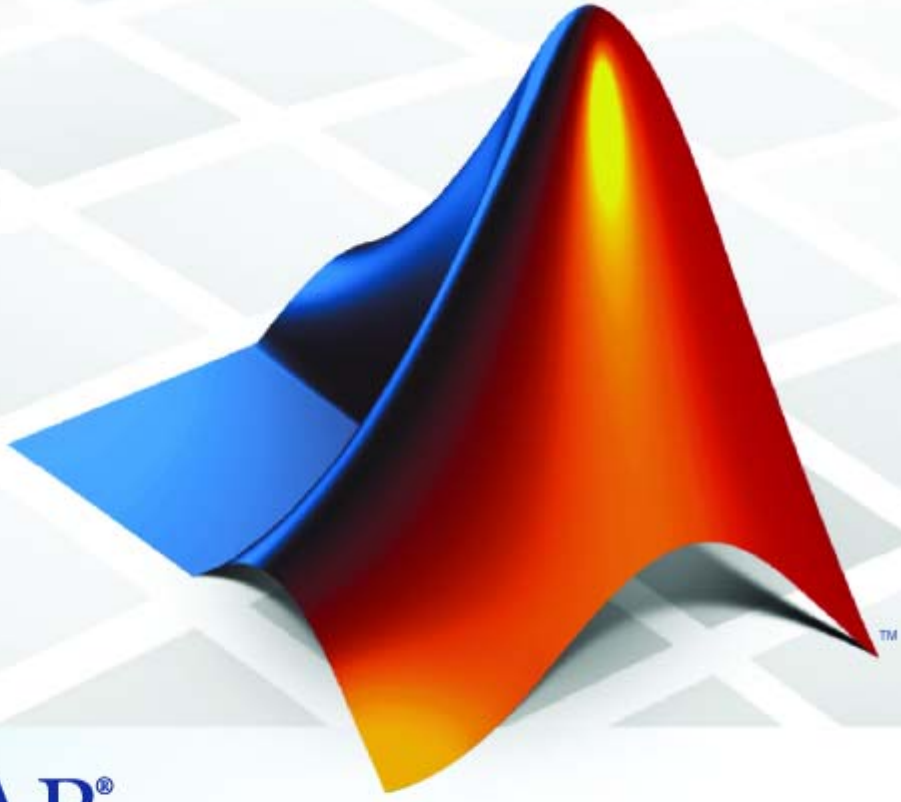
Simulink® Design Optimization™ 1

Getting Started Guide



MATLAB®
&SIMULINK®

The MathWorks™
Accelerating the pace of engineering and science

How to Contact The MathWorks

`www.mathworks.com`	Web
`comp.soft-sys.matlab`	Newsgroup
`www.mathworks.com/contact_TS.html`	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.
3 Apple Hill Drive
Natick, MA 01760-2098

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

Simulink® Design Optimization™ Getting Started Guide

© COPYRIGHT 1993–2009 by The MathWorks, Inc.

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

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

Trademarks

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

Revision History

March 2009	Online only	New for Version 1 (Release 2009a)
September 2009	Online only	Revised for Version 1.1 (Release 2009b)

Contents

1 Product Overview

What You Can Accomplish Using This Product 1-2

Upgrading from an Earlier Release 1-4

Required and Related Products 1-5

Documentation and Demos 1-6
Accessing Documentation 1-6
Accessing Demos 1-6

2 Estimating Model Parameters

Types of Data for Parameter Estimation 2-2

Quick Start — Estimating Model Parameters 2-4

Parallel Computing for Parameter Estimation 2-12

3 Optimizing Model Parameters

Types of Time-Domain Design Requirements for Optimizing Parameters 3-2

Quick Start — Optimizing Model Parameters 3-3

Parallel Computing for Parameter Optimization **3-11**

4 Optimization-Based Linear Control Design

When to Use Optimization-Based Linear Control Design **4-2**

Types of Time- and Frequency-Domain Design Requirements **4-3**

Quick Start — Optimization-Based Linear Control Design **4-4**

5 Tutorial — Preparing Data for Parameter Estimation Using the GUI

About This Tutorial **5-2**
Objectives **5-2**
About the Sample Data **5-2**

Configuring a Project for Parameter Estimation **5-4**

Importing Data into the GUI **5-6**
Importing Input Data and Time Vector **5-6**
Importing Output Data and Time Vector **5-11**

Analyzing Data **5-14**

Selecting Data for Estimation **5-16**
Selecting Output Data **5-16**
Selecting Input Data **5-22**

Removing Outliers 5-25
Why Remove Outliers 5-25
How to Remove Outliers 5-25

Filtering Data 5-29
Filtering Output Data 5-29
Filtering Input Data 5-32

Interpolating Missing Data 5-34

Saving the Project 5-37

6 Tutorial — Estimating Parameters from Measured Data Using the GUI

About This Tutorial 6-2
Objectives 6-2
About the Model 6-3

Estimating Model Parameters Using Default Estimation Settings 6-7
Overview of the Estimation Process 6-7
Specifying Parameters and Estimation Data 6-8
Validating Model Parameters 6-13

Improving Estimation Results Using Parameter Bounds 6-20
Strategy for Improving the Estimation Results 6-20
How to Specify Parameter Bounds 6-20

Validating Estimated Model Parameters 6-26

Tutorial — Optimizing Parameters to Meet Time-Domain Requirements Using the GUI

7

About This Tutorial **7-2**
Objectives **7-2**
About the Model **7-2**
Design Requirements **7-4**

Configuring a Model for Optimizing Parameters **7-5**

Optimizing Model Parameters to Meet Step Response Requirements **7-8**
Specify Time-Domain Design Requirements **7-8**
Specifying Parameters to Optimize **7-18**
Optimizing the Parameters **7-21**

Refining Model Parameters to Track a Reference Signal **7-25**

Saving the Project **7-31**

Tutorial — Optimizing Parameters to Meet Time-Domain Requirements Using the Command Line

8

About This Tutorial **8-2**
Objectives **8-2**
About the Model **8-2**
Design Requirements **8-4**

Configuring a Model for Optimizing Parameters **8-5**

Optimizing Model Parameters to Meet Step Response Requirements **8-8**

Refining Model Parameters to Track a Reference Signal 8-15

9 Tutorial — Designing a PID Controller Using Optimization-Based Tuning

About This Tutorial 9-2
Objectives 9-2
About the Model 9-2
Design Requirements 9-4

Configuring a Project for Optimization-Based Control Design 9-5

Designing an Initial PID Controller to Meet Bode Magnitude and Phase Margins Requirements 9-11
Specifying the Controller Parameters 9-11
Specifying Bode Magnitude and Phase Margin Design Requirements 9-15
Designing the Controller 9-25

Refining the Controller Design to Meet Controller Output Bounds 9-32

Saving the Project 9-48

10 Tutorial — Modeling a System Using Adaptive Lookup Table

About This Tutorial 10-2
Objectives 10-2
About the Data 10-2

Overview of Modeling a System Using Adaptive Lookup Table 10-3

Building a Model Using Adaptive Lookup Table Blocks **10-4**

Adapting the Lookup Table Values Using Time-Varying I/O Data **10-16**

Examples

A

Getting Started **A-2**

Index

1

Product Overview

- "What You Can Accomplish Using This Product" on page 1-2
- "Upgrading from an Earlier Release" on page 1-4
- "Required and Related Products" on page 1-5
- "Documentation and Demos" on page 1-6

What You Can Accomplish Using This Product

Simulink® Design Optimization™ software is a Simulink®-based product that lets you improve designs by estimating and optimizing model parameters using numerical optimization. You can increase model accuracy by using measured data to estimate parameters and then improve system performance by automatically optimizing parameters in your Simulink model.

The software uses methods from the Optimization Toolbox™ and Genetic Algorithm and Direct Search Toolbox™ products to estimate and optimize model parameters. You can select multiple parameters for estimation and optimization, and the parameters can be scalars, vectors, matrices or fields of structured variables defined in the MATLAB® or model workspace.

You can perform the following types of estimation using the Simulink Design Optimization software:

- Transient Estimation — Estimate parameters by comparing simulated model output to measured transient data.
- Initial Condition Estimation — Estimate the initial conditions of states from measured data.
- Adaptive Lookup Table Estimation— Estimate the lookup table values using time-varying data from a physical system. The software provides Adaptive Lookup Table blocks to model and estimate such types of systems.

Simulink Design Optimization software also lets you optimize model parameters to meet time-domain design requirements. The software provides a Signal Constraint block where you can graphically specify bounds on signals or a reference signal to track. The software also provides CRMS and DRMS blocks to compute the continuous and discrete root mean square values of signals. Use these blocks with the Signal Constraint block to optimize the root mean square value of the signal. You can also test and optimize the design for robustness by specifying uncertainty bounds on additional model parameters.

When you have the Control System Toolbox™ software installed, you can also refine controllers by optimizing the controller parameters in the SISO Design Tool. You can perform optimization-based control design for LTI models and Simulink models linearized using the Simulink® Control Design™ software. You can graphically specify both time- and frequency-domain requirements

on open- and closed-loop responses. When you use optimization-based control design, the controller parameters can be poles, zeros, and gains or Simulink block parameters.

You can work either in the Graphical User Interface (GUI) or at the command line to estimate and optimize model parameters. New users should start by using the GUI to become familiar with the product. The following operations are available only using the GUI:

- Interactive data preprocessing (see “Data Analysis and Processing”)
- Model validation using residual plot (see “Comparing Residuals”)
- Optimization-based controller design for time- and frequency-domain requirements (see “Optimization-Based Linear Compensator Design”)

Using Simulink Design Optimization software does not require that you have a strong background in optimization theory or practice. As you gain familiarity with the product, you may find it helpful to consult the Optimization Toolbox documentation to learn more about the optimization methods.

Upgrading from an Earlier Release

Simulink Design Optimization software combines the functionality of Simulink® Parameter Estimation™ and Simulink® Response Optimization™ products. Prior to R14, Simulink Response Optimization software was called the Nonlinear Control Design Blockset software. Models that are created using Nonlinear Control Design software will not work directly with Simulink Design Optimization software. To convert your Nonlinear Control Design Blockset models to work with Simulink Design Optimization software, use the `ncdupdate` command, as described in Simulink Design Optimization documentation.

Required and Related Products

Simulink Design Optimization software requires MATLAB technical computing software, Simulink software, and Optimization Toolbox software.

The following table summarizes MathWorks™ products that extend and complement the Simulink Design Optimization software. For current information about these and other MathWorks products, visit http://www.mathworks.com/products/product_listing/index.html.

Product	Description
Control System Toolbox	Design and analyze control systems.
Genetic Algorithm and Direct Search Toolbox	Solve optimization problems using genetic algorithms, simulated annealing, and direct search.
Neural Network Toolbox™	Design and simulate neural networks.
Parallel Computing Toolbox™	Perform parallel computations on multicore computers and computer clusters.
Simulink Control Design	Design and analyze control systems in Simulink.
System Identification Toolbox™	Create linear and nonlinear dynamic models from measured input-output data.

Documentation and Demos

In this section...
"Accessing Documentation" on page 1-6
"Accessing Demos" on page 1-6

Accessing Documentation

The Simulink Design Optimization documentation contains the following components:

- Getting Started Guide — Provides information for mapping your problem to the capabilities of the Simulink Design Optimization software. Step-by-step tutorials walk you through the most common tasks for estimating parameters, optimizing parameters, and designing controllers using optimization methods.
- User's Guide — Describes tasks for using the Simulink Design Optimization software.
- Reference — Describes commands and blocks for design optimization.
- Release Notes — Describes important changes in the current product version and compatibility considerations.

If you are new to using this product, the Getting Started Guide helps you begin using this product quickly. You can follow the steps in the tutorials to perform design optimization using the graphical user interface (GUI) or the MATLAB Command Window.

You can also search or browse the documentation for information about specific design optimization tasks.

Accessing Demos

The Simulink Design Optimization software provides demo files that show you how to estimate and optimize parameters of Simulink models, design compensators using optimization methods, and model systems using Adaptive Lookup Tables.

To access demos in the Help browser, type the following command at the MATLAB prompt:

```
demo('Simulink', 'Simulink Design Optimization')
```

2

Estimating Model Parameters

- "Types of Data for Parameter Estimation" on page 2-2
- "Quick Start — Estimating Model Parameters" on page 2-4
- "Parallel Computing for Parameter Estimation" on page 2-12

Types of Data for Parameter Estimation

You can estimate model parameters and initial conditions of single or multiple input and output Simulink models from *transient data*. You measure *transient data* when the system is not in steady-state to capture the system dynamics expected under normal operating conditions. For example, the response of a system to step or impulse inputs is transient data.

Simulink Design Optimization software lets you estimate model parameters from the following types of data:

- *Time-domain* data — Data with one or more input variables *u(t)* and one or more output variables *y(t)*, sampled as a function of time.
- *Time-series* data — Data stored in time-series objects. For more information, see "Time Series Objects" in the MATLAB documentation.

  When you import time-series data for parameter estimation, you must specify the data and time vector as *t.data* and *t.time*, respectively. For more information on how to import data into the GUI, see "Importing Data into the GUI" in the *Simulink Design Optimization User's Guide*.

Simulink Design Optimization does not directly support using complex data for parameter estimation. *Complex-valued* data is data whose value is a complex number. For example, a signal with the value `1+2j` is complex. You can use complex data when you want to estimate parameters of electrical systems. For example, you can use complex data to estimate the magnitude and phase of a system.

If you have complex data, use the MATLAB functions `real`, and `imag` to split the data into two data sets that contain the real and imaginary values. You can then import both data sets into the GUI, as described in "Importing Data into the GUI" in the *Simulink Design Optimization User's Guide*, and estimate the parameters by specifying both data sets simultaneously as estimation data.

Note You must sample the real and imaginary parts of the data as a function of the same time vector.

Simulink Design Optimization software estimates model parameters by comparing the transient data with simulation data generated from the Simulink model. Using optimization techniques, the software estimates the parameters and initial conditions of states to minimize a user-selected cost function. The cost function typically calculates a least-square error between the measured and simulated data. To learn more, see "Estimating Model Parameters" in the *Simulink Design Optimization User's Guide*.

Quick Start — Estimating Model Parameters

In this quick start, you get an overview of the typical tasks for estimating model parameters using the Control and Estimation Tools Manager GUI:

1. Start a parameter estimation task.
2. Import estimation and validation data sets.
3. Select parameters to estimate.
4. Estimate the parameters from the estimation data.
5. Validate the estimated parameters using the validation data set.

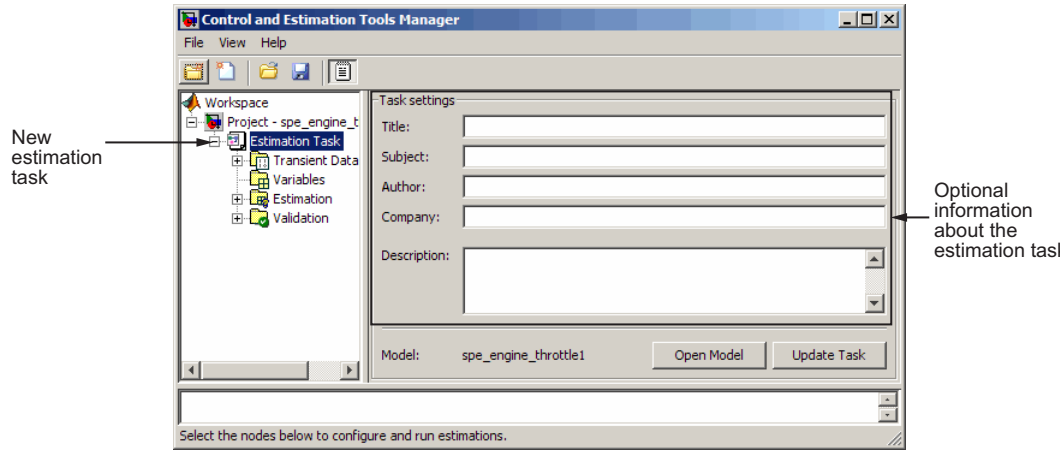
Prerequisites for parameter estimation include:

- Simulink model that contains inport or outport blocks, or signal logging

  For more information, see "Configuring a Model for Importing Data" in the *Simulink Design Optimization User's Guide*.
- Transient data in the MATLAB workspace

To estimate model parameters:

1. Start a parameter estimation task by selecting **Tools > Parameter Estimation** in the Simulink model window.

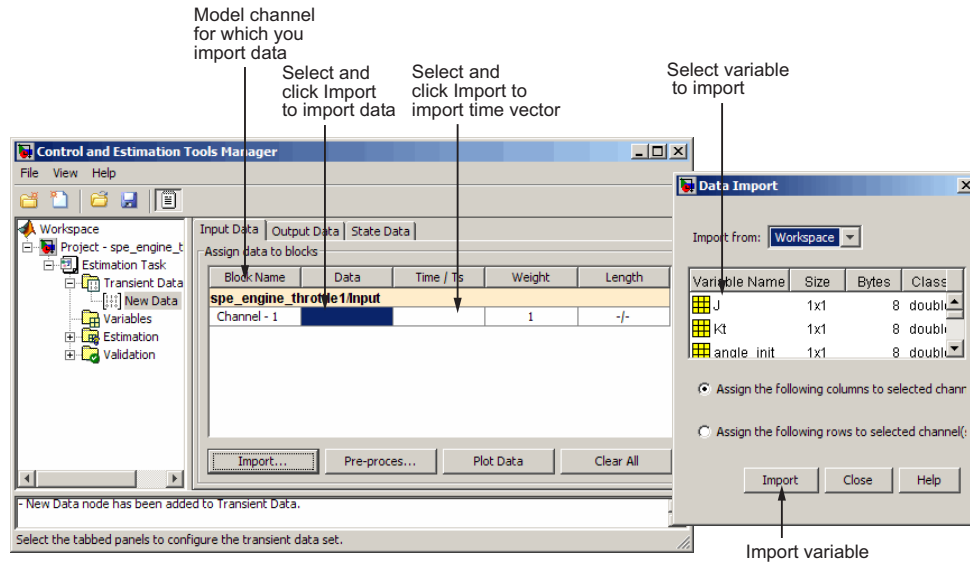

New estimation task
Control and Estimation Tools Manager
File View Help
Workspace
Project - spe_engine_t
Estimation Task
Transient Data
Variables
Estimation
Validation
Task settings
Title:
Subject:
Author:
Company:
Description:
Model:
spe_engine_throttle1
Open Model
Update Task
Select the nodes below to configure and run estimations.
Optional information about the estimation task

2 Import the input and output data for estimating and validating model parameters.

a Select the **Transient Data** node, and click **New**.

b Select the **New Data** node.

