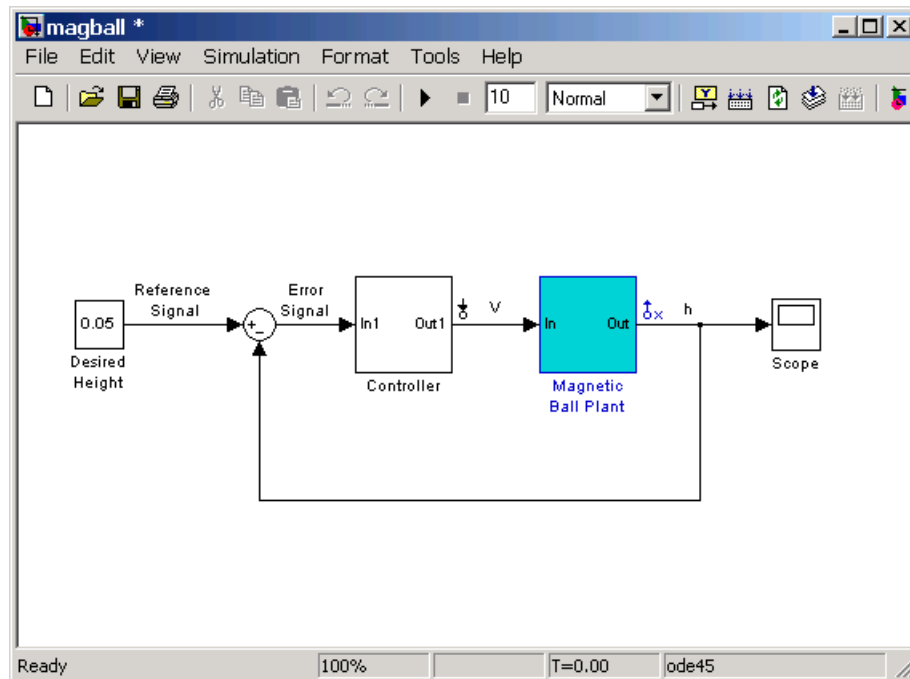
Switch between results based on different operating points



#### Linearization Result Panel

#### Highlighting Blocks in the Linearization

To verify that the correct portion of the model was linearized, you can highlight the blocks in the model that were used in the linearization. To do this, right-click the **Model** node and select **Highlight Blocks in Linearization**. The following figure shows that the Magnetic Ball Plant block, including all blocks within it, was used in the linearization.



### Highlight Blocks in Linearization Turned On

To remove the highlighting, right-click the **Model** node and select **Remove Highlighting**.

---

**Note** You cannot use the **Highlight Blocks in Linearization** or **Remove Highlighting** options for models that are linearized with numerical-perturbation linearization (when Numerical perturbation is selected as the **Linearization Algorithm** parameter in the Linearization Task Options window).

---

### Inspecting the Linearization Block-by-Block

To inspect the linearization block-by-block, click the **Linearization Inspector** tab within the **Model** node. This pane displays the linearizations of the individual blocks in the linearization path, that is, the blocks between the input and output points. Use this information to determine if the

linearization gave the expected results, and to compare linearizations using alternative operating points or parameter values.

This figure shows the results of the magnetic ball model in the **Linearization Inspector** pane with **Magnetic Ball Plant** highlighted.



Select a block to inspect.

State-space matrices A, B, C, and D are displayed along with the sample times, and whether the block is in the linearization path.

Sample time is zero for continuous systems, nonzero for discrete systems.

Plot results of this block linearization.

Linearization analysis result.

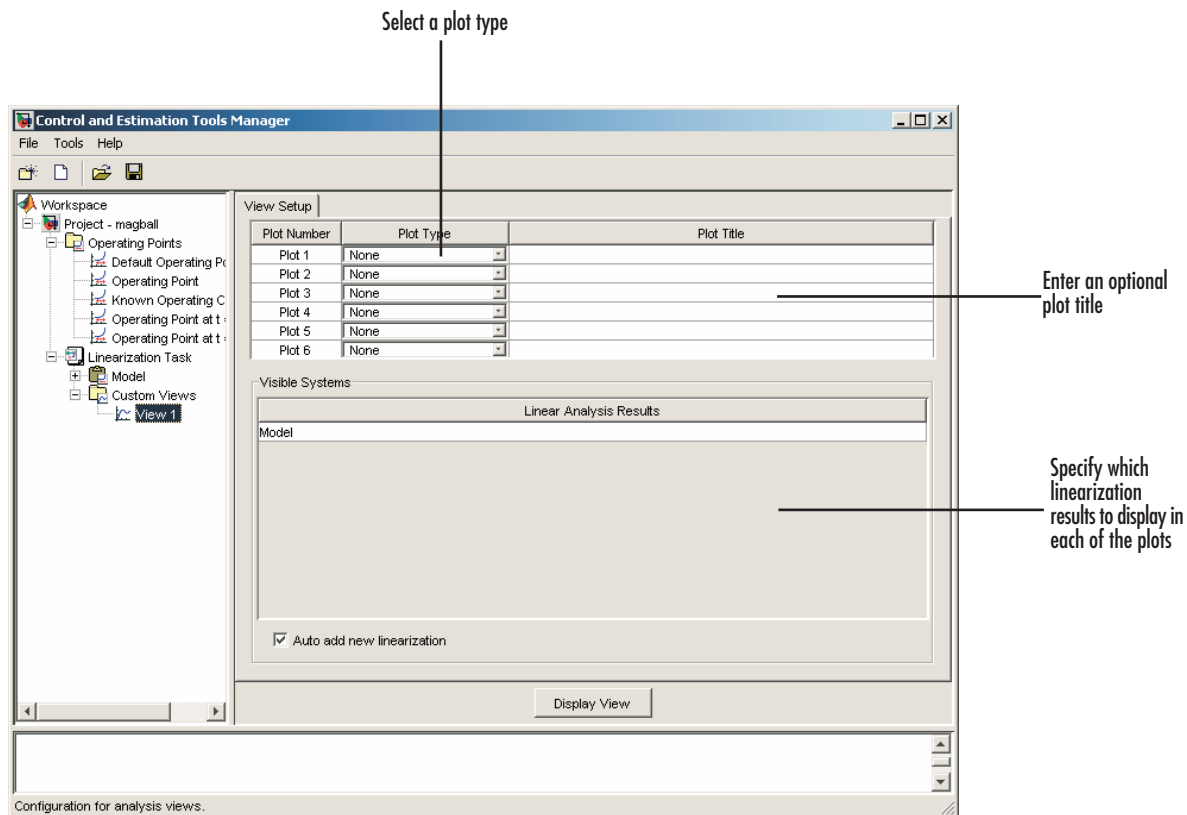
## Linearization Inspector Panel of Simulink Control Design

**Note** The **Linearization Inspector** pane does not exist for models that are linearized with numerical-perturbation linearization (when Numerical perturbation is selected as the **Linearization Algorithm** parameter in the Linearization Task Options window) as there are no individual block linearizations to inspect.

### Displaying the Results

This section continues the example from “Example: The Magnetic Ball” on page 3-4. At this stage in the example, a linearization project has been created, linearization points have been inserted, operating points have been created, and a linearized model has been computed and inspected. See “Inspecting the Linearization Block-by-Block” on page 3-43 for more information on inspecting the linearized model.

The LTI Viewer automatically displays a **Linearization Quick Plot** of the linearization results after the linearization. To create additional, customized plots, right-click the **Custom Views** node under the **Linearization Task** node and select **Add View**. A new view, **View1** is added to the **Custom Views** node. Select **View1** to display the **View Setup** pane and set up this view.