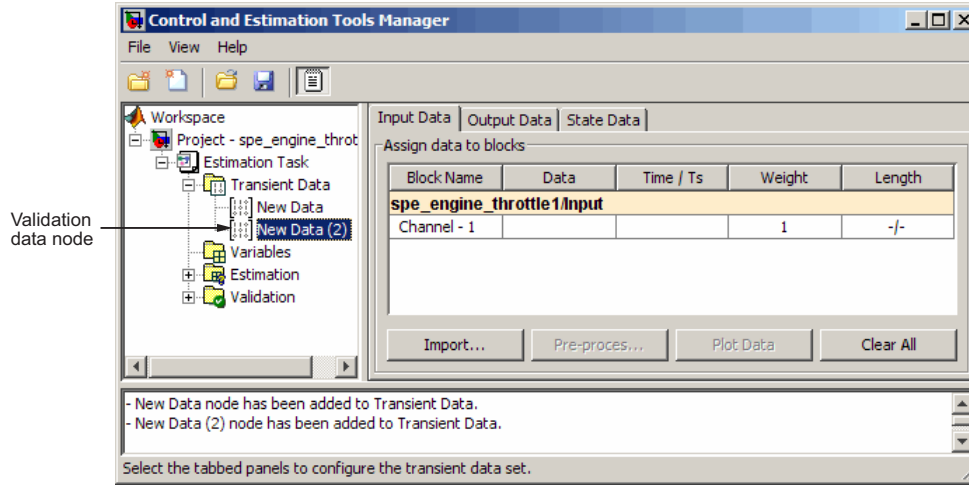
c In the **Input Data** tab, select the **Data** cell corresponding to the model channel, and click **Import**. Select the variable to import in the Data Import dialog box.



d Select the **Time /Ts** cell, and click **Import**. Select the time vector to import in the Data Import dialog box.

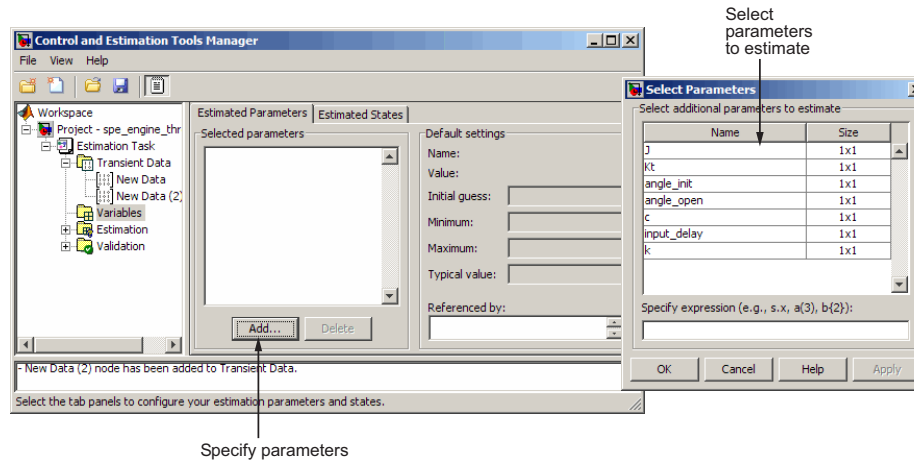
e In the **Output Data** tab, repeat steps c–d to import the output data and time vector.

f Repeat steps a–d to import the validation data set.



For more information, see "Importing Data into the GUI" in the *Simulink Design Optimization User's Guide*.

3 Specify parameters to estimate by selecting the **Variables** node, and clicking **Add**. Select the parameters in the Select Parameters dialog box.



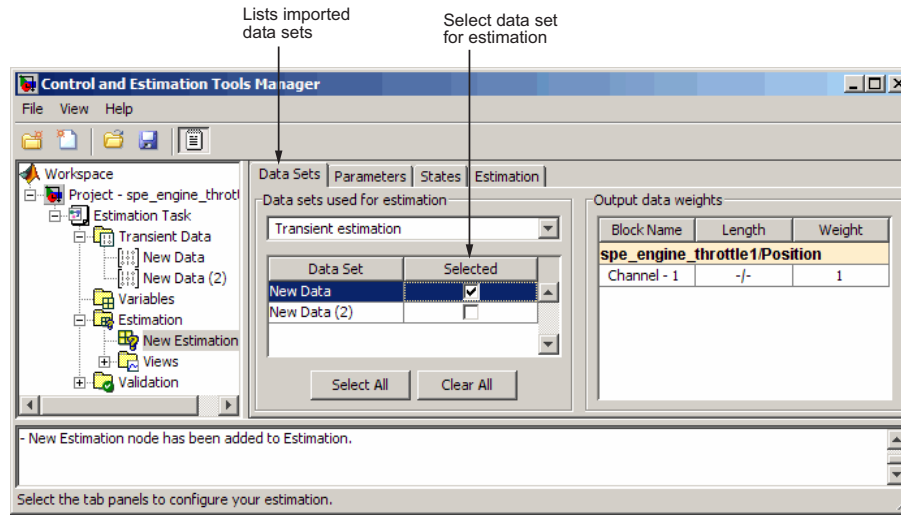
For more information, see “Specifying Parameters to Estimate” in the *Simulink Design Optimization User's Guide*.

4 Estimate the parameters.

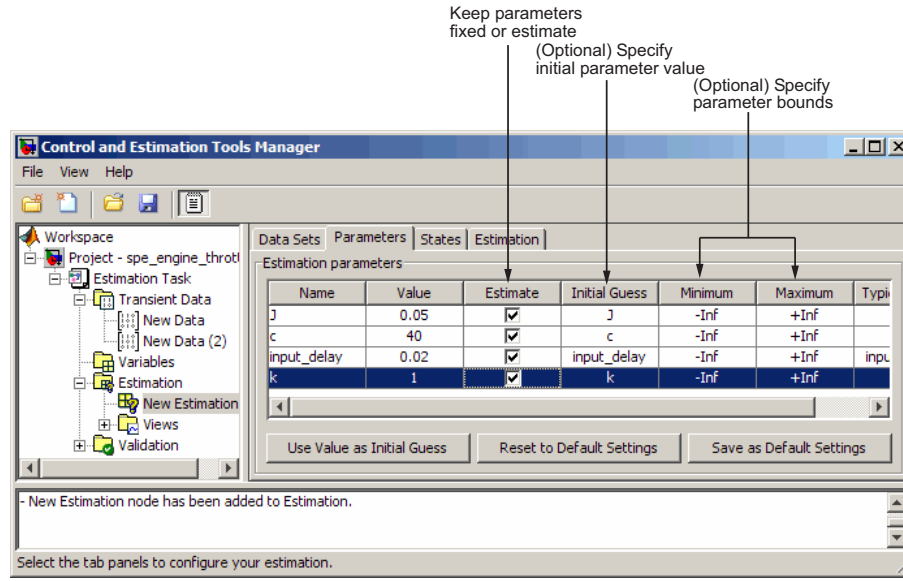
a Select the **Estimation** node, and click **New**.

b Select the **New Estimation** node.

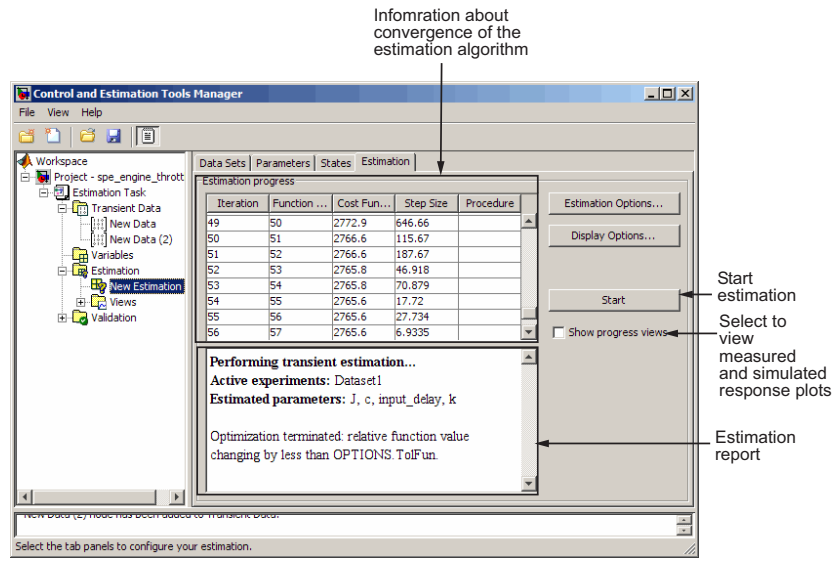
c In the **Data Sets** tab, select the estimation data set.



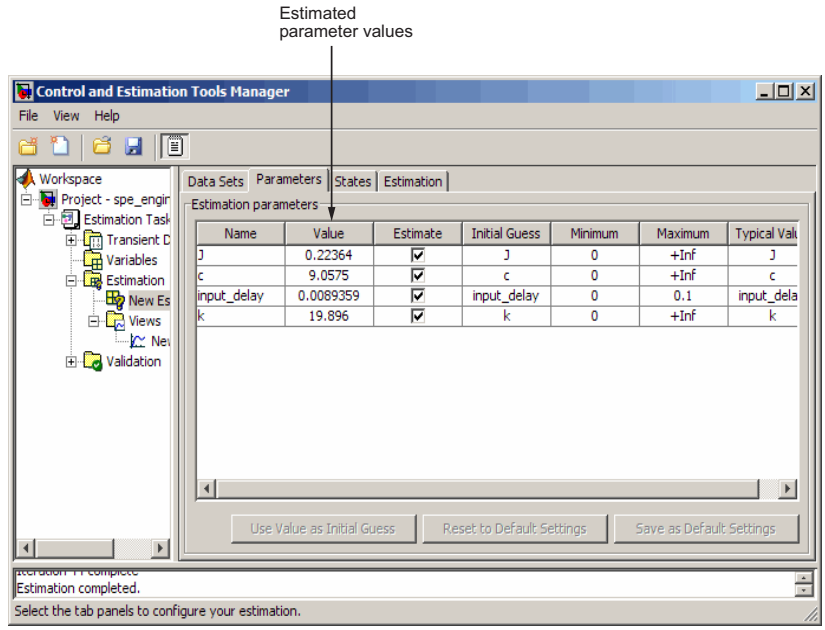
d In the **Parameters** tab, select the parameters to estimate.



e In the **Estimation** tab, begin estimation by clicking **Start**.



f In the **Parameters** tab, examine the estimated parameter values. The Simulink model also gets updated with the estimated parameter values.



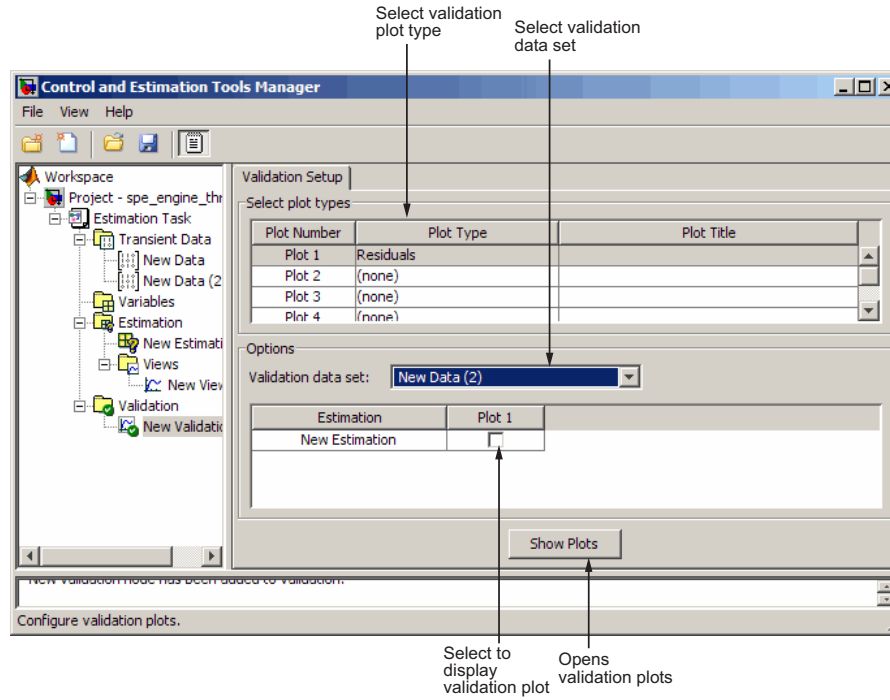
For more information, see "Performing Estimation" in the *Simulink Design Optimization User's Guide*.

5 Validate the estimated parameters.

a Select the **Validation** node, and click **New**.

b Select the **New Validation** node.

c Configure the validation plots and the validation data set.



For more information, see "Performing Validation" in the *Simulink Design Optimization User's Guide*.

See Also: Chapter 6, "Tutorial — Estimating Parameters from Measured Data Using the GUI"

Parallel Computing for Parameter Estimation

When you have the Parallel Computing Toolbox software, you can use parallel computing to speed up parameter estimation. When you use parallel computing, the software distributes the independent simulations on multiple MATLAB sessions. Thus, the simulations run in parallel which reduces the estimation time.

Using parallel computing may reduce the estimation time in the following cases:

- The model contains a large number parameters to estimate, and `Gradient descent` or `Nonlinear least squares` is selected as the estimation method.
- `Pattern search` is selected as the estimation method.
- The model is complex and takes a long time to simulate.

For more information, see "Speeding Up Parameter Estimation Using Parallel Computing" in the *Simulink Design Optimization User's Guide*.

3

Optimizing Model Parameters

- "Types of Time-Domain Design Requirements for Optimizing Parameters" on page 3-2
- "Quick Start — Optimizing Model Parameters" on page 3-3
- "Parallel Computing for Parameter Optimization" on page 3-11

Types of Time-Domain Design Requirements for Optimizing Parameters

You can optimize parameters of Simulink models to meet the following types of time-domain design requirements:

- Step-response characteristics such as overshoot, and rise time.
- Lower and upper bounds on signals
- Reference signal

Simulink Design Optimization software optimizes the model parameters by formulating the time-domain requirements into a constrained optimization problem. It then solves the problem using optimization methods. During the optimization, the software performs the following operations:

- Simulates the Simulink model,
- Compares the simulation data with the constraint objectives and any specified reference signal
- Uses gradient methods to modify selected model parameters to meet the objectives

To learn more, see "Optimizing Model Parameters" in the *Simulink Design Optimization User's Guide*.

Quick Start — Optimizing Model Parameters

In this quick start, you get an overview of the typical tasks for optimizing model parameters to meet time-domain requirements:

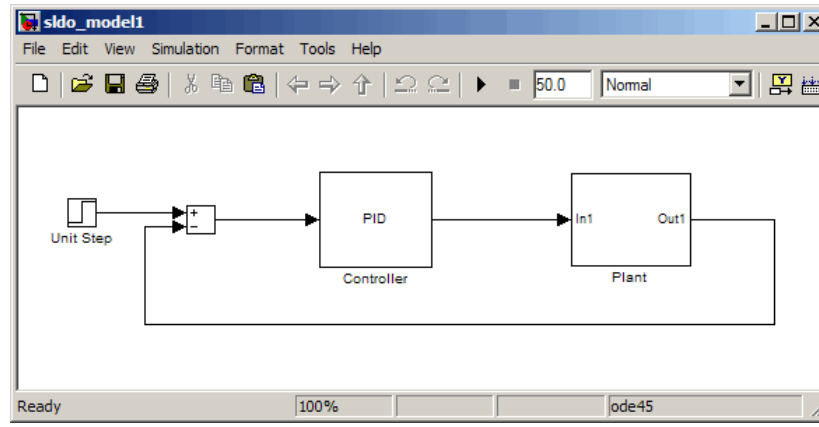
1. Specify an input signal in the Simulink system.
2. Specify the design requirements.
3. Specify parameters to optimize.
4. Optimize the parameters.
5. Evaluate the optimization results.

Prerequisites for optimizing model parameters include:

- Simulink model
- Time-domain design requirements

To optimize model parameters:

1. In the Simulink model, specify an input signal to the system. For example, add a Step block.

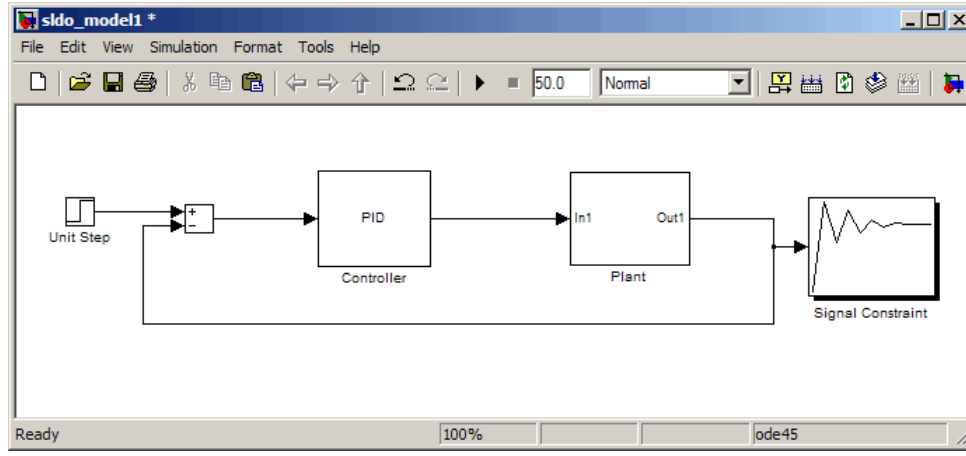


2. Specify the time-domain design requirements:

a In the Simulink Library Browser, select **Simulink Design Optimization**.

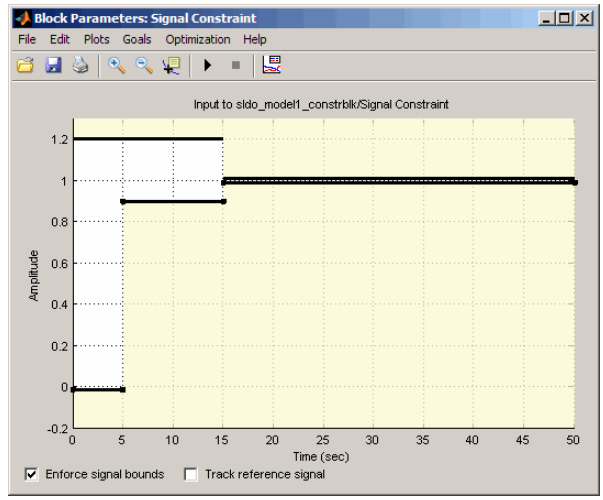
b Drag and drop the Signal Constraint block into the model.

c Connect the Signal Constraint block to the signal that should meet the design requirements.