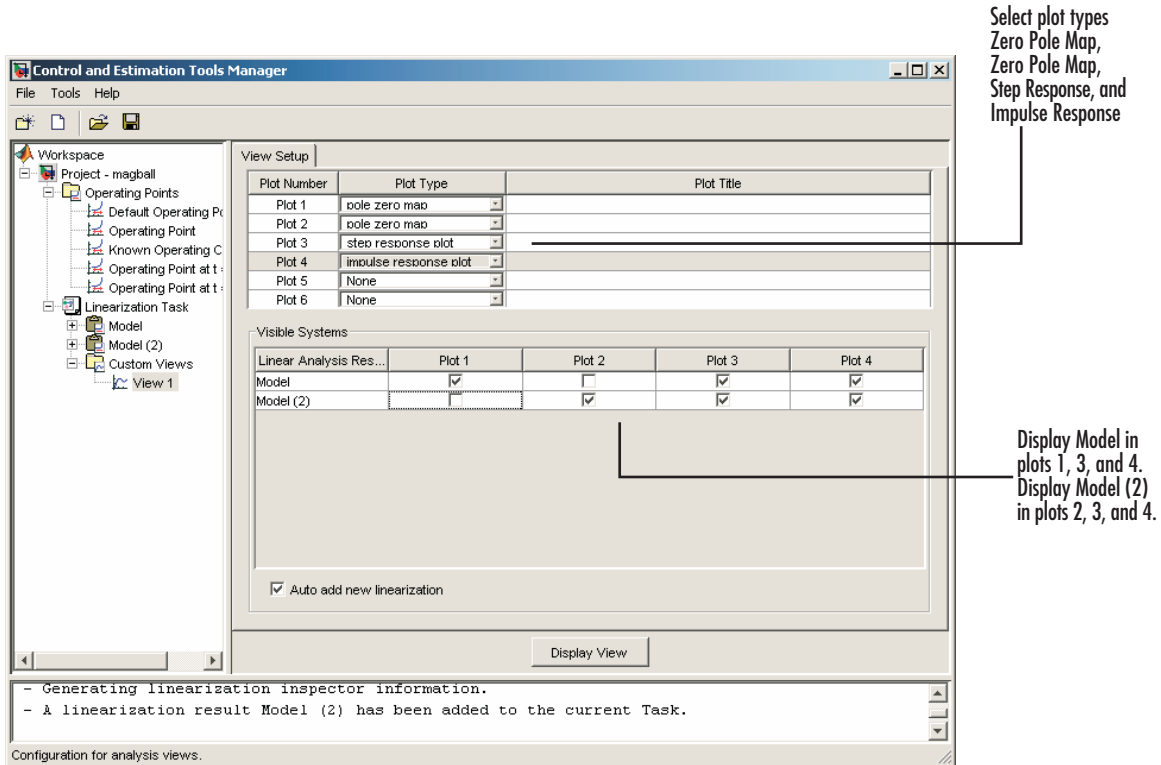
## View Setup Panel

### Example

Before configuring the **View Setup** pane for the magnetic ball system, run another linearization of this model using the operating point at  $t=10$  that you computed in “Extracting Operating Points from Simulation” on page 3-28. This adds **Model (2)** to the **Linearization Task** node.

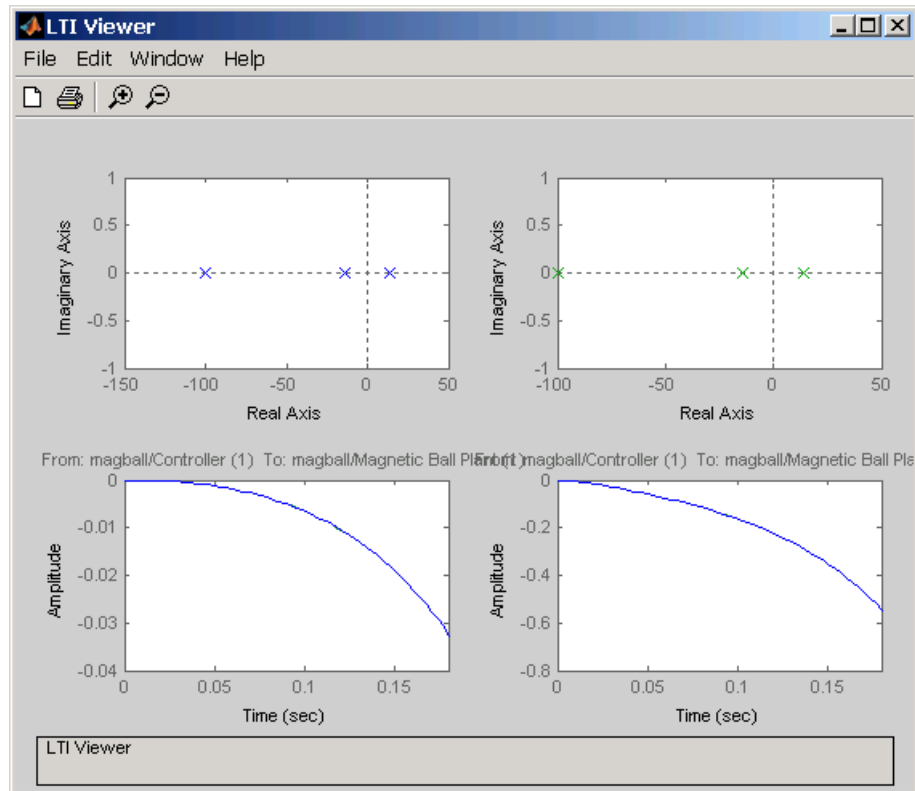
Configure **View Setup** as shown in the following figure.

### 3 Linearizing Models



#### Settings for Custom Plot Example

Click **Display View** to display the LTI Viewer, as shown in the following figure.

**Customized View of Results**

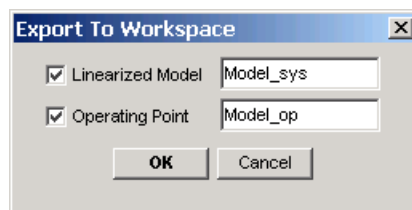
## Exporting and Saving Your Work

The Control and Estimation Tools Manager provides the capability to export your linearization results to the MATLAB workspace, and to save and reopen your linearization projects.

This section continues the example from “Example: The Magnetic Ball” on page 3-4. At this stage in the example, a linearization project has been created, linearization points have been inserted, operating points have been created, and a linearized model has been computed. See “Linearizing the Model” on page 3-35 for more information on computing the linearized model.

### Exporting Results

Export the operating point and results of the linearization to the MATLAB workspace for further analysis using command line functions. To export these results, right-click the results node, **Model**, under the **Linearization Task** node and choose **Export** from the menu. In the Export To Workspace dialog box, choose new names for the linearized model and operating point, or accept the defaults, and then click **OK**.



#### Export Dialog Box

The MATLAB workspace now contains two new objects, `Model_op` and `Model_sys`. To see this, type

```
who
```

at the MATLAB prompt. This returns

```
Your variables are:
```

```
L           Model_sys  beta         m
Model_op    R           g
```

Alternatively, you can export the results to the MATLAB workspace by selecting **File > Export** from the LTI Viewer window, or by clicking the **Export to Workspace** button at the bottom of the **Linearization Summary** panel within the **Model** node.

By right-clicking the results node, **Model**, you can also delete results.