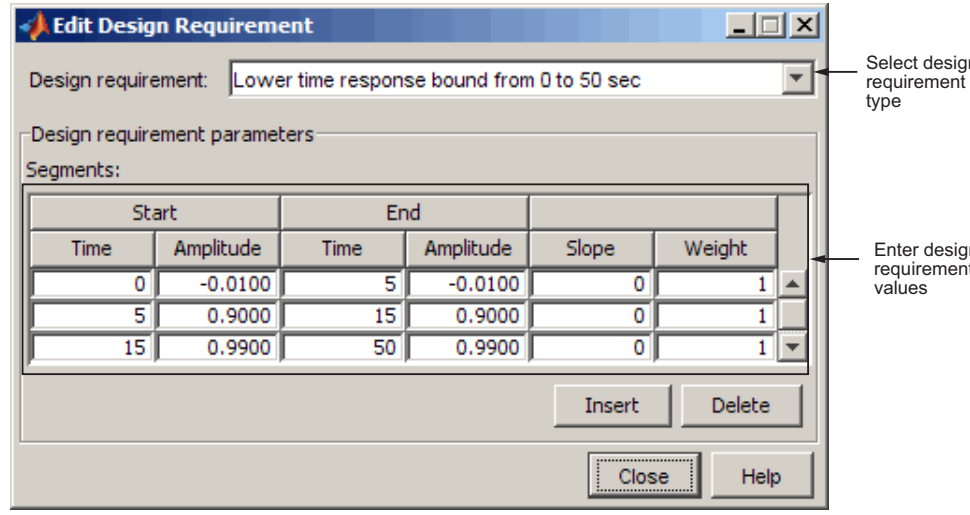


d Double-click the Signal Constraint block.

Design requirements appear as line segments in the Block Parameters: Signal Constraint block window. By default, the design requirements are step-response characteristics.

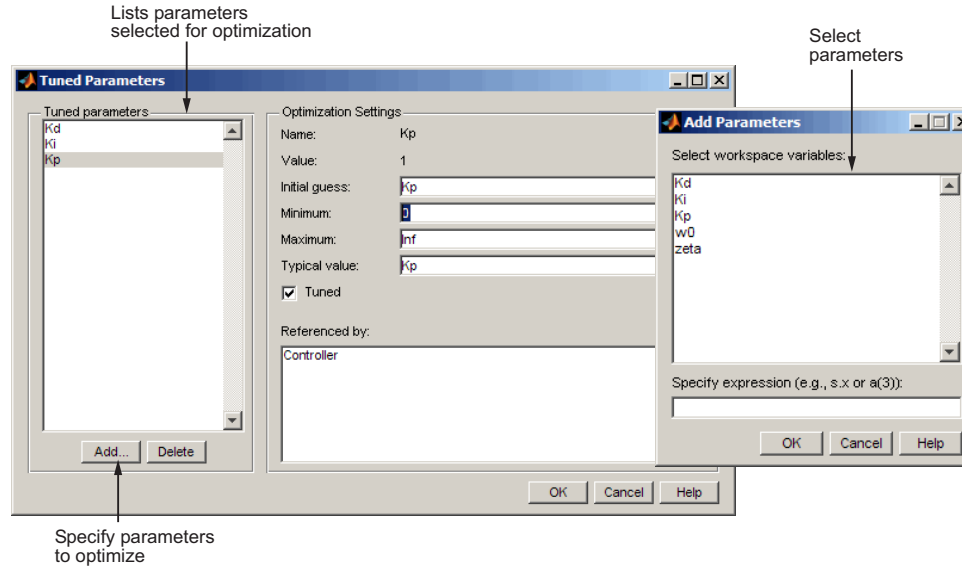


e Double-click the lower yellow region on the plot. Specify the design requirements in the Edit Design Requirement dialog box.



For more information, see "Specifying Design Requirements" in the *Simulink Design Optimization User's Guide*.

3 In the Block Parameters window, select **Optimization > Tuned Parameters**, and click **Add**. Select the parameters to optimize in the Add Parameters dialog box.

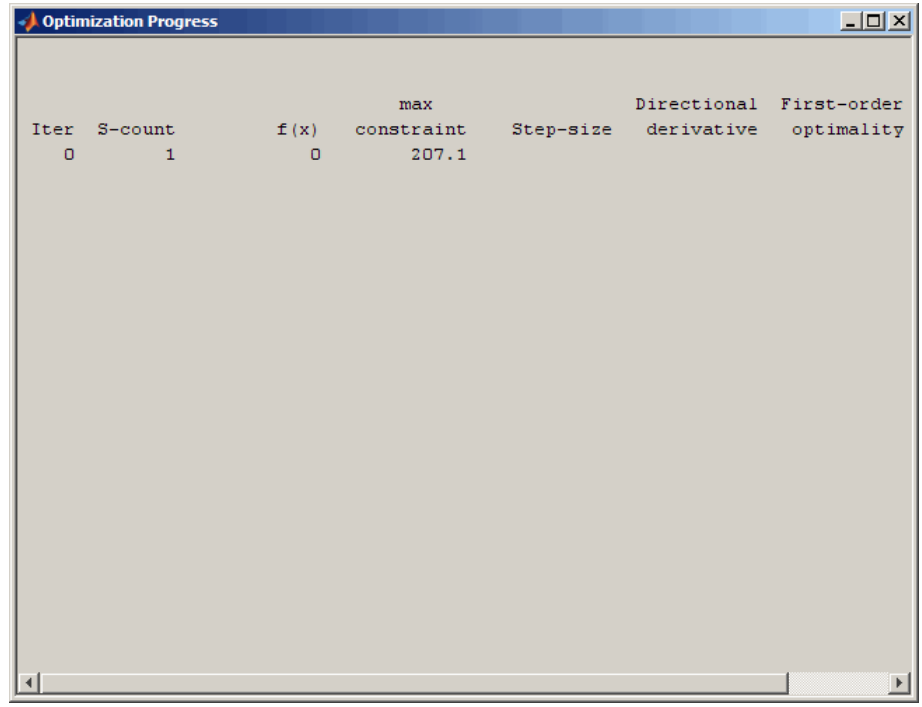


.

For more information, see “Specifying Parameters to Optimize” in the *Simulink Design Optimization User’s Guide*.

4 In the Block Parameters window, start the optimization by selecting **Optimization > Start**.

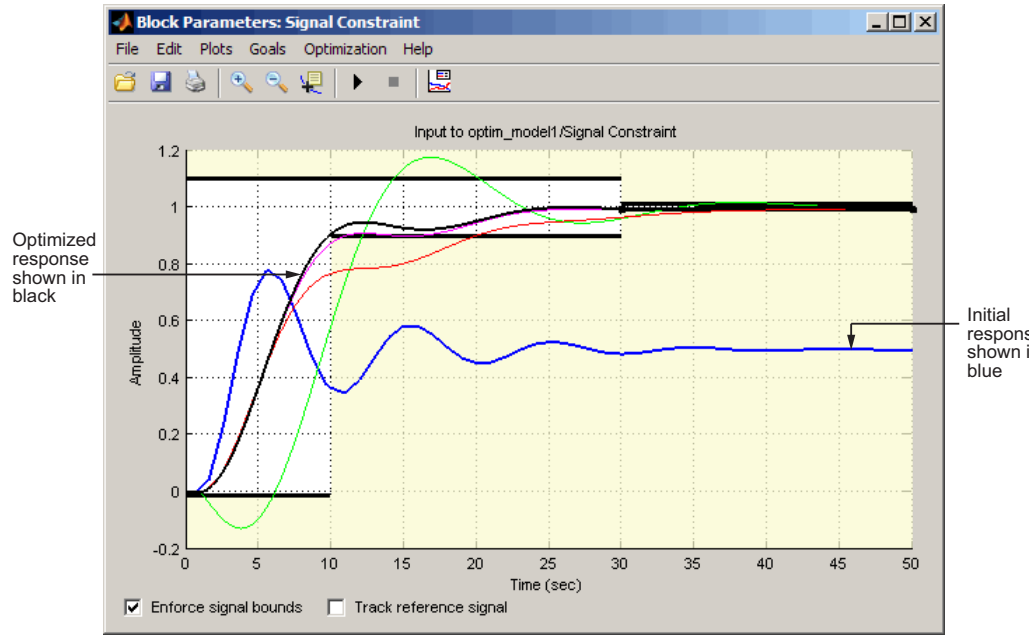
The Optimization Progress window opens where you see the optimization progress.



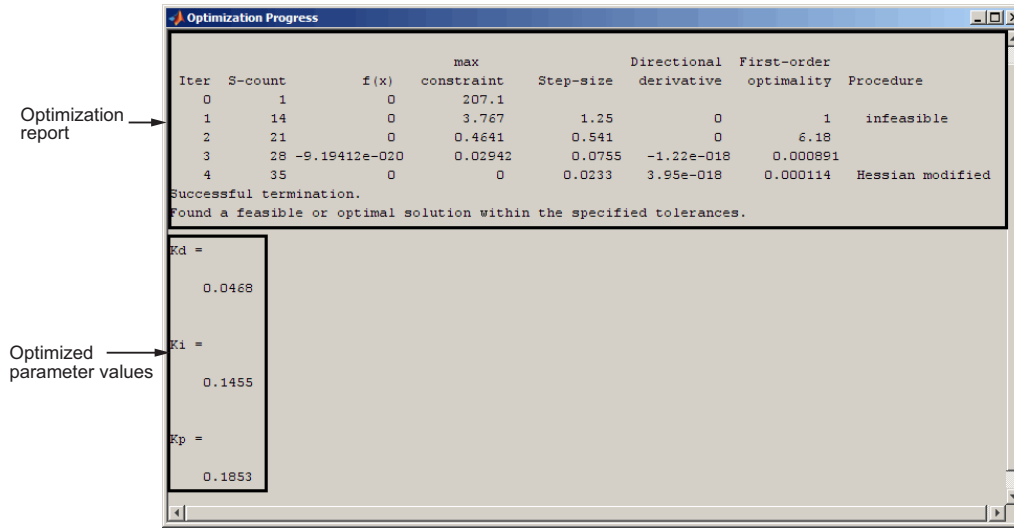
For more information, see “Running the Optimization” in the *Simulink Design Optimization User’s Guide*.

5 Evaluate the optimization results after the optimization completes.

a In the Block Parameters window, compare the response of the system against the design requirements.



b In the Optimization Progress window, view the optimized parameter values.



See Also: Chapter 7, "Tutorial — Optimizing Parameters to Meet Time-Domain Requirements Using the GUI".

Parallel Computing for Parameter Optimization

When you have the Parallel Computing Toolbox software, you can use parallel computing to speed up optimizing model parameters to meet time-domain design requirements. When you use parallel computing, the software distributes the independent simulations on multiple MATLAB sessions. Thus, the simulations run in parallel which reduces the optimization time.

Using parallel computing may reduce the optimization time in the following cases:

- The model contains a large number of parameters to optimize, and the `Gradient descent` method is selected for optimization.
- The`Pattern search` method is selected for optimization.
- The model contains a large number of uncertain parameters and uncertain parameter values.
- The model is complex and takes a long time to simulate.

For more information, see “Speeding Up Response Optimization Using Parallel Computing” in the *Simulink Design Optimization User’s Guide*.

4

Optimization-Based Linear Control Design

- "When to Use Optimization-Based Linear Control Design" on page 4-2
- "Types of Time- and Frequency-Domain Design Requirements" on page 4-3
- "Quick Start — Optimization-Based Linear Control Design" on page 4-4

When to Use Optimization-Based Linear Control Design

When you have Control System Toolbox software installed, you can design and optimize control systems for LTI models by optimizing controller parameters in the SISO Design Tool. To use optimization methods for linear control design, also known as *optimization-based tuning*, you must already have an initial controller. You can then use optimization-based tuning to refine the controller design to meet additional design requirements. For more information on designing controllers, see the Control System Toolbox documentation.

Note Optimization-based tuning only changes the value of the controller parameters and not the controller structure itself.

Optimization-based tuning provides flexibility in terms of specifying additional design requirements for the controller. When you have a large number of design requirements, you can first design an initial controller by selecting a subset of requirements and subsequently select additional requirements to refine the design.

Optimization-based tuning also provides flexibility in terms of selecting a subset of controller parameters to optimize, and specifying bounds on the controller parameters.

To design linear controllers for Simulink models using optimization-based tuning, you must first linearize the model using the Simulink Control Design software. For more information on linearizing Simulink models, see the Simulink Control Design documentation.

Types of Time- and Frequency-Domain Design Requirements

When you design linear controllers for LTI or Simulink models using the Simulink Design Optimization software, you can specify both time- and frequency-domain requirements on the system response. You can specify design requirements on the following plots:

- Root Locus plot
- Open-Loop and Prefilter Bode plots
- Open-Loop Nichols plot
- Step/Impulse Response plots

For more information, see "Supported Time- and Frequency-Domain Requirements" in the *Simulink Design Optimization User's Guide*.

Simulink Design Optimization software uses the frequency-domain requirements to compute the frequency response of the system. It then uses optimization methods to reduce the distance between the current response and the requirements by modifying the controller parameters. The software does not change the controller structure when optimizing the controller parameters. To learn more, see "Optimization-Based Linear Compensator Design" in the *Simulink Design Optimization User's Guide*.

Quick Start — Optimization-Based Linear Control Design

In this quick start, you get an overview of the typical tasks for optimization-based linear control design using the SISO Design Tool:

1. Open a SISO Design Tool session.
2. Configure a project for optimization-based control design.
3. Specify the controller parameters to design.
4. Specify the design requirements.
5. Design the controller.
6. Evaluate the controller design.

Note The same workflow applies to optimization-based control design for LTI models created at the command line using Control System Toolbox software. To learn how to create LTI models, see "Linear (LTI) Models" in the Control System Toolbox documentation.

Prerequisites for optimization-based linear control design include:

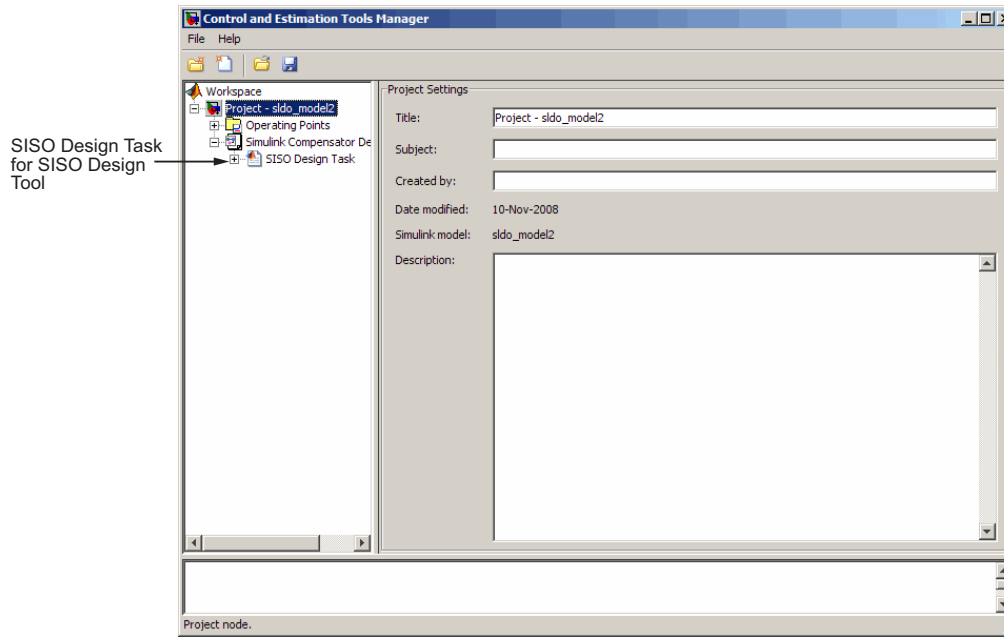
- Simulink Compensator Design Task that contains a linearized version of the Simulink model and, optionally, any response plots you configure.

  For more information on how to linearize a Simulink model for control design, see "Designing Compensators" in the Simulink Control Design documentation.
- Time- and frequency-domain design requirements

To design a controller using optimization methods:

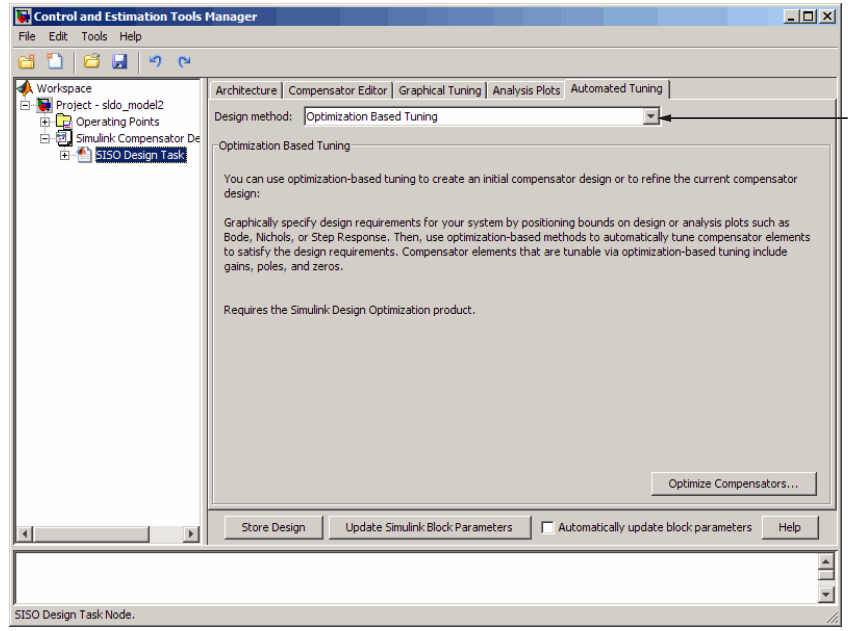
1 Open a SISO Design Tool session by typing the following command at the MATLAB prompt:

```
sisotool('projectname.mat')
```

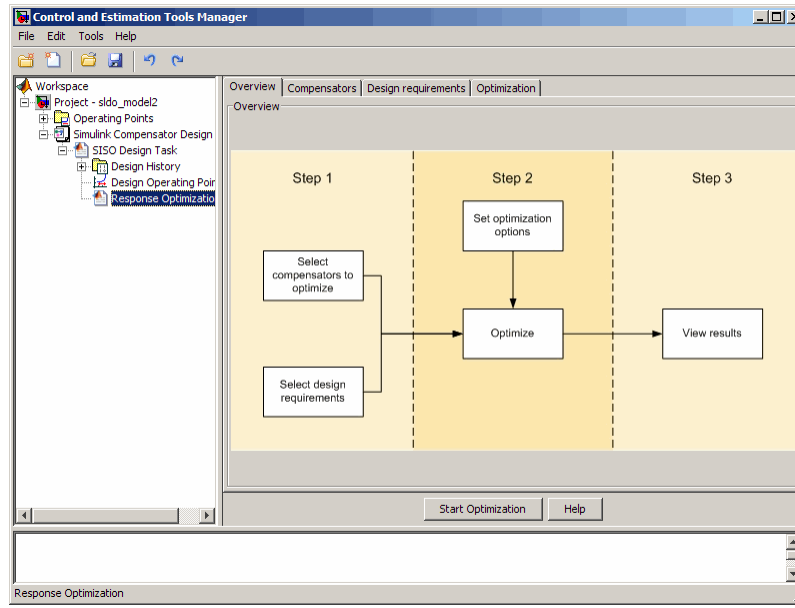


The command also opens a SISO Design for SISO Design Task window by default and any response plots you configured when you linearized the Simulink model using Simulink Control Design software.

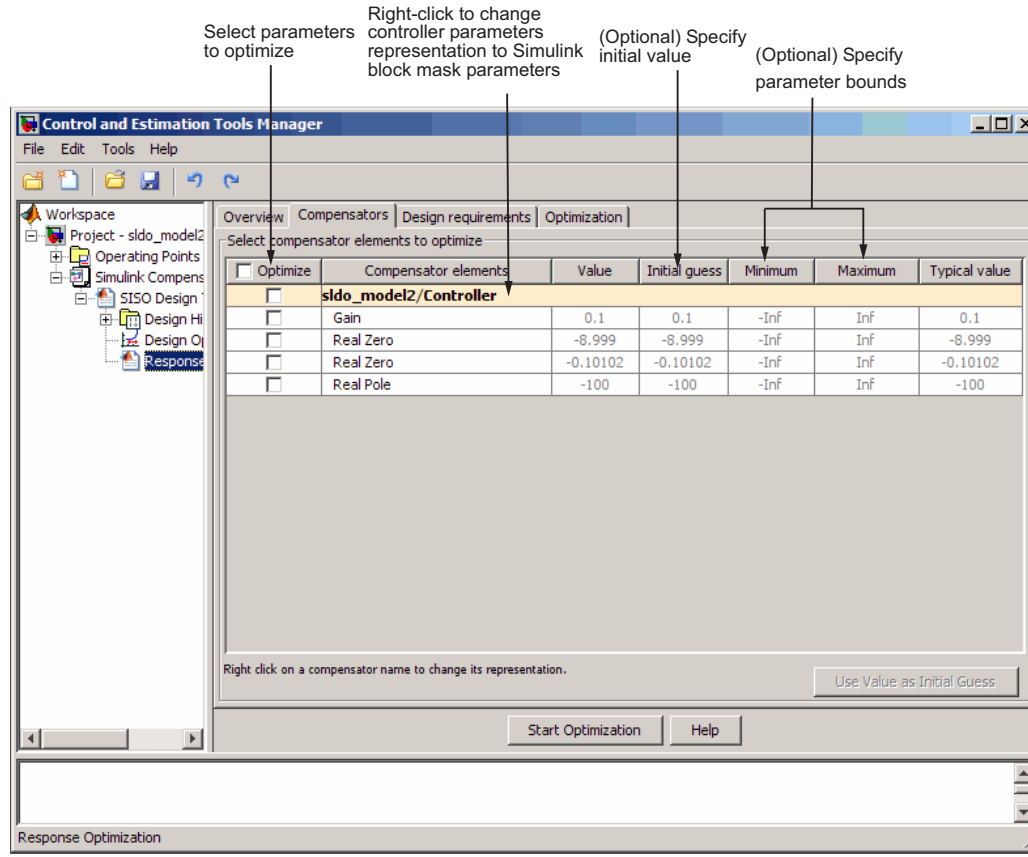
2 Configure a project for optimization-based control design by clicking **Optimize Compensators** in the Automated Tuning tab of the **SISO Design Task**.



This action creates a new **Response Optimization** node in the Control and Estimation Tools Manager.

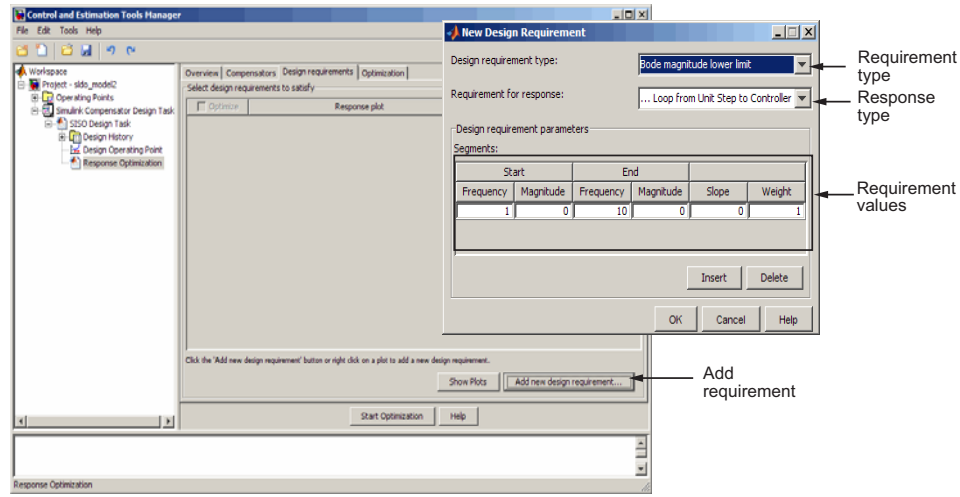


3 Specify the controller parameters to design in the **Compensators** tab.

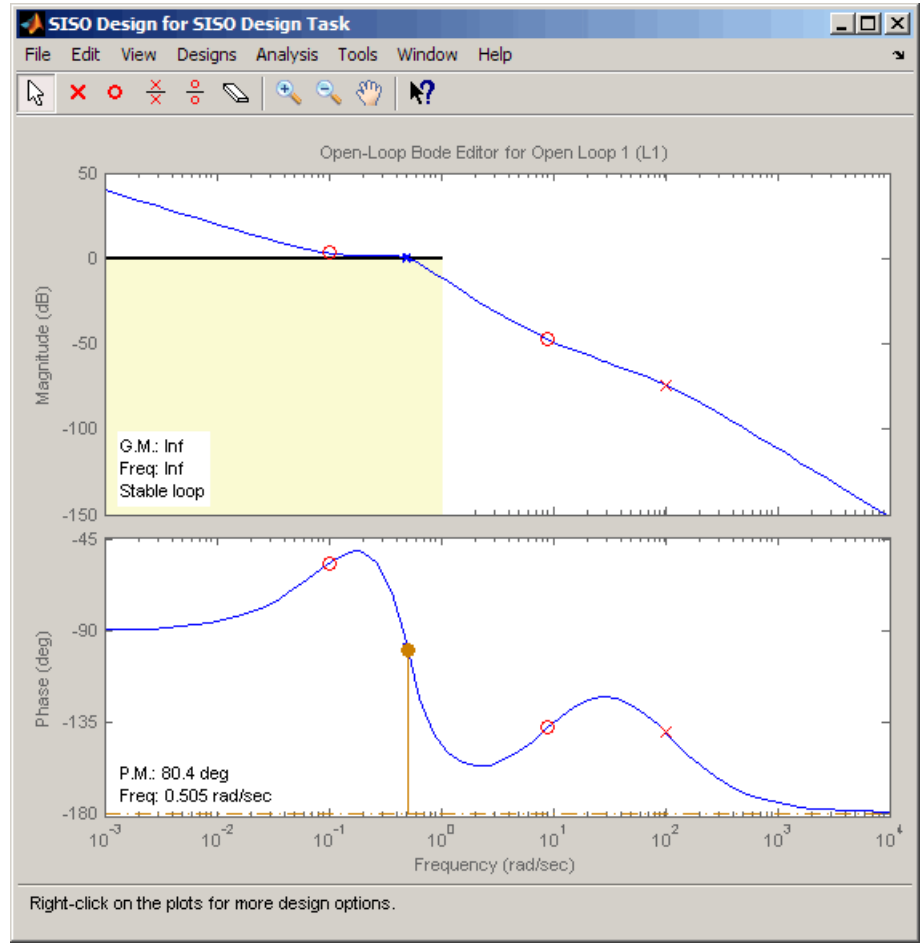


4 Specify the design requirements.

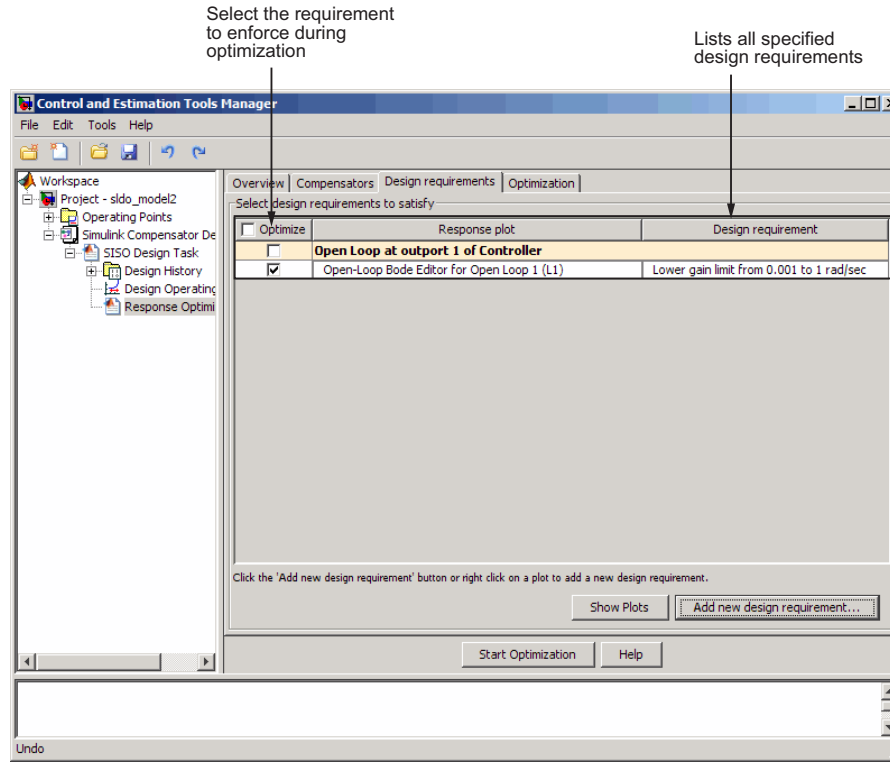
a In the **Design requirements** tab, click **Add new design requirement**. Specify the design requirements, for example Bode magnitude lower limit, in the New Design Requirement dialog box.



In the SISO Design window, the yellow region with the black line segment represents the design requirement on the response plot.

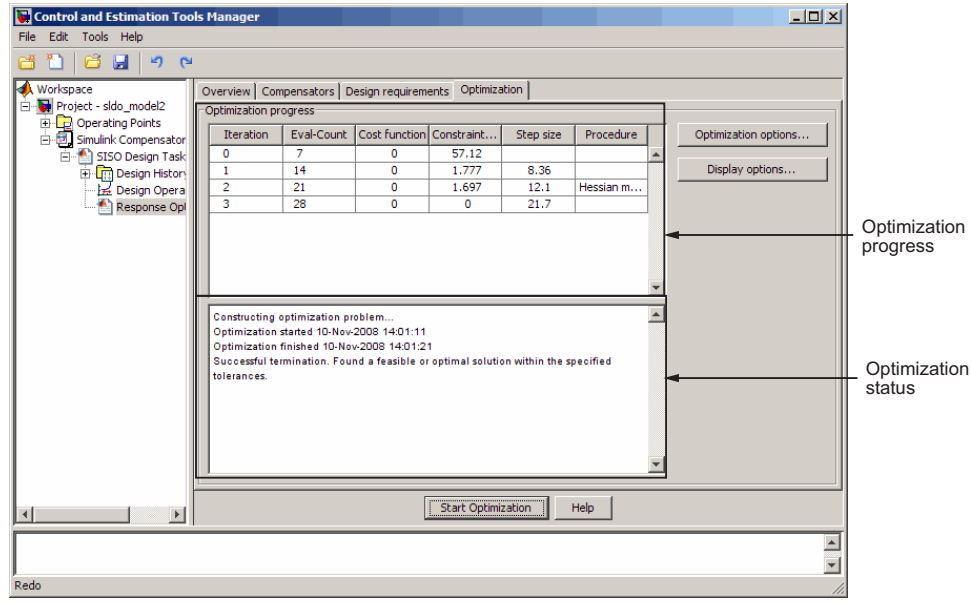


The **Design Requirements** tab also lists the design requirement.



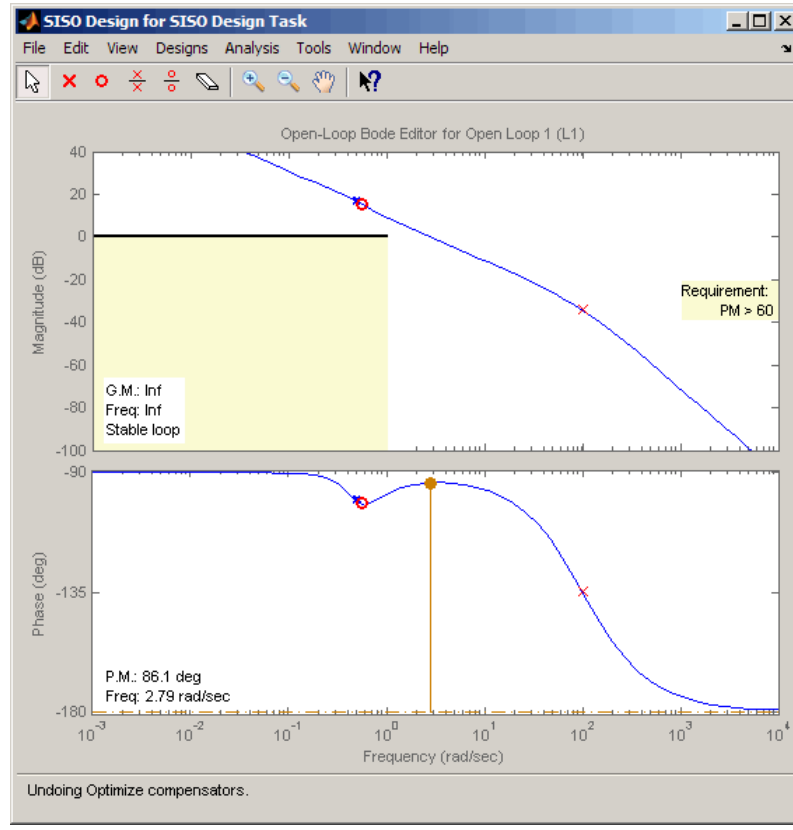
b Repeat step a to specify additional time- and frequency-domain requirements.

5 Start the optimization to design the controller by clicking **Start Optimization** in the **Optimization** tab.

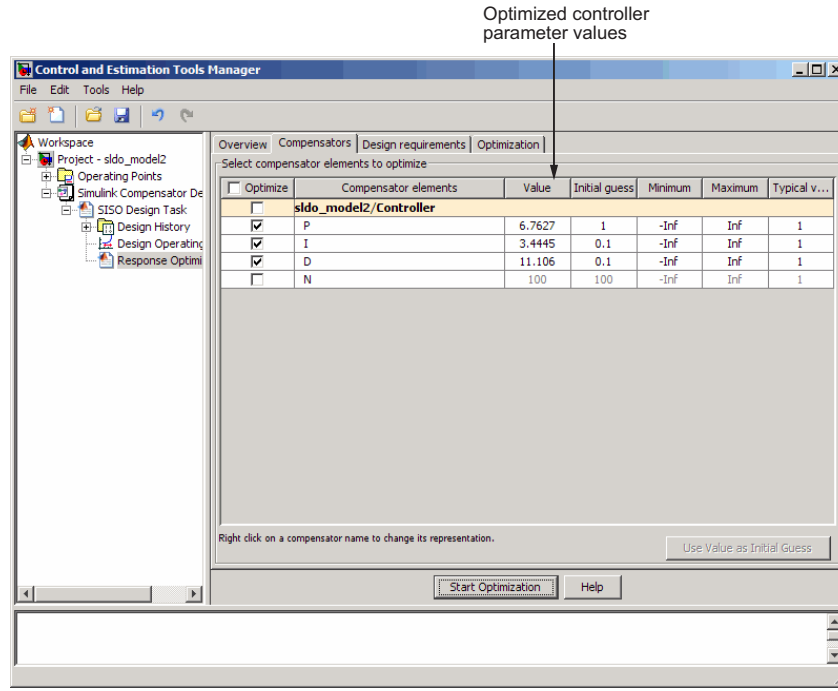


6 Evaluate the controller design.

a Examine the system's response in the response plot, for example the Bode plot, to see if it meets the requirements. The system's response must lie in the white region in order to meet the design requirement.



b Examine the controller parameter values in the **Compensator** tab.



7 Write the controller parameter values into the Simulink model. To do so, click **Update Simulink Block Parameters** in the **SISO Design Task** node.

See Also: Chapter 9, “Tutorial — Designing a PID Controller Using Optimization-Based Tuning”.

5

Tutorial — Preparing Data for Parameter Estimation Using the GUI

- "About This Tutorial" on page 5-2
- "Configuring a Project for Parameter Estimation" on page 5-4
- "Importing Data into the GUI" on page 5-6
- "Analyzing Data" on page 5-14
- "Selecting Data for Estimation" on page 5-16
- "Removing Outliers" on page 5-25
- "Filtering Data" on page 5-29
- "Interpolating Missing Data" on page 5-34
- "Saving the Project" on page 5-37

About This Tutorial

In this section...
"Objectives" on page 5-2
"About the Sample Data" on page 5-2

Objectives

In this tutorial, you learn how to import, analyze, and prepare measured input and output (I/O) data for estimating parameters of a Simulink model.

Note Simulink Design Optimization software estimates parameters from real, time-domain data only.

You learn to perform the following tasks using the GUI:

- Import data from the MATLAB workspace.
- Analyze data quality using a time plot.
- Select a subset of data for estimation.
- Remove outliers.
- Filter high-frequency noise.
- Compute missing data using interpolation.