## Exporting and Restoring Linearization I/O Settings

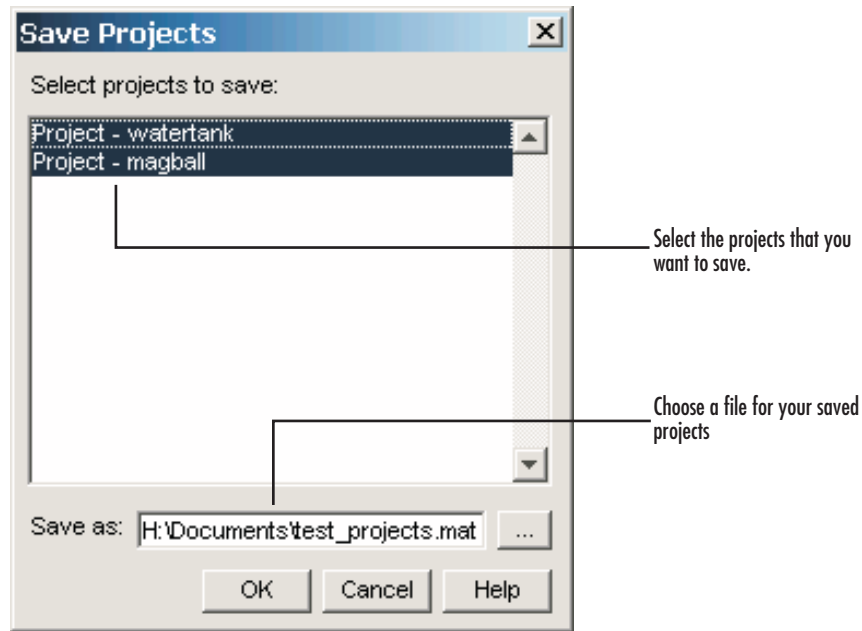
To export linearization I/O settings to the MATLAB workspace, use the `getlinio` function. You can save these settings using the `save` function. Use them in a later session by reloading them with the `load` function. Upload them to the model diagram with the `setlinio` function.

For more information, see the function reference pages for `getlinio` and `setlinio`.

## Saving Control and Estimation Tools Manager Projects

A Control and Estimation Tools Manager project can consist of multiple tasks including those from Simulink Control Design, Simulink Parameter Estimation, and the Model Predictive Control Toolbox. Each task contains data, objects, and results for the analysis of a particular model.

To save your projects, choose **File > Save** in the Control and Estimation Tools Manager window.



#### Save Projects Dialog

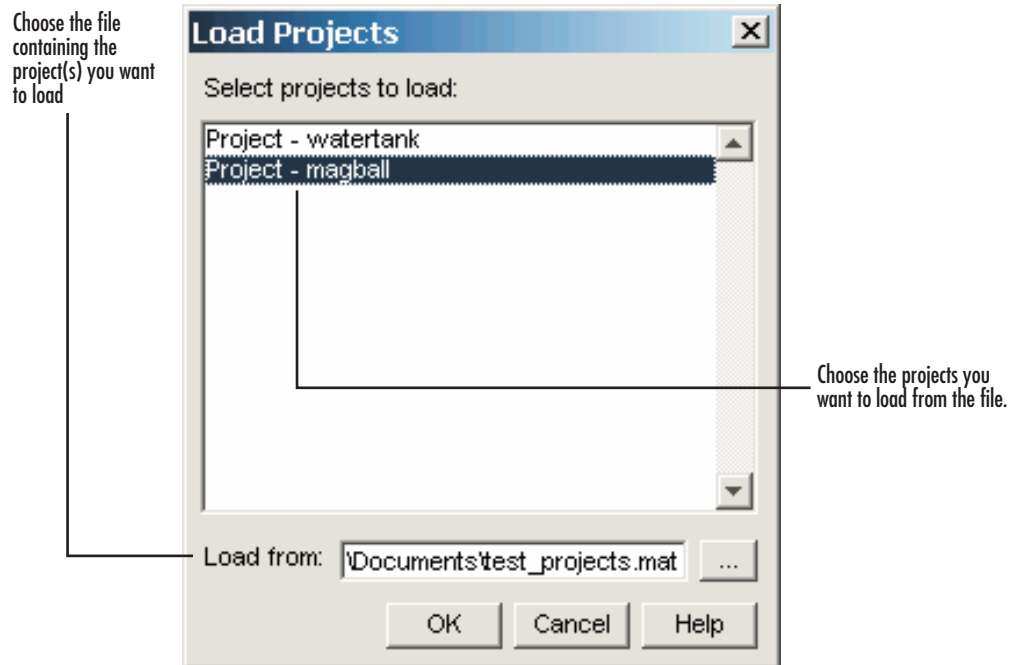
In the Save Projects dialog box, select the projects that you want to save. You can save multiple projects within one file. Next, choose a directory and name for your project file by either browsing for a file or typing the full path and filename in the **Save as** box, and then click **OK**. The project-file is saved as a MAT-file.

### Opening Control and Estimation Tools Manager Projects

A Control and Estimation Tools Manager project can consist of multiple tasks including those from Simulink Control Design, Simulink Parameter Estimation, and the Model Predictive Control Toolbox. Each task contains data, objects, and results for the analysis of a particular model.

To open previously saved projects, choose **File > Load** in the Control and Estimation Tools Manager window.





In the Load Projects dialog box, choose a project-file by either browsing for the directory and file, or typing the full path and filename in the **Load from** box. Project-files are always MAT-files. The projects within this file appear in the list-box. Select the projects that you want to load, then click **OK**. When a file contains multiple projects you can choose to load them all or just a few.



# Linearizing Models Using Functions

---

Use the Simulink Control Design linear analysis functions to linearize Simulink models in the MATLAB Command Window. Use the functions when you want to create M-files to automate the linearization process, or perform *batch* linearization.

Overview (p. 4-2)

An introduction to the process of linearization using functions

Example: Water-Tank System (p. 4-3)

Derivation of a nonlinear system that illustrates features of Simulink Control Design functions

Creating or Opening a Simulink Model (p. 4-5)

Opening the Simulink model of the water-tank example

Configuring the Model for Linearization (p. 4-7)

Functions for selecting and editing linearization points, and performing open loop analysis

Specifying Operating Points Using Functions (p. 4-13)

Specifying the operating points for linearization using exact values, trimmed values, or values from a simulation

Linearizing the Model (p. 4-24)

Running the linearization

Analyzing the Results (p. 4-27)

Inspecting the linearized state-space model of your system. Displaying the results in the LTI Viewer and saving them for later use

# Overview

As discussed in “Purpose of Linearization” on page 1-3, linearization is an important process in the design and analysis of control systems. The main steps involved in the linearization of Simulink models using the Simulink Control Design functions are

- 1 Creating or opening a Simulink model
- 2 Configuring the model
- 3 Specifying operating points
- 4 Linearizing the model
- 5 Analyzing the results and saving your work

The following section introduces an example containing a nonlinear system, a water-filled tank. Subsequent sections of this chapter use the example for a detailed discussion of each step.

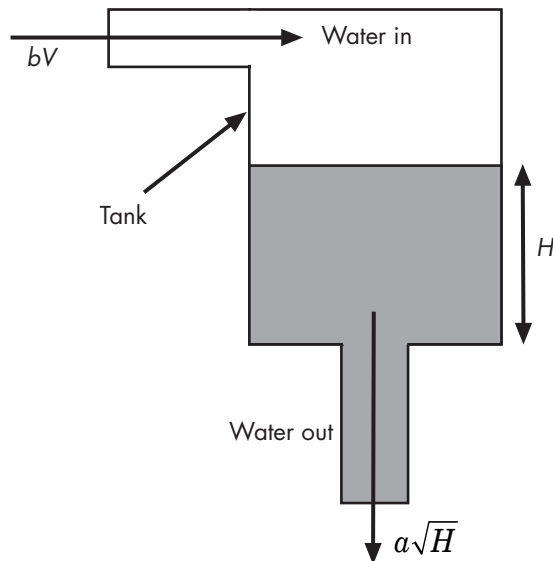
Although this chapter focuses on the Simulink Control Design functions for linearizing models, you can also use the Graphical User Interface (GUI) for some steps in the process. For example, after specifying the operating points in the GUI, you can export the results to the MATLAB workspace and use the functions to continue the analysis. gives detailed information on use of the GUI. For discussion of the advantages and disadvantages of the GUI versus the functions, refer to “Using the GUI vs. Command Line Functions” on page 1-4. A particular advantage of the linearization functions is the ability to write scripts to automate the linearization process or perform *batch linearization*.

## Example: Water-Tank System

This section introduces an example that continues throughout the remaining sections of this chapter. By following this example, you will learn the process of linearizing a model using functions from Simulink Control Design.

### The Water-Tank System

Water enters a tank from the top and leaves through an orifice in its base. The rate that water enters is proportional to the voltage,  $V$ , applied to the pump. The rate that water leaves is proportional to the square root of the height of water in the tank.



### Schematic Diagram for the Water-Tank System

### Model Equations

This section describes the model equations for the example started in the previous section “Example: Water-Tank System” on page 4-3.

A differential equation for the height of water in the tank,  $H$ , is given by

$$\frac{d}{dt} Vol = A \frac{dH}{dt} = bV - a\sqrt{H}$$

where  $Vol$  is the volume of water in the tank,  $A$  is the cross-sectional area of the tank,  $b$  is a constant related to the flow rate into the tank, and  $a$  is a constant related to the flow rate out of the tank. The equation describes the height of water,  $H$ , as a function of time, due to the difference between flow rates into and out of the tank.

The equation contains one state,  $H$ , one input,  $V$ , and one output,  $H$ . It is nonlinear due to its dependence on the square-root of  $H$ . Linearizing the model, using Simulink Control Design, simplifies the analysis of this model. For information on the linearization process, see “Linearization of Nonlinear Models” on page 2-4.

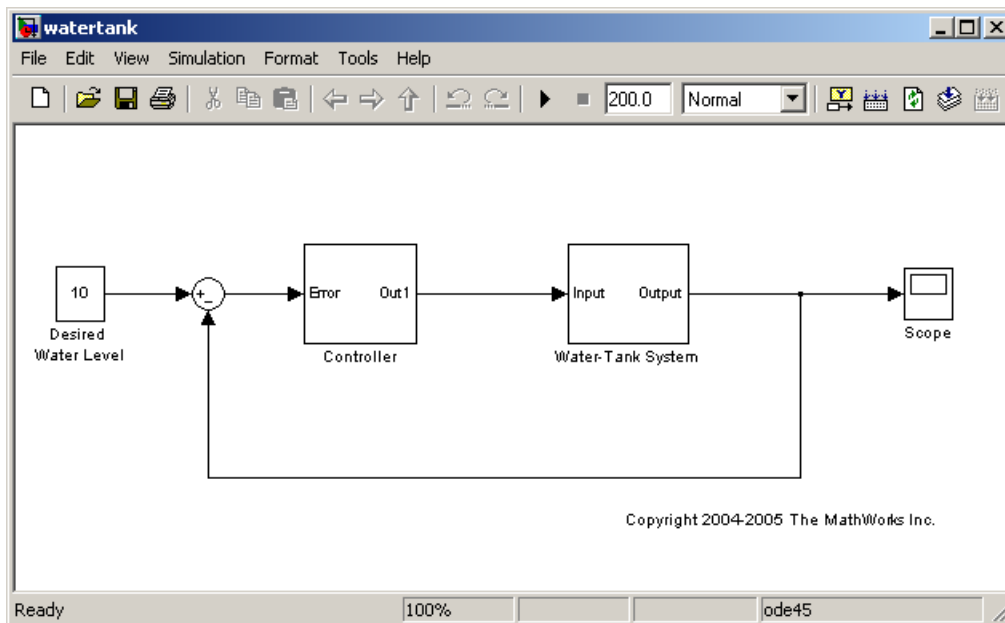
## Creating or Opening a Simulink Model

To begin linearization using functions, you must first create or open a Simulink model of your system. The model can have any number of inputs and outputs (including none) and any number of states. The model can include user-defined blocks or S-functions. Your model can involve a plant only, a plant with a feedback loop and controller, or any number of subsystems.

To continue with the water-tank example, type

```
watertank
```

at the MATLAB prompt. This opens a Simulink model containing the water-tank system as shown in this figure.



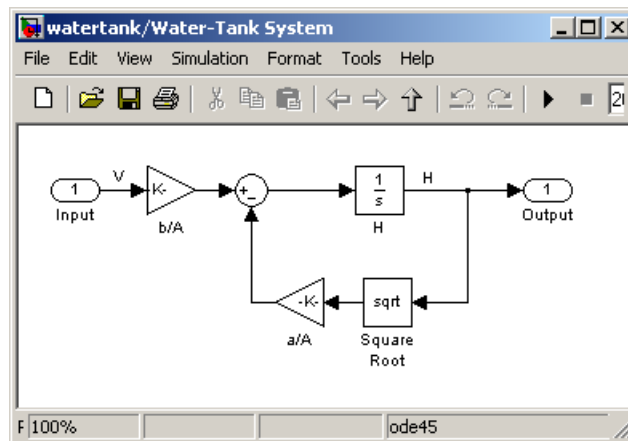
### Simulink Model of the Water-Tank System

The watertank model consists of

- The water-tank system itself

- A Controller subsystem to control the height of water in the tank by varying the voltage applied to the pump
- A reference signal that sets the desired water level
- A Scope block that displays the height of water as a function of time

Double-click a block to view its contents. The Controller block contains a simple proportional-integral-derivative controller. The Water-Tank System block is shown in this figure.



### Water-Tank System Block

The input to the Water-Tank System block, which is also the output of the Controller, is the voltage,  $V$ . The output is the height of water,  $H$ . The system contains just one state (within the integrator),  $H$ . Values of the parameters are given as  $a=2 \text{ cm}^{2.5}/\text{s}$ ,  $A=20 \text{ cm}^2$ ,  $b=5 \text{ cm}^3/(\text{s}\cdot\text{V})$ .



## Configuring the Model for Linearization

This section describes how to configure the model for linearization using functions. For a description of how to use the graphical interface for this task, see “Configuring Inputs and Outputs for the Linearized Model” on page 3-12.

Before linearizing the Simulink model of your system, configure it by

- 1 Choosing linearization input and output points.
- 2 Storing linearization points in an input/output (I/O) object.
- 3 Editing the I/O object, when necessary, such as when computing the open loop model.

The input and output points define the portion of your model being linearized. Setting the `OpenLoop` property of a linearization point to 'on' allows you to compute an open loop model. Refer to “Linearization of Simulink Models” on page 2-8 for more information on linearization input and output points.

### Choosing and Storing Linearization Points

This section continues the example from “Example: Water-Tank System” on page 4-3.

In the `watertank` model, the nonlinearities are in the water-tank system itself. To linearize this portion of the model, place an input point before it and an output point after it. Information about the linearization points is stored in an input/output (I/O) object in the MATLAB workspace.

Each linearization point is associated with an output of a block. To place an input point before the Water-Tank System block, you need to associate this input point with the output of the Controller block.

To create an I/O object for the input point, use the `linio` function.

```
watertank_io(1)=linio('watertank/Controller',1,'in')
```

This creates an object, `watertank_io`, in the MATLAB workspace and displays the object as shown below.

Linearization IOs:

-----

Block watertank/Controller, Port 1 is marked with the following properties:

- No Loop Opening
- An Input Perturbation
- No signal name. Linearization will use the block name

The first input argument of the `linio` function is the name of the block that the linearization point is associated with. The second argument is the number of the output on this block that the linearization point is associated with. These two arguments allow the linearization point to be placed on a specific signal line. The third argument is the type of linearization point. Available types are

'in'	input point
'out'	output point
'inout'	input point followed by output point
'outin'	output point followed by input point

To create a second object within `watertank_io` for an output point, use the following command.

```
watertank_io(2)=linio('watertank/Water-Tank System',1,'out')
```

This creates an I/O object for the output point that is located at the first output of the block `watertank/Water-Tank System`. The object `watertank_io` is displayed, as shown below.