About the Sample Data

In this tutorial, you use `spe_engine_throttle1.mat`, which contains I/O data measured from an engine throttle system. The MAT-file includes the following variables:

- `input1` — Input data samples
- `position1` — Output data samples
- `time1` — Time vector

Note The number of input and output data samples must be equal to the length of the corresponding time vector.

The engine throttle system controls the flow of air and fuel mixture to the engine cylinders. The throttle body contains a butterfly valve which opens when a driver presses the accelerator pedal. Opening this valve increases the amount of fuel mixture entering the cylinders, which increases the engine speed. A DC motor controls the opening angle of the butterfly valve in the throttle system. The input to the throttle system is the motor current (in amperes), and the output is the angular position of the butterfly valve (in degrees).

`spe_engine_throttle1.mdl` contains the Simulink model of the engine throttle system. For more information on building models, see "Creating a Simulink Model" in the Simulink documentation.

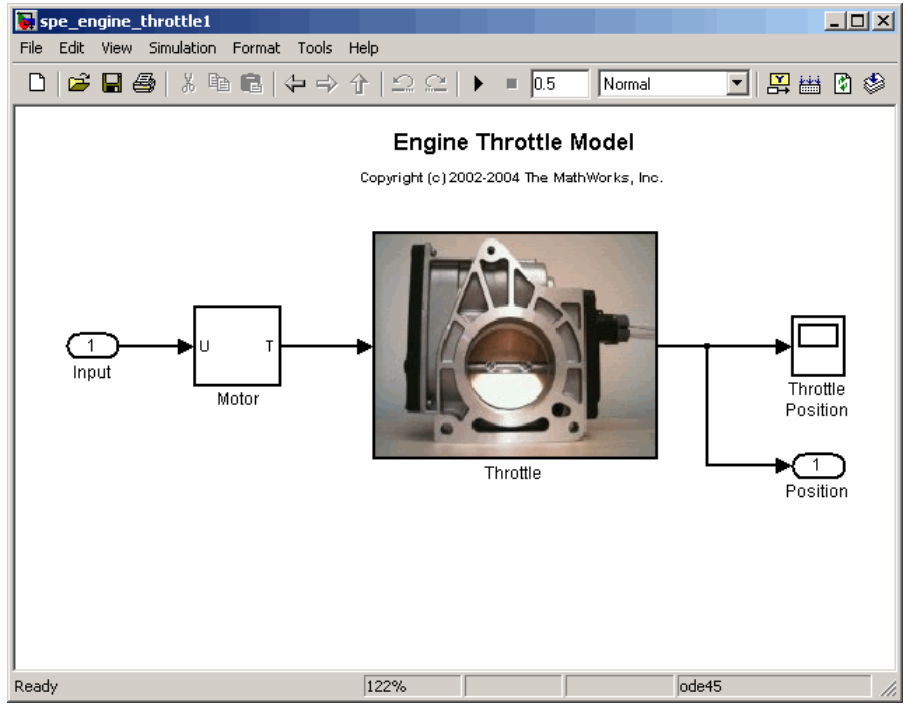
Configuring a Project for Parameter Estimation

To perform parameter estimation, you must first configure a Control and Estimation Tools Manager project.

1 Open the engine throttle system model by typing the following at the MATLAB prompt:

```
spe_engine_throttle1
```

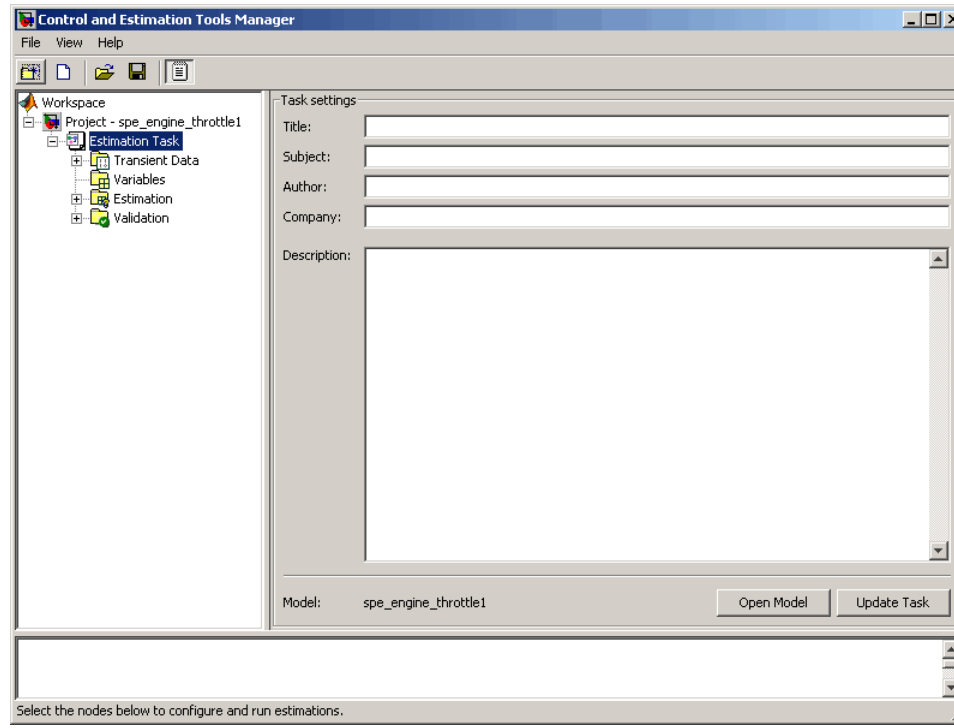
This command opens the Simulink model, and loads the data into the MATLAB workspace.



Simulink® Model of Engine Throttle System

2 In the Simulink model window, select **Tools > Parameter Estimation**.

This action opens a new project named **Project - spe_engine_throttle1** in the Control and Estimation Tools Manager GUI. This project contains the **Estimation Task**, as shown in the next figure.



Note The Simulink model must remain open to perform parameter estimation tasks.

Importing Data into the GUI

In this section...

"Importing Input Data and Time Vector" on page 5-6

"Importing Output Data and Time Vector" on page 5-11

Importing Input Data and Time Vector

In this portion of the tutorial, you import measured I/O data into the Control and Estimation Tools Manager GUI. Importing data assigns the data to the corresponding model input and output signals.

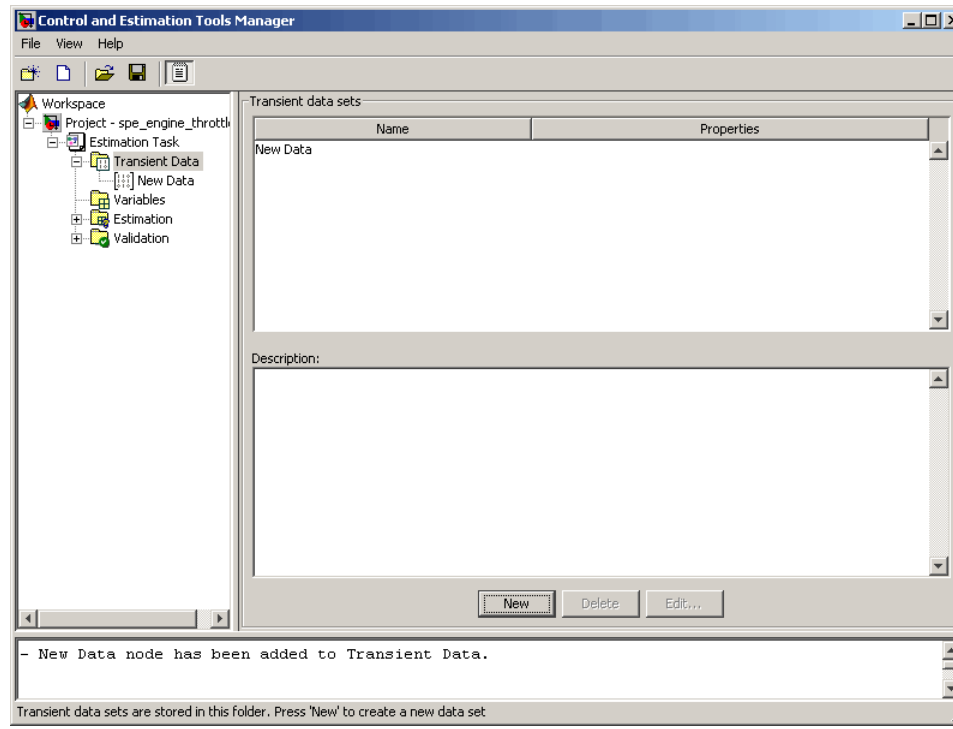
The model input and output signals are designated with the Inport `Input` and Outport `Position` blocks respectively, as shown in the figure Simulink® Model of Engine Throttle System on page 5-4. These blocks let you import I/O data into the GUI. To learn more about the blocks, see the Inport and Outport block reference pages in the Simulink documentation.

You must have already configured the parameter estimation project, as described in "Configuring a Project for Parameter Estimation" on page 5-4.

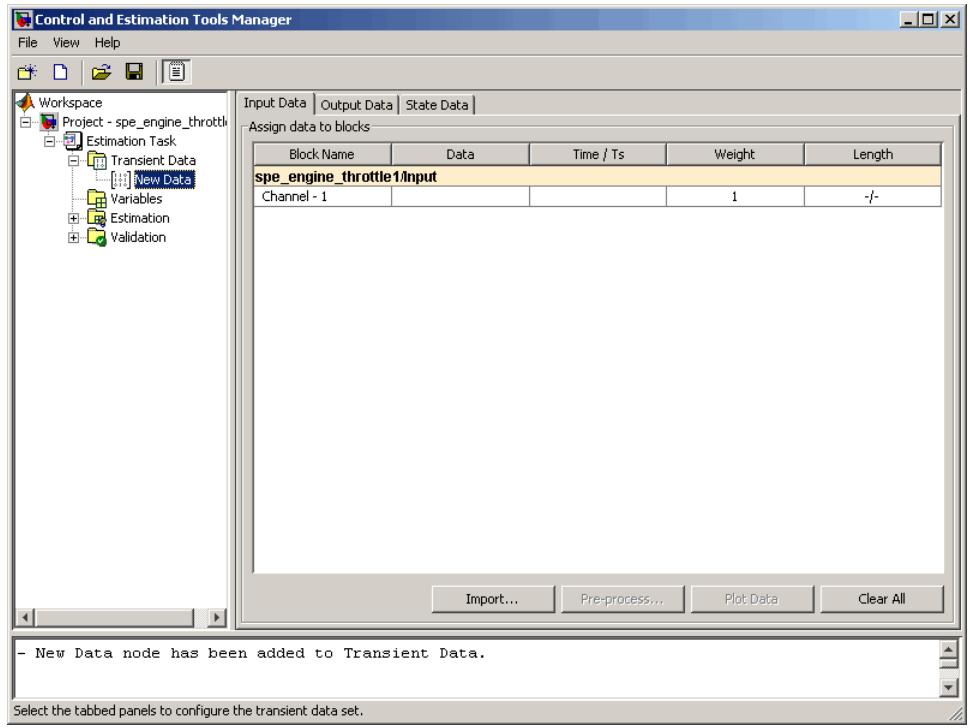
To import input data and time vector into the Control and Estimation Tools Manager GUI:

1 In the Control and Estimation Tools Manger GUI, select **Transient Data** under the **Estimation Task** node, and click **New**.

This action creates a **New Data** node under the **Transient Data** node.



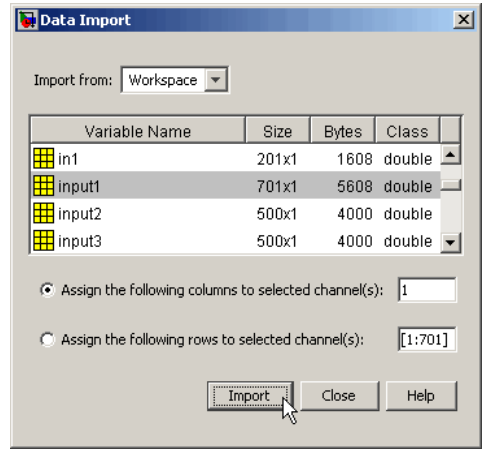
2 Select the **New Data** node.



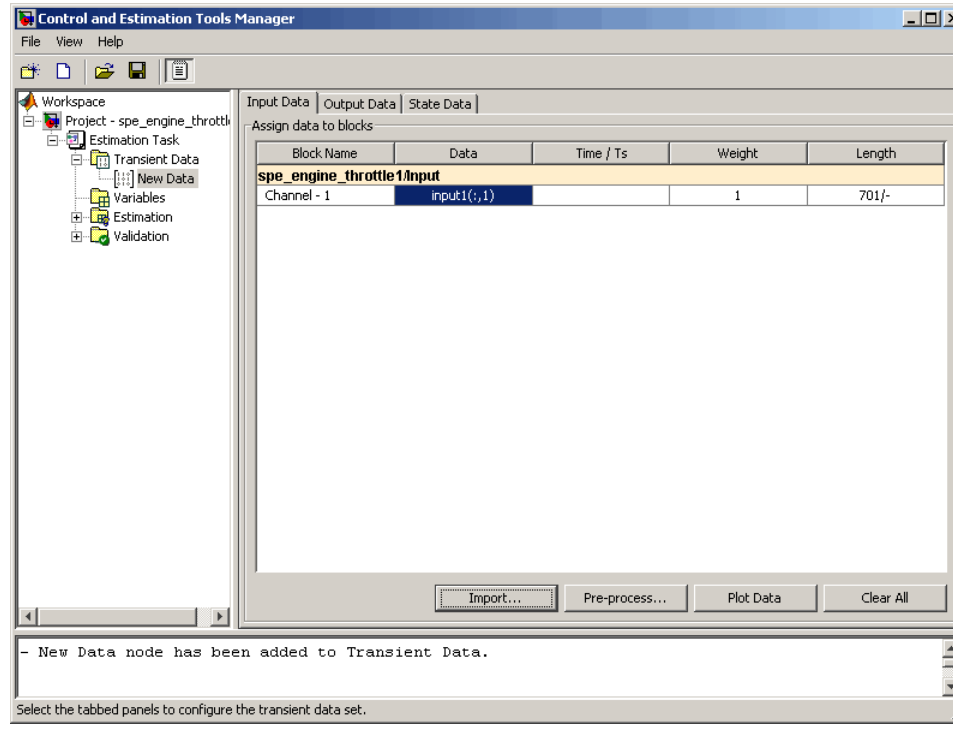
3 In the **Input Data** tab, select the **Data** cell for **Channel - 1**, and click **Import**.

This action opens the Data Import dialog box.

4 In the Data Import dialog box, select `input1`, and click **Import**.



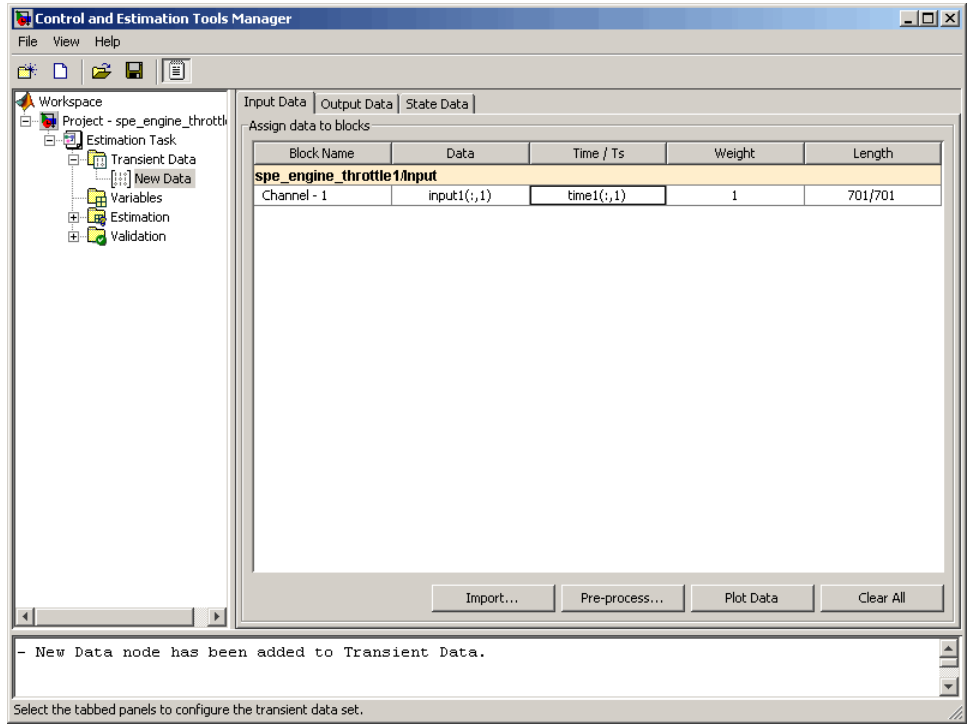
This action assigns the input data `input1` to the model input signal **spe_engine_throttle1/Input**.



5 Select the **Time / Ts** cell for **Channel - 1**.

6 In the Data Import dialog box, select `time1`, and click **Import**.

This action assigns the time vector to the model input signal **spe_engine_throttle1/Input**.