Linearization IOs:

-----

Block watertank/Controller, Port 1 is marked with the following properties:

- No Loop Opening
- An Input Perturbation
- No signal name. Linearization will use the block name

Block watertank/Water-Tank System, Port 1 is marked with the

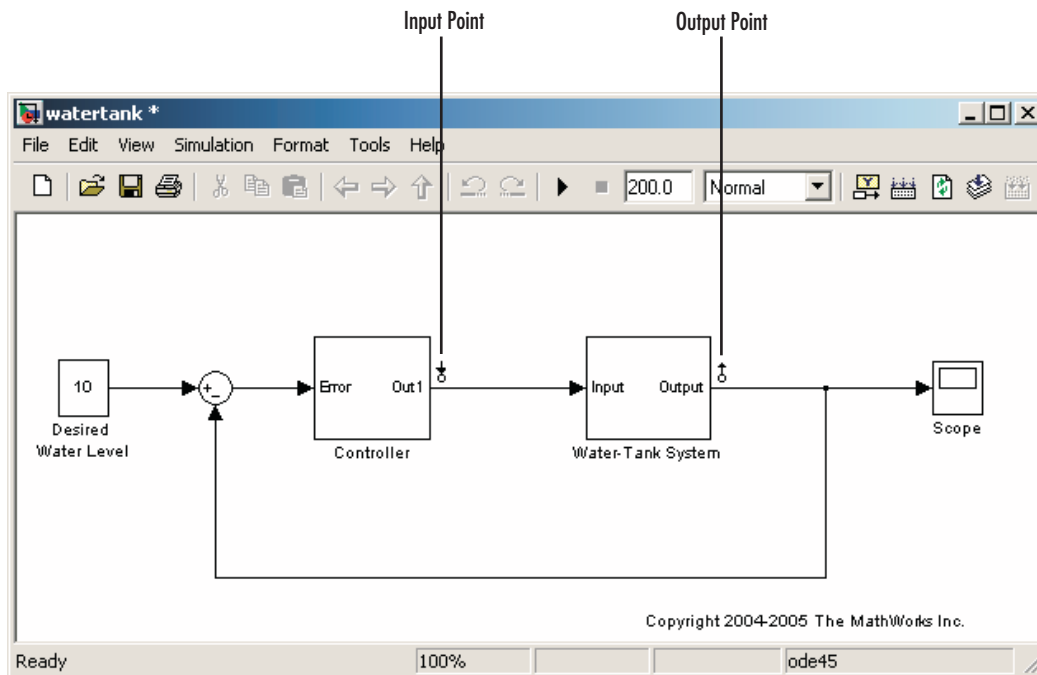
following properties:

- An Output Measurement
- No Loop Opening
- No signal name. Linearization will use the block name

Both the input and output points are now stored in the MATLAB workspace in the I/O object `watertank_io`. To view the linearization points on the model diagram, upload the settings in `watertank_io` using the `setlinio` function.

```
setlinio('watertank', watertank_io)
```

The model diagram should now look like that in the following figure.



**Water-Tank Model with Input and Output Points Selected**

## Extracting Linearization Points from a Model

This section continues the example from “Example: Water-Tank System” on page 4-3. At this stage in the example, linearization points have been

inserted in the model. See “Choosing and Storing Linearization Points” on page 4-7 for more information on inserting linearization points in the model using functions.

An alternative way to create an I/O object is to extract the linearization points from the model diagram when they have been selected using the right-click menus described in “Inserting Linearization Points” on page 3-13. The extracted linearization points are stored in an I/O object. Use the `getlinio` function to extract the linearization points in the following way.

```
watertank_io=getlinio('watertank')
```

This returns

```
Linearization IOs:
-----
Block watertank/Controller, Port 1 is marked with the following
properties:
- No Loop Opening
- An Input Perturbation
- No signal name. Linearization will use the block name

Block watertank/Water-Tank System, Port 1 is marked with the
following properties:
- An Output Measurement
- No Loop Opening
- No signal name. Linearization will use the block name
```

### Editing an I/O Object

This section continues the example from “Example: Water-Tank System” on page 4-3. At this stage in the example, linearization points have been inserted in the model and extracted to an object in the MATLAB workspace. See “Extracting Linearization Points from a Model” on page 4-9 for more information on extracting linearization points from a model using functions.

Typing the name of the I/O object at the command line returns a formatted display of key object properties. To view a list of all properties, use the `get` function. Each object within the I/O object has six properties. For example, to view the properties of the second object in `watertank_io`, type

```
get(watertank_io(2))
```

MATLAB displays

```
    Active: 'on'  
    Block: 'watertank/Water-Tank System'  
    OpenLoop: 'off'  
    PortNumber: 1  
    Type: 'out'  
    Description: ''
```

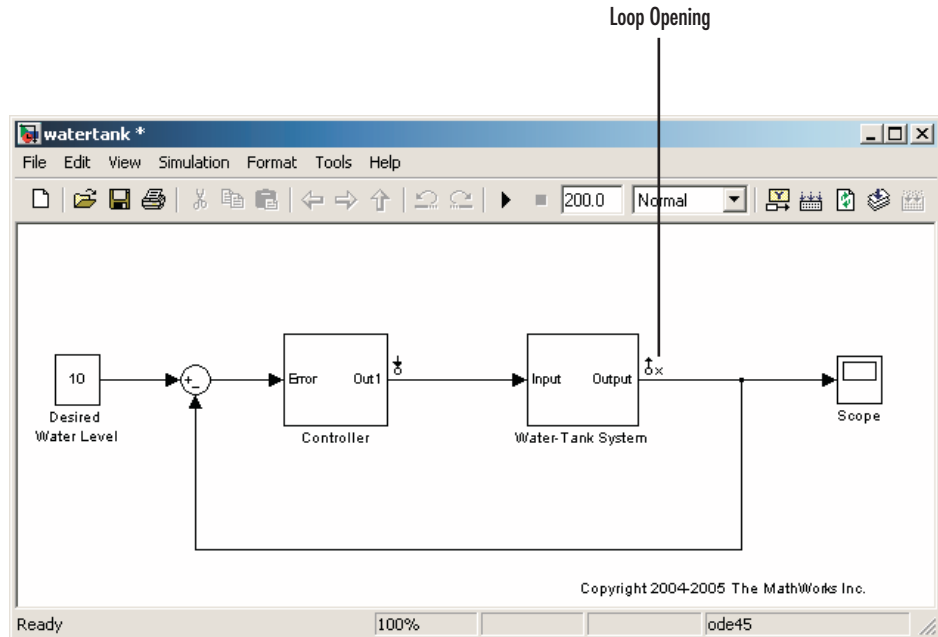
You can edit this object to make any necessary changes. For example, to make this linearization point a loop opening as well, type

```
watertank_io(2).OpenLoop='on'
```

To refresh the model diagram so that it reflects any changes made to the I/O object using the functions, use the `setlinio` function.

```
setlinio('watertank', watertank_io);
```

A small x appears next to the output point in the diagram, indicating a new loop opening, as shown in this figure.



### Water-Tank Model with Loop Opening

You can edit the other properties of I/O objects in a similar way. For more information about each property and the possible values it can take, see the `getlinio` reference page.

### Open Loop Analysis Using Functions

When you want to remove the effect of signals feeding back into the portion of the model you are linearizing, it is often convenient to insert open loop points in the model. For more information on this topic, refer to “Understanding Open Loop Analysis” on page 2-10. For methods on inserting loop openings with the Simulink Control Design GUI, refer to “Performing Open Loop Analysis” on page 3-16. An alternative method of inserting loop openings, using functions, is to edit the I/O object as described in “Editing an I/O Object” on page 4-10.

## Specifying Operating Points Using Functions

This section describes how to specify operating points for a model using functions. For a description of how to use the graphical interface for this task, see “Specifying Operating Points” on page 3-20.

Before linearizing the model, you must choose an operating point to linearize the system about. This is often a steady-state value. Refer to “Operating Points” on page 2-2 for more information on the role of operating points in linearization.

Use the Simulink Control Design functions for any of the following methods of specifying the operating point:

- You do not know all the input and state values, but you can characterize the operating point indirectly by specifying operating point values and constraints for specific signals and variables in the model (implicit specification).
- You know the operating point explicitly, i.e., you know the values of all inputs and states in the model.
- You want to simulate the model and extract the operating point at a given time.

---

**Note** The operating point consists of values for *all* the states in the model although only those states between the linearization points will be linearized. This is because the whole model contributes to the operating point values of the states/inputs/outputs of the portion of the model you are linearizing.

---

## Computing Operating Points from Specifications

This section continues the example from “Example: Water-Tank System” on page 4-3. At this stage in the example, linearization point objects have been created in the MATLAB workspace. See “Configuring the Model for Linearization” on page 4-7 for more information on creating linearization point objects.

To determine the operating points from specifications,

- 1 Create an operating point specification object. See, “Creating an Operating Point Specification Object” on page 4-14.
- 2 Configure the object to store the specifications such as any constraints or known information about the operating point. See “Configuring the Operating Point Specification Object” on page 4-15.
- 3 Use the `findop` function to find the operating point values by optimization. See “Computing the Complete Operating Point” on page 4-16.

### Creating an Operating Point Specification Object

When you know only some values exactly, or you know constraints on the values in the operating point, use the function `operspec` to create an operating point specification object for your model.

For example, to create an operating point specification object for the `watertank` model, type

```
watertank_spec = operspec('watertank')
```

MATLAB displays

```
Operating Specifacaton for the Model watertank.  
(Time-Varying Components Evaluated at time t=0)  
  
States:  
-----  
(1.) watertank/Controller/Integrator  
    spec: dx = 0, initial guess:           0  
(2.) watertank/Water-Tank System/H  
    spec: dx = 0, initial guess:           0  
  
Inputs: None  
  
Outputs: None
```



## Configuring the Operating Point Specification Object

The operating point specification object contains objects for all the states, inputs, and outputs in the model. By typing the object's name at the command line you can see a formatted display of key object properties. Alternatively, to list all the properties for a particular object, use the `get` function. For example

```
get(watertank_spec.States(1))
```

returns

```

                Block: 'watertank/Controller/Integrator'
                  x: 0
                 Nx: 1
                 Ts: [0 0]
        SampleType: 'CSTATE'
inReferencedModel: 0
                 Known: 0
        SteadyState: 1
                 Min: -Inf
                 Max: Inf
        Description: ''

```

Edit these properties to provide specifications for the operating point. For example:

- To set the second state to a known value (such as the desired height of water), first change `Known` to 1.

```
watertank_spec.States(2).Known=1
```

Next, provide the known value.

```
watertank_spec.States(2).x=10
```

- To find a steady-state value for the first state, set `SteadyState` to 1.

```
watertank_spec.States(1).SteadyState=1
```

- To provide an initial guess of 2 for this steady-state value, first make sure that `Known` is set to 0 for this state.

```
watertank_spec.States(1).Known=0
```

Then, provide the initial guess.

```
watertank_spec.States(1).x=2
```

- To set a lower bound of 0 on this state,

```
watertank_spec.States(1).Min=0
```

Optimization settings used with the `findop` function can be configured with the `linoptions` function.

### Computing the Complete Operating Point

The operating point specification object, `watertank_spec`, now contains specifications for the operating point. Use this information to find the complete operating point using the `findop` command. Type

```
[watertank_op,op_report]=findop('watertank',watertank_spec)
```

This returns the optimized operating point. The optimized values of the states are contained in the `x` property, or `u` property for inputs.

```
Operating Point for the Model watertank.  
(Time-Varying Components Evaluated at time t=0)
```

```
States:
```

```
-----
```

```
(1.) watertank/Controller/Integrator
```

```
    x: 1.26
```

```
(2.) watertank/Water-Tank System/H
```

```
    x: 10
```

```
Inputs: None
```

The operating point search report, `op_report`, is also generated. The `x` or `u` values give the state or input values. The `dx` values give the time derivatives of each state, with desired `dx` values in parentheses. The fact that the `dx` values are both zero indicates that the operating point is at steady state.

Operating Point Search Report:

-----

Operating Point Search Report for the Model watertank.  
(Time-Varying Components Evaluated at time t=0)

Operating condition specifications were successfully met.

States:

-----

(1.) watertank/Controller/Integrator			
x:	1.26	dx:	0 (0)
(2.) watertank/Water-Tank System/H			
x:	10	dx:	0 (0)

Inputs: None

Outputs: None

### Alternative Method for Specifying Initial Guesses

In some cases you might want to use a previously created operating point to specify initial guesses in another operating point specification object. For example, after extracting an operating point from a simulation, as in “Extracting Values from Simulation” on page 4-21, you can use this operating point as a starting point for finding a more accurate steady state value using `findop`. You can do this with the `initopspec` function.

For example, first extract an operating point from simulation, in this case after 10 time units.

```
watertank_op2=findop('watertank',10);
```

Then create an operating point specification object.

```
watertank_spec=operspec('watertank');
```

Specify initial guesses in this object with the following command.

```
watertank_spec=initopspec(watertank_spec,watertank_op2)
```

This returns an operating point specification with the initial guess, or `x` property filled with operating point values from `watertank_op2`.

```
Operating Specifcaton for the Model watertank.  
(Time-Varying Components Evaluated at time t=0)
```

```
States:
```

```
-----
```

```
(1.) watertank/Controller/Integrator  
    spec: dx = 0, initial guess:      0.872  
(2.) watertank/Water-Tank System/H  
    spec: dx = 0, initial guess:      9.7
```

```
Inputs: None
```

```
Outputs: None
```

This operating point specification can now be used with `findop` to find an optimized steady state operating point. You can access individual elements of this object using the `get` function or dot-notation as in “Configuring the Operating Point Specification Object” on page 4-15.

### Adding Output Constraints to Specifications

When you want to constrain additional signals of the model, you can add an output constraint to the operating point specification object with the function `addoutputspec`.

For example, to add an output constraint to the operating point specification created in “Alternative Method for Specifying Initial Guesses” on page 4-17, use the following command.

```
watertank_spec=addoutputspec(watertank_spec, 'watertank/Water-Tank System/Sum',1)
```

This adds a constraint on the signal between the Sum block and the integrator block, H, within the Water-Tank System block. The constraint is associated with an outport of a block, in this case outport 1 of the block 'watertank/Water-Tank System/Sum' (the preceding block).

```
Operating Specifcaton for the Model watertank.
```

```

(Time-Varying Components Evaluated at time t=0)

States:
-----
(1.) watertank/Controller/Integrator
    spec: dx = 0, initial guess:      0.872
(2.) watertank/Water-Tank System/H
    spec: dx = 0, initial guess:      9.7

Inputs: None

Outputs:
-----
(1.) watertank/Water-Tank System/Sum
    spec: none

```

You can edit the specifications for this output in the same way as you would for any other specifications, by changing the values of Known, y, Min, etc. There is no SteadyState option for outputs.

## Specifying Completely Known Operating Points

This section continues the example from “Example: Water-Tank System” on page 4-3. At this stage in the example, linearization point objects have been created in the MATLAB workspace. See “Configuring the Model for Linearization” on page 4-7 for more information on creating linearization point objects.

To use functions to specify completely known operating points, first create an operating point object, and then make changes to the object values.