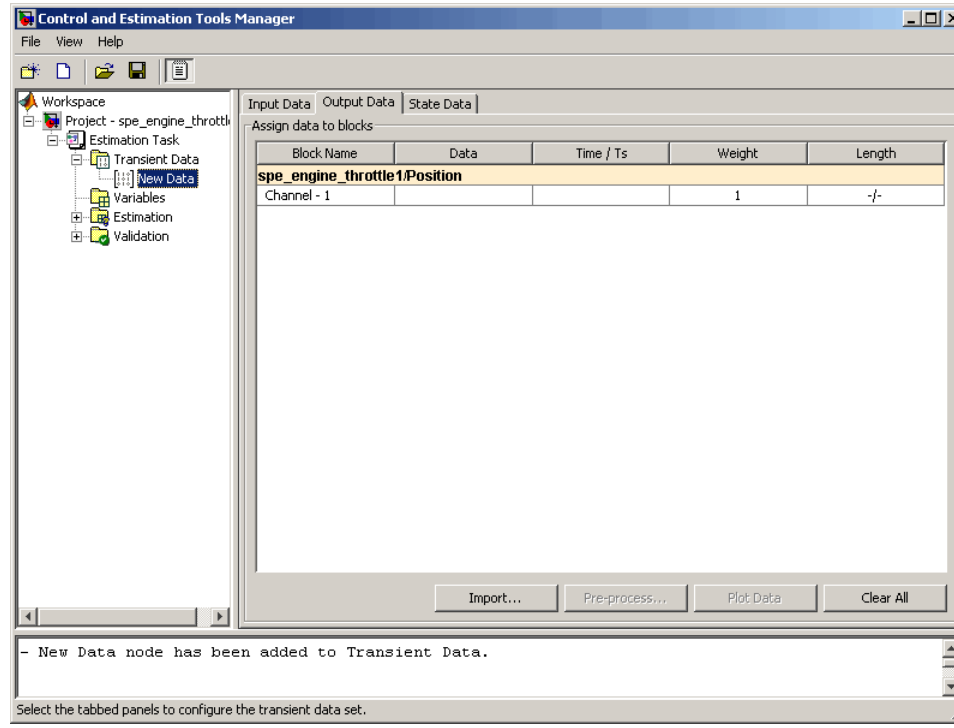


7 In the Data Import dialog box, click **Close**.

Importing Output Data and Time Vector

To import output data and time vector into the Control and Estimation Tools Manager GUI:

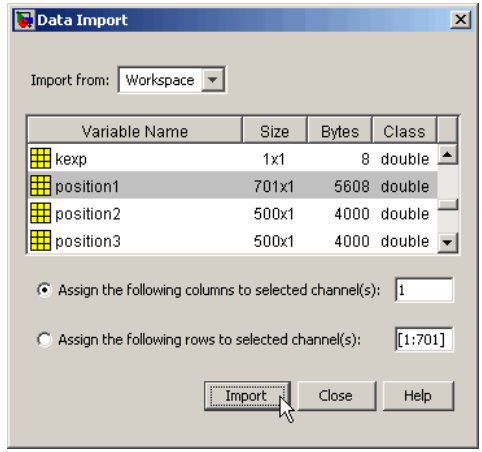
1 In the Control and Estimation Tools Manger GUI, select the **Output Data** tab of the **New Data** node.



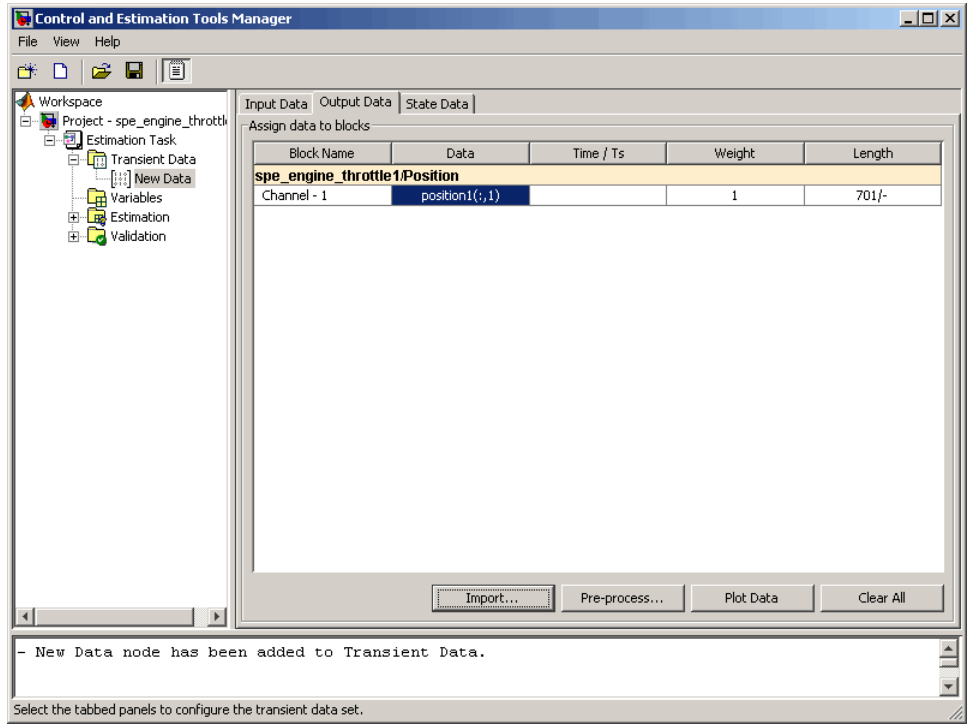
2 Select the **Data** cell for **Channel - 1**, and click **Import**.

This action opens the Data Import dialog box.

3 In the Data Import dialog box, select `position1`, and click **Import**.



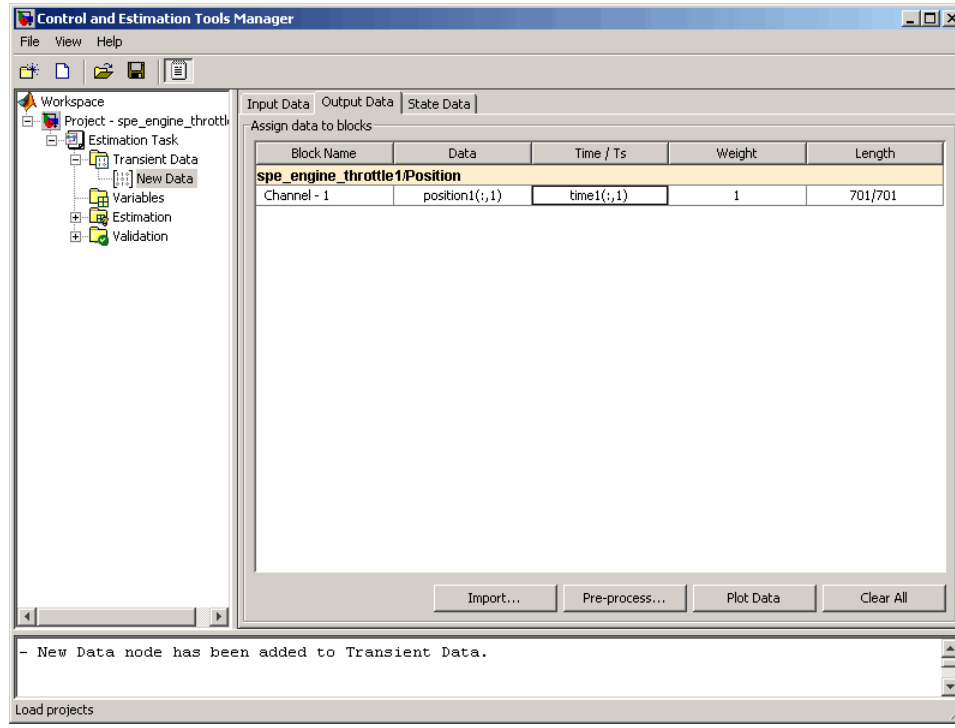
This action assigns the output data `position1` to the model output signal **spe_engine_throttle1/Position**.



4 Select the **Time / Ts** cell for **Channel - 1**.

5 In the Data Import dialog box, select `time1`, and click **Import**.

This action assigns the time vector to the model output signal **spe_engine_throttle1/Position**.



6 In Data Import dialog box, click **Close**.

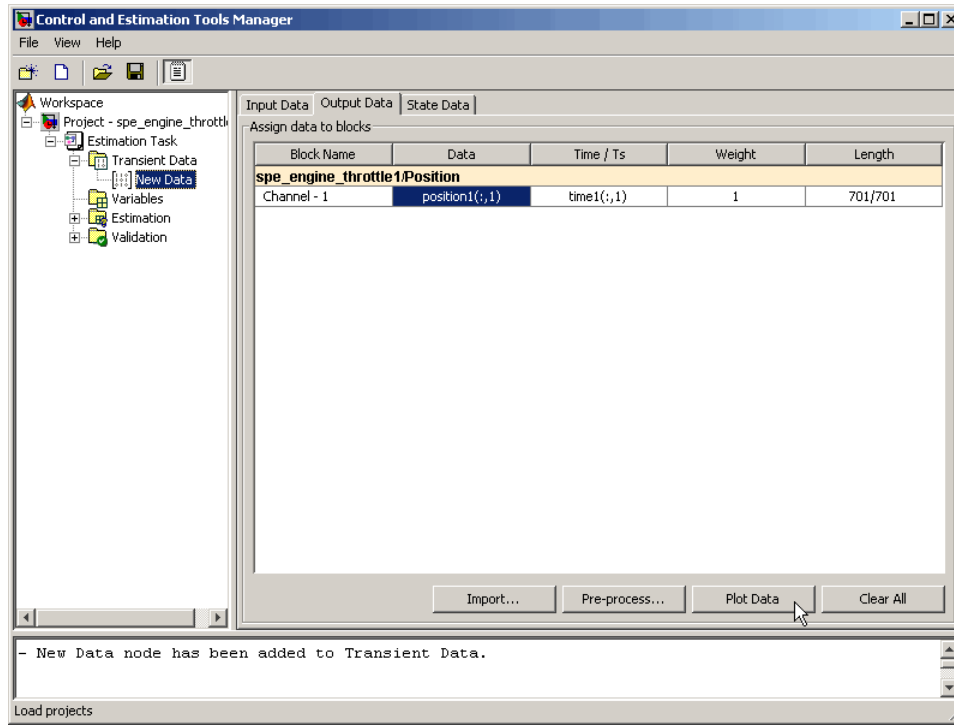
You have now imported the I/O data into the Control and Estimation Tools Manager GUI, and assigned the data to the corresponding model signals.

Analyzing Data

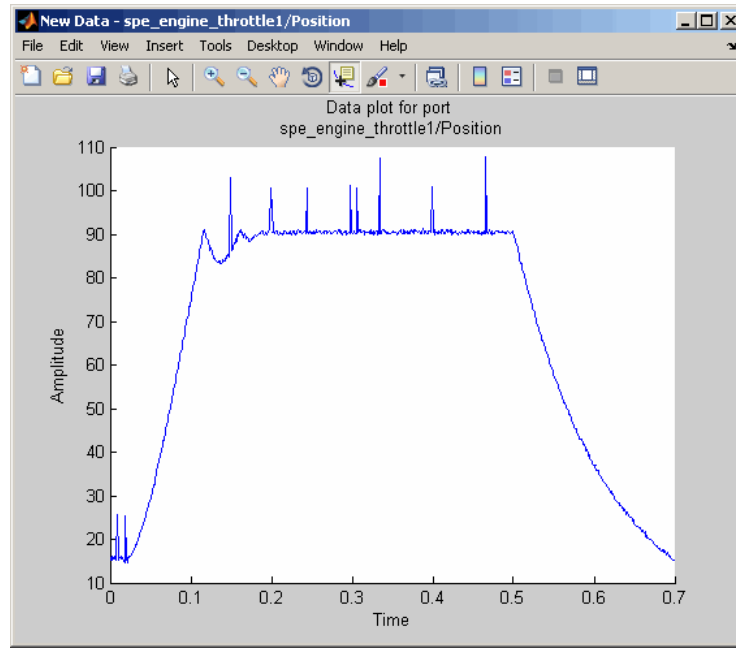
In this portion of the tutorial, you analyze the output data quality by viewing the data characteristics on a time plot. Based on the analysis, you decide whether to preprocess the data before estimating parameters. For example, if the data contains noise, you might want to filter the noise from the system dynamics before estimating parameters.

You must have already imported the data into the Control and Estimation Tools Manager GUI, as described in "Importing Data into the GUI" on page 5-6. If you have not imported the data, click here.

To plot the output data on a time plot, select the `position1(:,1)` cell in the **Output Data** tab, and click **Plot Data**.



This action plots the measured output data `position1(:,1)`, as shown in the next figure.



The time plot shows the output data in response to a step input, as described in "About the Sample Data" on page 5-2. The plot shows a rapid decrease in the response after t = 0.5 s because the system is shut down. To focus parameter estimation on the time period when the system is active, you select the data samples between t = 0 s and t = 0.5 s, as described in "Selecting Data for Estimation" on page 5-16 section of this tutorial.

The spikes in the data indicate *outliers*, defined as data values that deviate from the mean by more than three standard deviations. They may be caused by measurement errors or sensor problems. The response also contains noise. Before estimating model parameters from this data, you remove the outliers and filter the noise, as described in "Removing Outliers" on page 5-25, and "Filtering Data" on page 5-29 sections of this tutorial.

Tip You can also plot the input data on a time plot by selecting the `input1(:,1)` cell in the **Input Data** tab, and clicking **Plot Data**.

Selecting Data for Estimation

In this section...
"Selecting Output Data" on page 5-16
"Selecting Input Data" on page 5-22

Selecting Output Data

In this portion of the tutorial, you select a subset of I/O data for estimation. As described in "Analyzing Data" on page 5-14, the system is shut down at t = 0.5 s. To focus the estimation on the time period before t = 0.5 s, you exclude the data samples beyond t = 0.5 s. This operation selects the data between t = 0 s and t = 0.5 s for estimation.

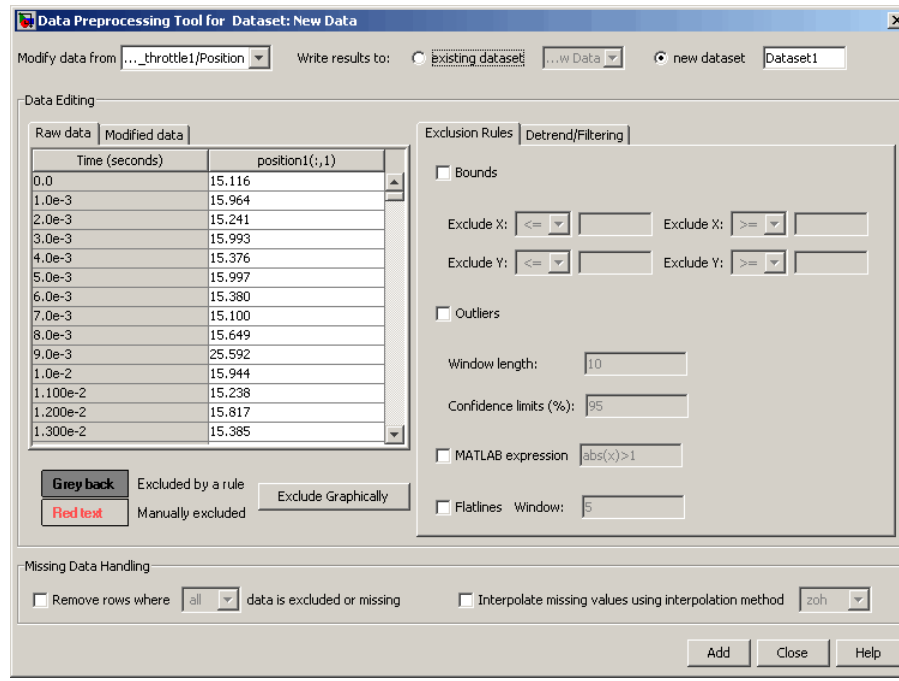
You must have already imported the data into the Control and Estimation Tools Manager GUI, as described in "Importing Data into the GUI" on page 5-6. If you have not imported the data, click here.

To select the portion of data between t = 0 s and t = 0.5 s:

1. In the Control and Estimation Tools Manager window, select the **New Data** node under the **Transient Data** node.

2 Select the `position1(:,1)` cell in the **Output Data** tab, and click **Pre-process**.

This action opens the Data Preprocessing Tool GUI.

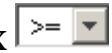


The **Data Editing** area of the Data Preprocessing Tool GUI shows the output data and time vector in the **position1(:,1)** and **Time (seconds)** columns, respectively. The Data Preprocessing Tool GUI lets you perform the following types of preprocessing operations:

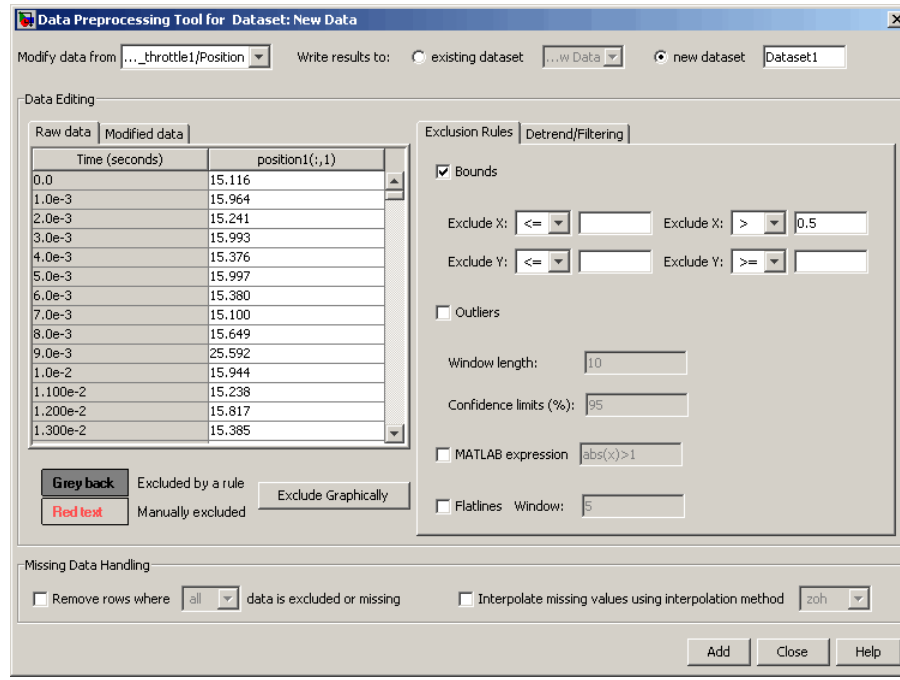
- Excluding data
- Detrending and filtering data
- Handling missing data

3 To exclude the output data beyond t = 0.5 s:

a In the **Exclusion Rules** tab, select the **Bounds** check box.