### Creating an Operating Point Object

An operating point object contains information about your system’s states and inputs at the operating point. When you know your operating point explicitly, use the function `operpoint` to create an operating point object for your model.

For example, to create an operating point object for the water-tank model, type

```
watertank_op=operpoint('watertank')
```

MATLAB displays

```
Operating Point for the Model watertank.  
(Time-Varying Components Evaluated at time t=0)
```

```
States:
```

```
-----
```

```
(1.) watertank/Controller/Integrator
```

```
    x: 0
```

```
(2.) watertank/Water-Tank System/H
```

```
    x: 0
```

```
Inputs: None
```

Within the operating point object are objects for all the states and inputs in the model. Each of these objects has a property called `x`, or `u` in the case of inputs, that gives the value of that state or input.

### Changing Operating Point Values

Change the `x` and `u` properties of the operating point object to known values at the operating point. For example, to change the value of the first state to 1.26 and the second state to 10, type

```
watertank_op.States(1).x=1.26, watertank_op.States(2).x=10
```

which returns

```
Operating Point for the Model watertank.  
(Time-Varying Components Evaluated at time t=0)
```

```
States:
```

```
-----
```

```
(1.) watertank/Controller/Integrator
```

```
    x: 1.26
```

```
(2.) watertank/Water-Tank System/H
```

```
    x: 10
```

```
Inputs: None
```

The operating point object, `watertank_op`, now contains the known operating point of the system.

## Extracting Values from Simulation

This section continues the example from “Example: Water-Tank System” on page 4-3. At this stage in the example, linearization point objects have been created in the MATLAB workspace. See “Configuring the Model for Linearization” on page 4-7 for more information on creating linearization point objects.

Use Simulink Control Design to extract operating points from a simulation of your model at specified times, such as when the simulation reaches a steady state solution.

For example, to create an operating point object for the water-tank model after it has simulated for 20 time units, type

```
watertank_op=findop('watertank',20)
```

MATLAB displays the operating point at time `t=20`.

```
Operating Point for the Model watertank.
(Time-Varying Components Evaluated at time t=20)
```

```
States:
```

```
-----
```

```
(1.) watertank/Controller/Integrator
```

```
    x: 1.25
```

```
(2.) watertank/Water-Tank System/H
```

```
    x: 9.99
```

```
Inputs: None
```

## Using Structures and Vectors of Operating Point Values

This section continues the example from “Example: Water-Tank System” on page 4-3. At this stage in the example, linearization point objects and operating points have been created in the MATLAB workspace. See

“Specifying Operating Points Using Functions” on page 4-13 for more information on creating operating point objects using functions.

Operating point objects store the operating point values. However, when you want to use an operating point’s values to initialize the simulation of a model, it is useful to work with *vectors* of operating point values, or with Simulink structures. Simulink structures have the benefits that you can use them to initialize models that reference other models using the Model block, and you do not need to worry about the ordering of states in the structure.

You can extract vectors and structures of operating point values from operating point objects using the functions `getxu` and `getstatestruct`. You can then use these vectors or structures to initialize a model for simulation. Models that reference other models using the Model block, must be initialized with a Simulink structure of values, such as those from simulation data, extracted with the `getstatestruct` function. See [Importing and Exporting States in the Simulink documentation](#) for details on initializing model reference models.

To extract a structure of operating point values from the operating point object, `watertank_op`, created in “Extracting Values from Simulation” on page 4-21, use the following command.

```
x=getstatestruct(watertank_op)
```

This extracts a structure of state values, `x` from the operating point object:

```
x =  
    time: 0  
  signals: [1x2 struct]
```

To access the values within this structure, use the following syntax

```
x.signals.values
```

which returns

```
ans =  
    1.2469  
  
ans =
```



9.9927

Note that these values are in the same order as those used by Simulink.

To extract a vector of operating point values from the operating point object, `watertank_op`, use the following command.

```
[x,u]=getxu(watertank_op)
```

This extracts vectors of states, `x`, and inputs, `u`, as shown below.

```
x =  
    9.9927  
    1.2469
```

```
u =  
    []
```

To create an operating point object from a vector, or structure, of values, such as those returned from a simulation, you can use the function `setxu`. To set operating point values in an operating point object using a vector or structure of known values, you can use the following command.

```
new_op=setxu(watertank_op2,x,u)
```

This command creates a new operating point, `new_op`, that is based on the operating point `watertank_op2`, but with the values from the vector or structure of state values, `x`, and the vector of input values, `u`.

The ordering of the states in `x` and the inputs in `u` must be the same as the ordering used by Simulink which is not necessarily the same as the order the states appear in the operating point object. When you extract values from simulation data they will already be in the correct order.

## Linearizing the Model

This section describes how to linearize the model using functions. For a description of how to use the graphical interface for this task, see “Linearizing the Model” on page 3-35.

This section also continues the example from “Example: Water-Tank System” on page 4-3. At this stage in the example, linearization point objects and operating point have been created in the MATLAB workspace. See “Specifying Operating Points Using Functions” on page 4-13 for more information on creating operating point objects using functions.

After creating an I/O object and determining the operating point, you are ready to linearize the system, using the `linearize` command. For example:

```
watertank_lin=linearize('watertank',watertank_op,watertank_io)
```

MATLAB will return the matrices a, b, c, and d of a linear, time-invariant, state-space model that approximates your nonlinear system in a region around the operating point.

```
a =
      H
      H -0.01582

b =
      Controller (
      H          0.25

c =
      H
      Water-Tank S 1

d =
      Controller (
      Water-Tank S 0
```

Continuous-time model.

To change the linearization options, use the `linoptions` function before running the linearization. For example, to change the sample time for the linearization model to be 1 instead of continuous, use the following command.

```
linopt=linoptions('SampleTime',1);
```

Then, run the linearization with these options.

```
watertank_lin2=linearize('watertank',watertank_op,watertank_io,linopt)
```

This returns the discrete-time model shown below.

```
a =
      H
H  0.9843
```

```
b =
      Controller (
H      0.248
```

```
c =
      H
Water-Tank S  1
```

```
d =
      Controller (
Water-Tank S      0
```

```
Sampling time: 1
Discrete-time model.
```

## Linearizing Discrete-Time and Multi-Rate Models

The linearization method is the same for models containing discrete-time states or several different sampling rates. However, you can choose to adjust the `SampleTime` parameter with the `linoptions` function as shown in the

previous section. By default this parameter is set to -1, in which case Simulink Control Design will find the slowest sample rate in the model to use for the sample rate of the linearized model. To create a linearized model with different sample time, specify a new parameter value before linearizing the model. A value of 0 will give a continuous-time model.

## Analyzing the Results

This section describes how to analyze the linearization results using functions. For a description of how to use the graphical interface for this task, see “Analyzing the Results” on page 3-41.

Analyze the linearized model at the MATLAB prompt using functions from the Control System Toolbox, or display it in the LTI Viewer. Alternatively, incorporate the results into a block in a Simulink model. For methods on simulating the linearized model for comparison with the original model, refer to “Comparing the Linearized and Original Models”.