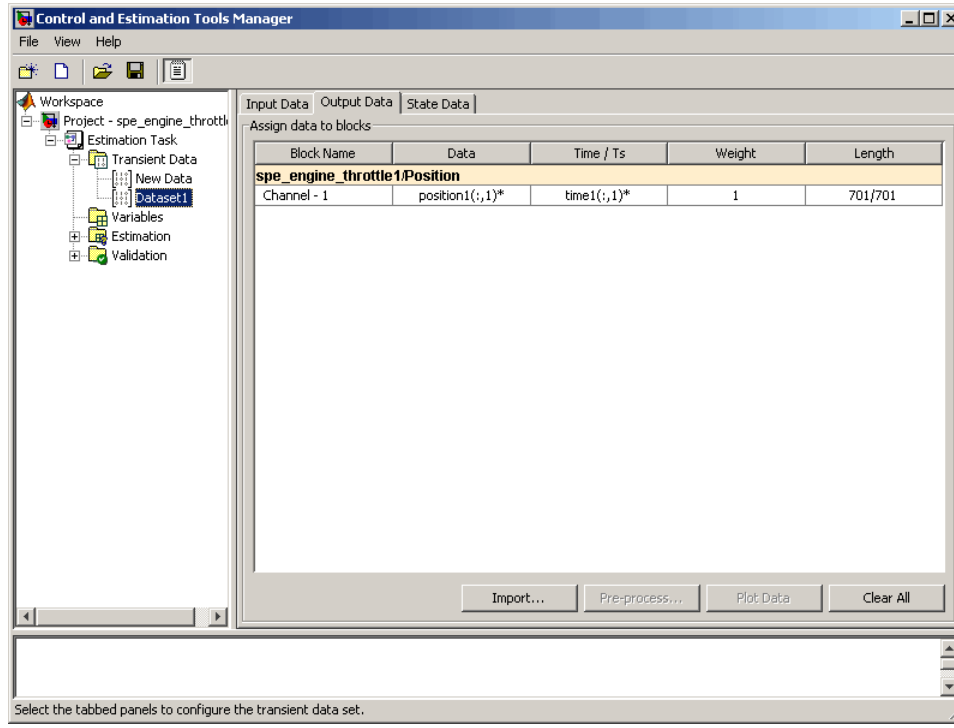
b In the **Exclude X**  field, where **X** corresponds to the time vector, select > from the drop-down list. Enter `0.5` in the adjacent field to specify the upper limit of the data to select for estimation.

The Data Preprocessing Tool GUI resembles the next figure.

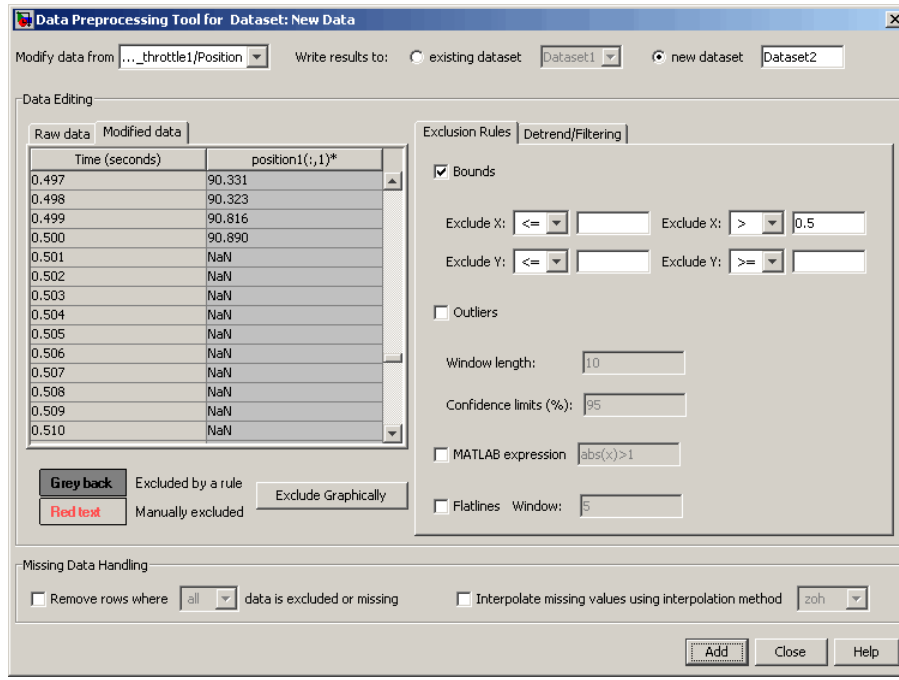


c Click **Add**.

This action adds a new **Dataset1** node under the **Transient Data** node in the Control and Estimation Tools Manager GUI. The **Dataset1** node contains the modified output data `position1(:,1)*` in the **Output Data** tab.



This operation also replaces the output data samples beyond t = 0.5 s in `position1(:,1)*` with `NaN`s. You can view the `NaN`s by selecting the **Modified data** tab in the Data Preprocessing Tool GUI, as shown in the next figure.



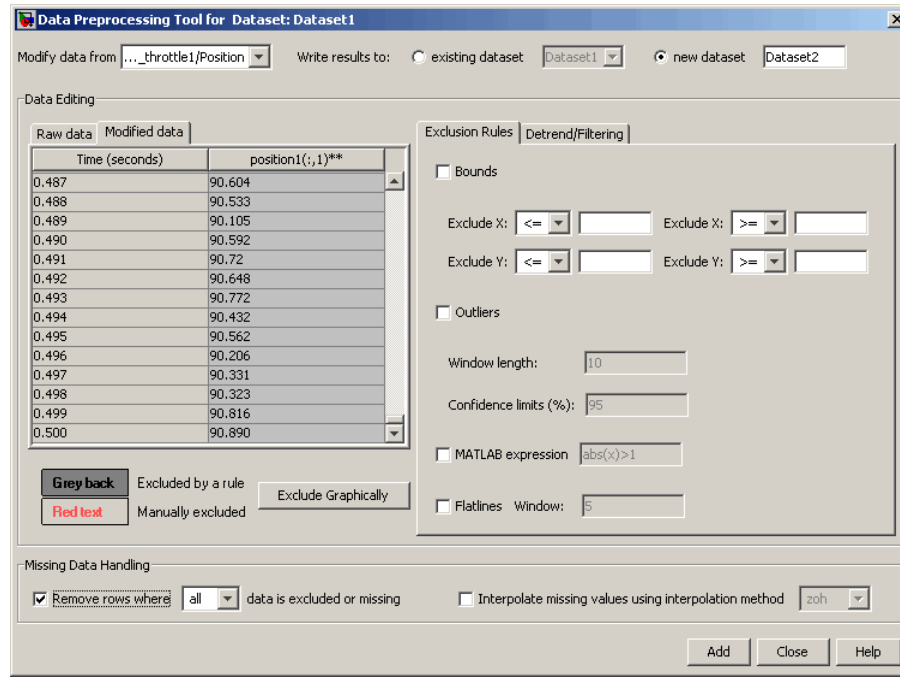
4 To remove the `NaN`s:

a In the Control and Estimation Tools Manager GUI, select the **Output Data** tab of the **Dataset1** node.

b Select the `position1(:,1)*` cell, and click **Pre-process**.

This action updates the Data Preprocessing Tool GUI with the selected data `position1(:,1)*`.

c In the **Missing Data Handling** area, select the **Remove rows where** all **data is excluded or missing** check box.



Tip You can view the results of this operation in the **Modified data** tab.

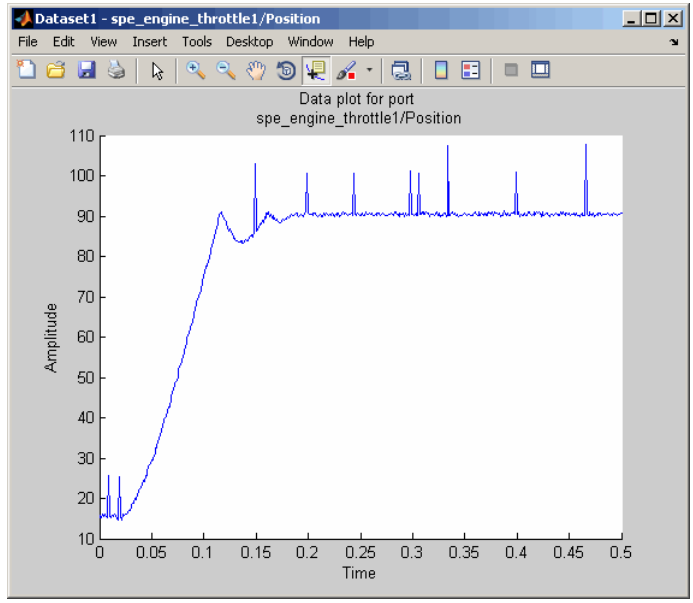
d In the **Write results to** area, select the **existing dataset** Dataset1 option.

e Click **Add**.

The Update table data dialog box appears. Click **Yes** to overwrite the `position1(:,1)*` data set with the modified data.

5 To plot the data, select the `position1(:,1)*` cell in the **Output Data** tab of the **Dataset1** node, and click **Plot Data**.

The selected output data from t = 0 s to t = 0.5 s is shown in the next figure.



Selecting Input Data

After you select the output data between t = 0 s and t = 0.5 s, as described in “Selecting Output Data” on page 5-16, you must also select the corresponding input data samples. This operation makes the number of I/O data samples equal.

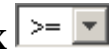
1 In the Control and Estimation Tools Manger GUI, select the **New Data** node under the **Transient Data** node.

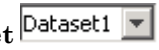
2 In the **Input Data** tab, select the `input1(:,1)` cell, and click **Pre-process**.

This action updates the Data Preprocessing Tool GUI with the selected data `input1(:,1)`.

3 To exclude the data beyond t = 0.5 s:

a In the **Exclusion Rules** tab, select the **Bounds** check box.

b In the **Exclude X**  field, where **X** corresponds to the time vector, select > from the drop-down list. Enter `0.5` in the adjacent field to specify the upper limit of the input data to select for estimation.

c In the **Write results to** area, verify that the **existing dataset**  option remains selected.

d Click **Add**.

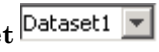
This action adds the modified data `input1(:,1)*` to the **Input Data** tab of the **Dataset1** node. This operation also replaces the input data samples beyond t = 0.5 s in `input1(:,1)*` with `NaN`s.

4 To remove the `NaN`s:

a In the Control and Estimation Tools Manager GUI, select the `input1(:,1)*` cell in the **Input Data** tab of the **Dataset1** node, and click **Pre-process**.

This action updates the Data Preprocessing Tool GUI with the selected data `input1(:,1)*`.

b In the **Missing Data Handling** area, select the **Remove rows where** **data is excluded or missing** check box.

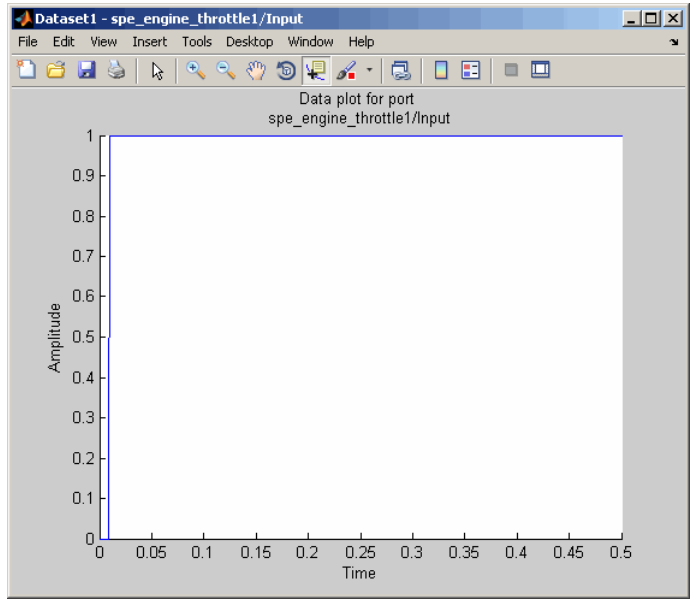
c In the **Write results to** area, verify that the **existing dataset**  option remains selected.

d Click **Add**.

The Update table data dialog box appears. Click **Yes** to overwrite the `input1(:,1)*` data set with the modified data.

5 To plot the data, select the `input1(:,1)*` cell in the **Input Data** tab of the **Dataset1** node, and click **Plot Data**.

The selected input data from t = 0 s to t = 0.5 s is shown in the next figure.



Removing Outliers

In this section...
"Why Remove Outliers" on page 5-25
"How to Remove Outliers" on page 5-25

Why Remove Outliers

Outliers are data values that deviate from the mean by more than three standard deviations. When estimating parameters from data containing outliers, the results may not be accurate.

Removing outliers replaces the data samples containing outliers with `NaN`s, which represent missing data. You interpolate the missing data values in a subsequent operation, as described in "Interpolating Missing Data" on page 5-34.

How to Remove Outliers

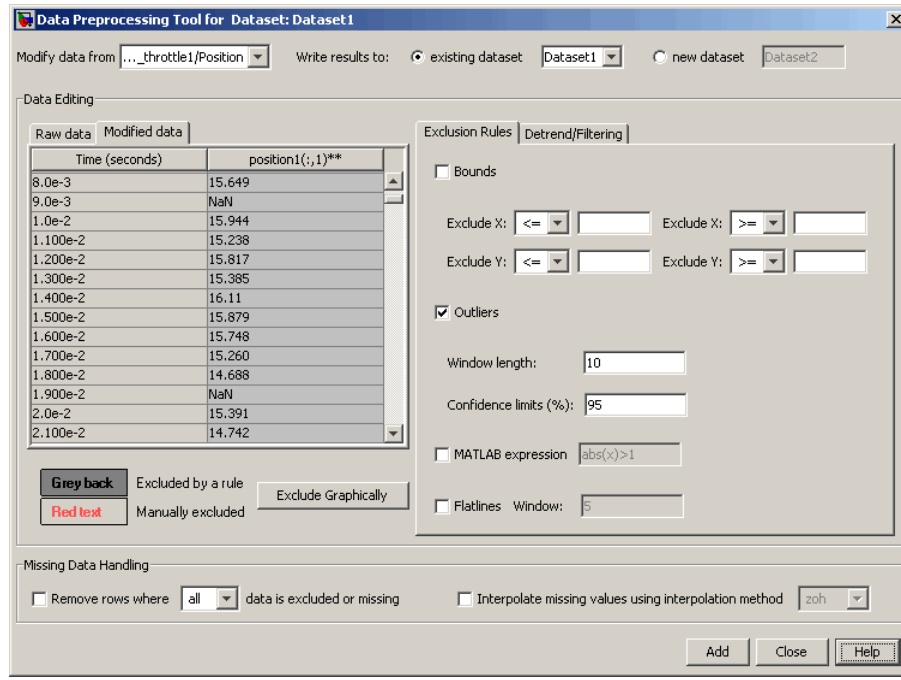
In this portion of the tutorial, you remove outliers from the output data. You must have already selected a subset of the data, as described in "Selecting Data for Estimation" on page 5-16. If you have not done this preparation, click here.

1. In the Control and Estimation Tools Manger, select the `position1(:,1)*` cell in the **Output Data** tab of the **Dataset1** node, and click **Pre-process**.

   This action updates the Data Preprocessing Tool GUI with the selected data `position1(:,1)*`.

2 In the **Exclusion Rules** tab, select the **Outliers** check box.

By default, the **Window length** and **Confidence limits** fields are set to 10 and 95 respectively. The **Window length** field specifies the number of successive data samples the software uses to compute the mean and standard deviation. The **Confidence limits** field specifies the threshold number for identifying outliers. In this example, the mean and standard deviation of 10 successive data samples are computed, and data values that exceed 95% of standard deviation are identified as outliers.



Note The data samples containing outliers are replaced with `NaN`s.

3 In the **Write results to** area, verify that the **existing dataset**

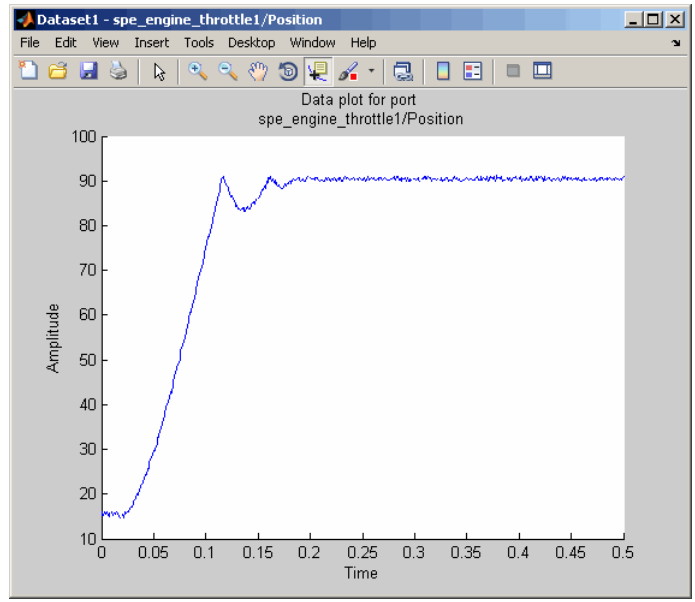
option remains selected.

4 Click **Add**.

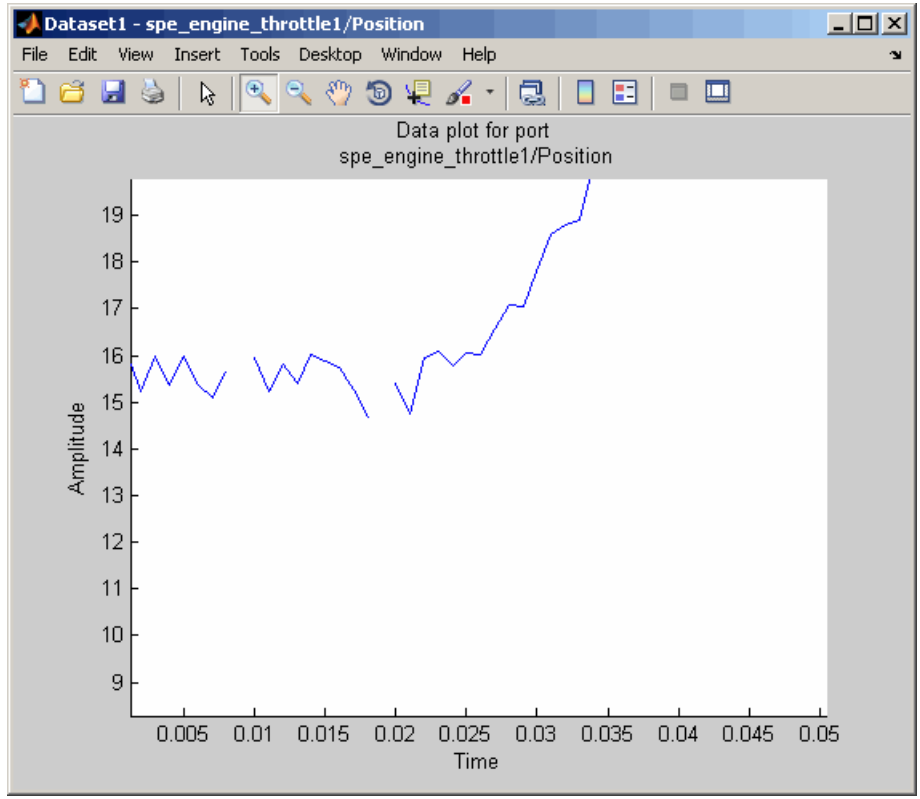
The Update table data dialog box appears. Click **Yes** to overwrite the `position1(:,1)*` data set with the modified data.