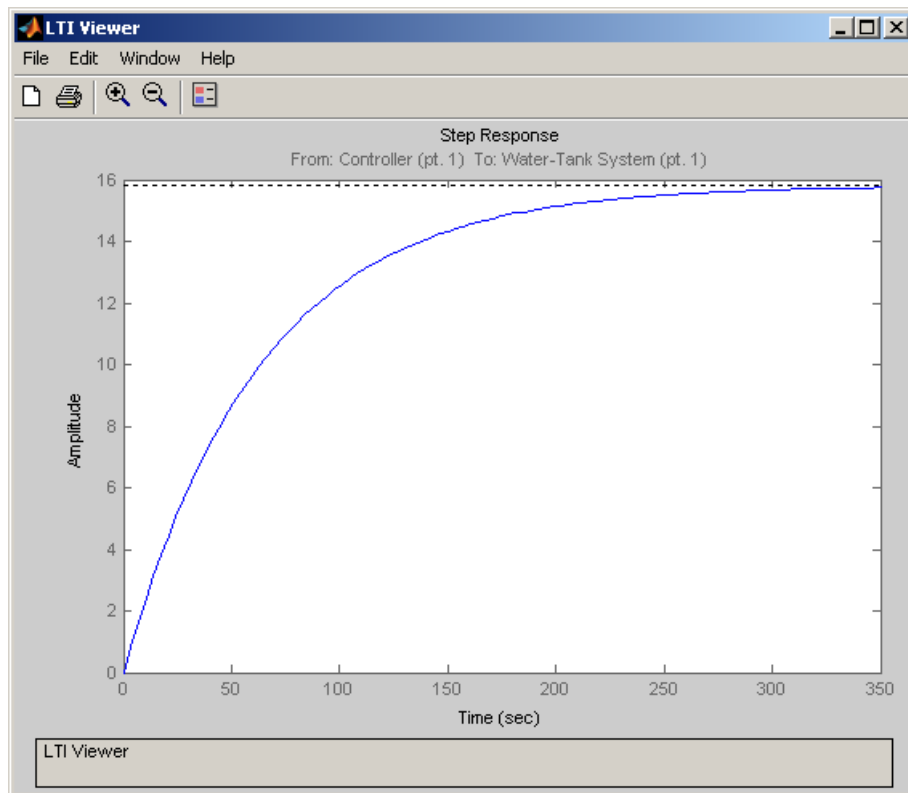
### Using the LTI Viewer

This section continues the example from “Example: Water-Tank System” on page 4-3. At this stage in the example, linearization point objects and operating point have been created in the MATLAB workspace, and a linearized model has been computed. See “Linearizing the Model” on page 4-24 for more information on computing a linearized model using functions.

To send your linearized model to the LTI Viewer for display, type

```
ltiview(watertank_lin)
```

The LTI Viewer opens, by default, with a step response of the linearized system, as shown in the following figure.



### LTI Viewer Displaying a Step Response of the Linearized Model

You can use standard LTI Viewer features to display your results. For example, change the plot type by right-clicking anywhere in the plot area and choosing from the **Plot Types** menu. To add characteristics such as settling time or peak response to your plot, right-click anywhere in the plot area and choose from the **Characteristics** menu. Add data markers by clicking the point you want to mark.

You can display up to six plots in the LTI Viewer window. To change the number of plots, select **Edit > Plot Configurations**, choose a configuration in the Plot Configurations window, and then click **OK**.

For more information on the LTI Viewer, refer to the Control System Toolbox documentation.

## Saving Your Work

This section continues the example from “Example: Water-Tank System” on page 4-3. At this stage in the example, linearization point objects and operating point have been created in the MATLAB workspace, and a linearized model has been computed. See “Linearizing the Model” on page 4-24 for more information on computing a linearized model using functions.

This section describes how to save a linearization project using functions. For a description of how to use the graphical interface for this task, see “Saving Control and Estimation Tools Manager Projects” on page 3-51.

To save your linearized model for later analysis, use the save command. For example, to save the linearized model, operating points, and I/O object of the watertank model, type

```
save watertank_project watertank_lin watertank_op watertank_io
```

This creates a file named `watertank_project.mat` in the current directory. To reload this file, use the load function.

```
load watertank_project
```

## Restoring Linearization I/O Settings

To save linearization I/O settings for use in a later session, use the save function. You can then restore the settings by loading them with the load function and using the `setlinio` function to upload them to the model diagram. For more information, see the function reference page for `setlinio`.

Alternatively, you can use the reloaded I/O settings object with the `linearize` function without uploading it to the model diagram.





## A

- addoutputspec function
  - example 4-18
- analysis I/O\ud5 s
  - See input and output points 3-13

## C

- constraining outputs 3-25
  - using functions 4-18
- Control and Estimation Tools Manager
  - opening 3-9
  - overview 3-10

## D

- discrete-time models
  - linearizing using functions 4-25
  - linearizing with GUI 3-38

## E

- equilibrium operating points 2-3
  - computing 3-21
- equilibrium states
  - computing 3-21

## F

- findop function
  - example using operating point specification objects 4-16
  - example using simulation 4-21

## G

- get function
  - example with I/O objects 4-10
- getlinio function
  - example 4-9
- getstatestruct function

- example 4-21
- getxu function
  - example 4-21

## I

- initial guesses
  - specifying with functions 4-17
- initializing simulations 3-33
  - using structures or vectors 4-21
- initopspec function
  - example 4-17
- inport and outport blocks
  - difference from input and output points 3-13
- input and output points
  - editing I/O objects 4-10
  - inspecting with GUI 3-18
  - purpose 3-12
  - removing 3-16
  - specifying 3-13
  - specifying using functions 4-7
- input points
  - See input and output points 3-13

## L

- linearization
  - beginning a new project 3-9
  - benefits of command-line functions 1-4
  - benefits of GUI 1-4
  - custom plots 3-46
  - discrete-time models 3-38
  - factors affecting 1-3
  - highlighting blocks 3-42
  - inspecting 3-43
  - multi-rate models 3-38
  - of blocks 3-39
  - overview 3-3
  - purpose of 1-3

- results 3-41
- using functions 4-24
- using GUI 3-35
- linearization method
  - for nonlinear models 2-4
  - for Simulink models 2-8
- linearization options
  - using functions 4-25
- linearization points
  - adding 3-13
  - deleting 3-16
  - removing 3-16
  - selecting 3-13
- linearization projects
  - loading using load function 4-29
  - opening 3-52
  - saving 3-51
- linearization results
  - custom plots 3-46
  - displaying using `ltiview` function 4-27
  - exporting 3-50
  - in LTI viewer 3-37
  - inspecting 3-43
  - saving using `save` function 4-29
- `linearize` function
  - example 4-24
- `linio` function
  - example 4-7
- `linoptions` function
  - example 4-25
- loop opening
  - purpose 2-10

## M

- magnetic ball
  - equations 3-4
  - example 3-4
  - Simulink model 3-6
- multi-rate models

- linearizing using functions 4-25
- linearizing with GUI 3-38

## O

- open loop analysis 3-16
  - using functions 4-12
- operating point objects
  - creating 4-19
  - editing 4-20
- operating point specification objects
  - creating 4-14
  - editing 4-15
- operating point specifications
  - importing initial values 3-25
- operating point structures
  - initializing simulations with 4-21
- operating points
  - computing equilibrium 3-21
  - computing from specifications 3-21
  - constraining outputs 3-25
  - copying 3-32
  - creating vectors 4-21
  - definition 2-2
  - equilibrium 2-3
  - exporting 3-33
  - extracted from simulation 3-28
  - importing 3-31
  - initializing simulations with 3-33
  - specifying with completely known values 3-26
  - specifying with functions 4-13
  - specifying with GUI 3-20
  - steady state 2-3
  - trimming 3-21
- operating points specifications
  - using functions 4-13
- `operpoint` function
  - example 4-19
- `operspec` function

example 4-14  
output constraints  
    See Constraining Outputs 3-25  
output points  
    See input and output points 3-13

## **P**

project  
    linearization 3-9

## **S**

set function  
    example with I/O objects 4-11  
    example with operating point objects 4-20  
    example with operating point  
        specifications 4-15  
setlinio function

example 4-9  
setxu function  
    example 4-23  
Simulink Control Design 1-2  
steady state operating points 2-3  
    computing 3-21

## **T**

trimming  
    operating points 3-21

## **W**

water-tank system  
    equations 4-3  
    example 4-3  
    Simulink model 4-5