5 To plot the data, select the `position1(:,1)*` cell in the **Output Data** tab of the **Dataset1** node, and click **Plot Data**.

The spikes, which indicate outliers, no longer appear on the time plot, as shown in the next figure.



The missing data samples, represented by `NaN`s, appear as gaps on the time plot. To see an example, zoom in to the bottom-left corner of the plot. As shown in the next figure, the data values corresponding to t = 0.009 s and t = 0.019 s are missing.



Filtering Data

In this section...
"Filtering Output Data" on page 5-29
"Filtering Input Data" on page 5-32

Filtering Output Data

In this portion of the tutorial, you filter the noise, and remove any periodic trends from the output data. To avoid relative phase shift between the I/O data, you must also apply the same filter to the input data.

You must have already removed outliers from the output data, as described in "Removing Outliers" on page 5-25. If you have not done this preparation, click here.

1. In the Control and Estimation Tools Manager window, select the `position1(:,1)*` cell in the **Output Data** tab of the **Dataset1** node, and click **Pre-process**.

   This action updates the Data Preprocessing Tool GUI with the selected data `position1(:,1)*`.

2 In the **Detrend/Filtering** tab:

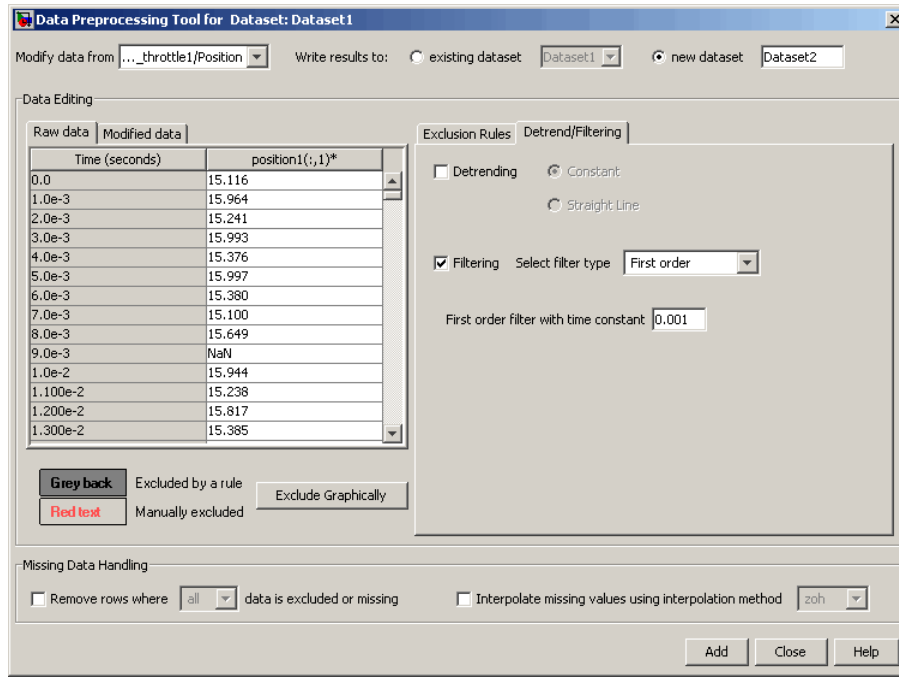
a Select the **Filtering** check box.

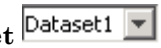
By default, `First order` is selected as the filter type. To learn more about the filters, see "Filtering Data".

b Specify `0.001` in the **First order filter with time constant** field.

This field specifies the time constant for the first-order filter.

Tip For calculating the time constant, you can visually inspect the time plot to determine the frequency components.



3 In the **Write results to** area, verify that the **existing dataset** 

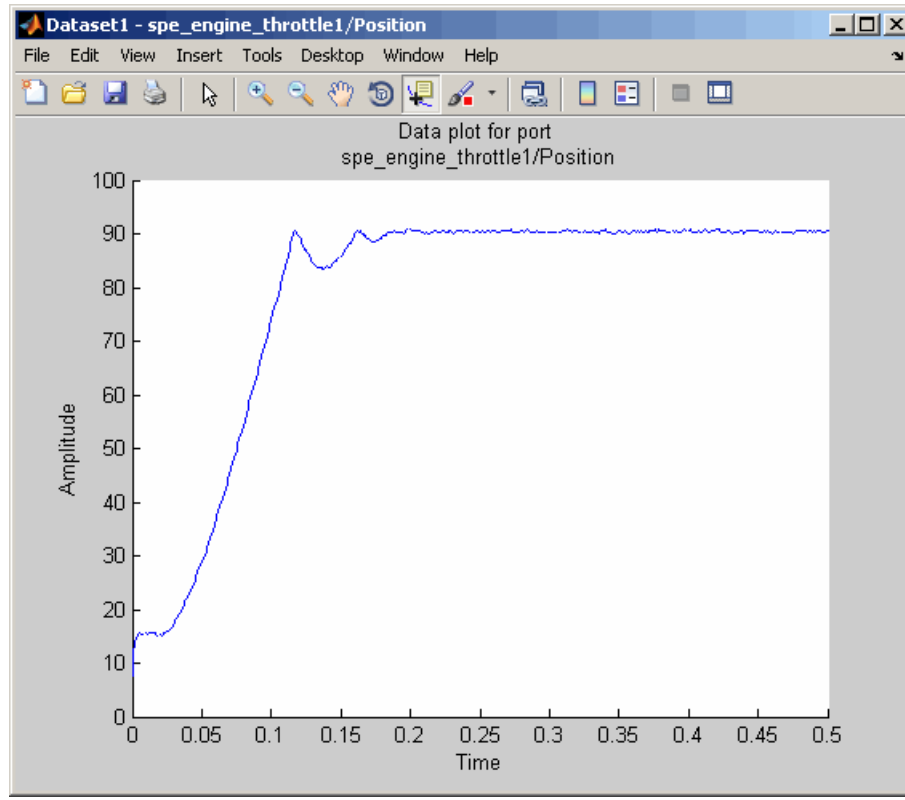
option remains selected.

4 Click **Add**.

The Update table data dialog box appears. Click **Yes** to overwrite the `position1(:,1)*` data set with the modified data.

5 To plot the data, select the `position1(:,1)*` cell in the **Output Data** tab of the **Dataset1** node, and click **Plot Data**.

The noise is filtered and the output data appears smooth, as shown in the next figure.



Filtering Input Data

After you filter the output data, as described in "Filtering Output Data" on page 5-29, you must also filter the input data with the same filter.

1. In the Control and Estimation Tools Manager window, select the `input1(:,1)*` cell in the **Input Data** tab of the **Dataset1** node, and click **Pre-process**.

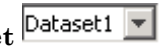
   This action updates the Data Preprocessing Tool GUI with the selected data `input1(:,1)*`.

2. In the **Detrend/Filtering** tab:

   a. Select the **Filtering** check box.

      By default, `First order` is selected as the filter type.

   b. Specify `0.001` in the **First order filter with time constant** field.

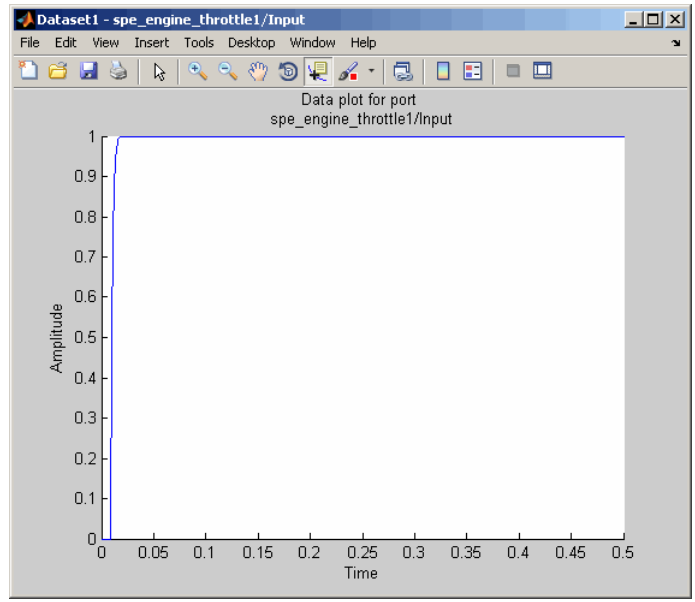
3. In the **Write results to** area, verify that the **existing dataset** 

option remains selected.

4. Click **Add**.

   The Update table data dialog box appears. Click **Yes** to overwrite the `input1(:,1)*` data set with the modified data.

5 To plot the data, select the `input1(:,1)*` cell in the **Input Data** tab of the **Dataset1** node, and click **Plot Data**.



Interpolating Missing Data

In this portion of the tutorial, you interpolate the output data to compute the missing values created when removing outliers, as described in "Removing Outliers" on page 5-25. The interpolation operation uses the known data values to compute the missing data values.

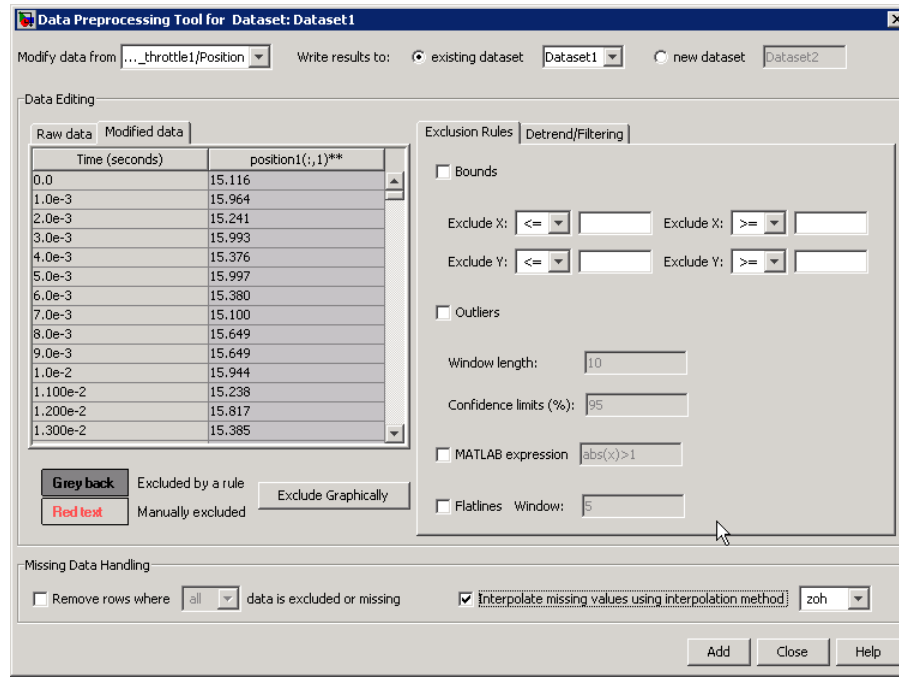
You must have already filtered the noise, as described in "Filtering Data" on page 5-29. If you have not already done this preparation, click here.

1. In the Control and Estimation Tools Manager GUI, select the `position1(:,1)*` cell in the **Output Data** tab of the **Dataset1** node, and click **Pre-process**.

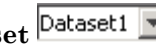
   This action updates the Data Preprocessing Tool GUI with the selected data `position1(:,1)*`.

2 In the **Missing Data Handling** area, select the **Interpolate missing values using interpolation method** check box.

By default, `zoh` is selected as the interpolation method. This method fills the missing data sample with the data value immediately preceding it.



Tip You can view the interpolated data samples in the **Modified data** tab.

3 In the **Write results to** area, verify that the **existing dataset** 

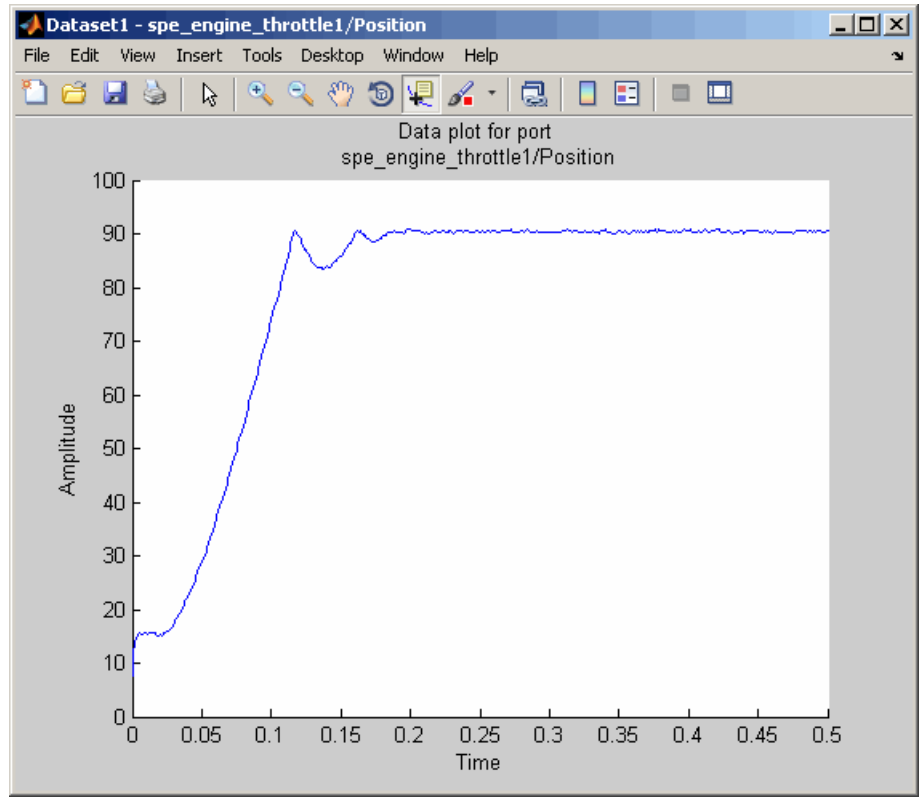
option remains selected.

4 Click **Add**.

The Update table data dialog box appears. Click **Yes** to overwrite the `position1(:,1)*` data set with the modified data.

5 To plot the data, select the `position1(:,1)*` cell in the **Output Data** tab of the **Dataset1** node, and click **Plot Data**.

The new estimation data, prepared by removing outliers, filtering noise, and interpolating missing data, is shown the next figure.

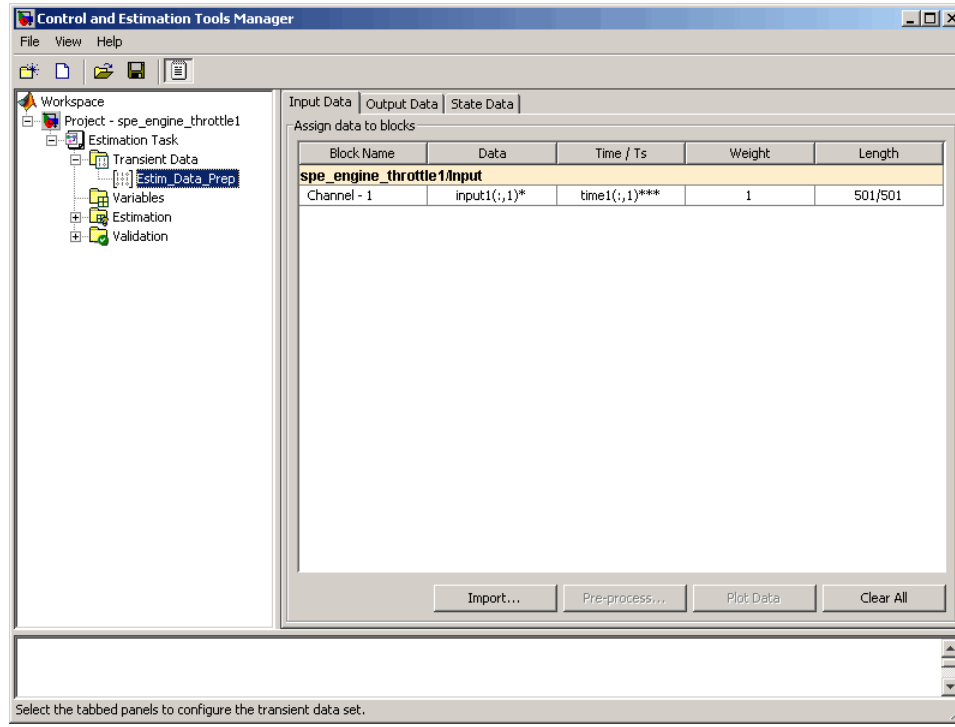


Saving the Project

After you prepare the data, you can delete the data in the **New Data** node, rename the prepared data, and save the session. To skip the data preparation steps, click here.

1. In the Control and Estimation Tools Manger GUI, select the **New Data** node under the **Transient Data** node.
2. Right-click the **New Data** node, and select **Delete**.
3. Select the **Dataset1** node under the **Transient Data** node.
4. Right-click the **Dataset1** node, and select **Rename**. Specify `Estim_Data_Prep` as the name of the new estimation data set.

   The Control and Estimation Tools Manager GUI resembles the next figure.

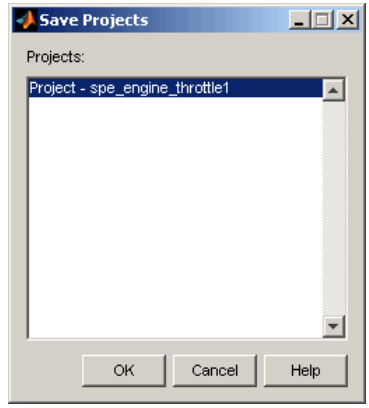


5 Save the Control and Estimation Tools Manager project:

a In the Control and Estimation Tools Manager GUI, select **File > Save**.

This action opens the Save Projects dialog box.

b In the Save Projects dialog box, click **OK**.



c In the Save Projects window, specify `spe_engine_throttle1p.mat` in the **File name** field, and click **Save**.

The action saves the project as a MAT-file.

To learn how to estimate parameters from this data, see Chapter 6, "Tutorial — Estimating Parameters from Measured Data Using the GUI".

6

Tutorial — Estimating Parameters from Measured Data Using the GUI

- "About This Tutorial" on page 6-2
- "Estimating Model Parameters Using Default Estimation Settings" on page 6-7
- "Improving Estimation Results Using Parameter Bounds" on page 6-20
- "Validating Estimated Model Parameters" on page 6-26

About This Tutorial

In this section...

"Objectives" on page 6-2

"About the Model" on page 6-3

Objectives

In this tutorial, you learn how to estimate parameters of a single-input single-output (SISO) Simulink model from measured input and output (I/O) data.

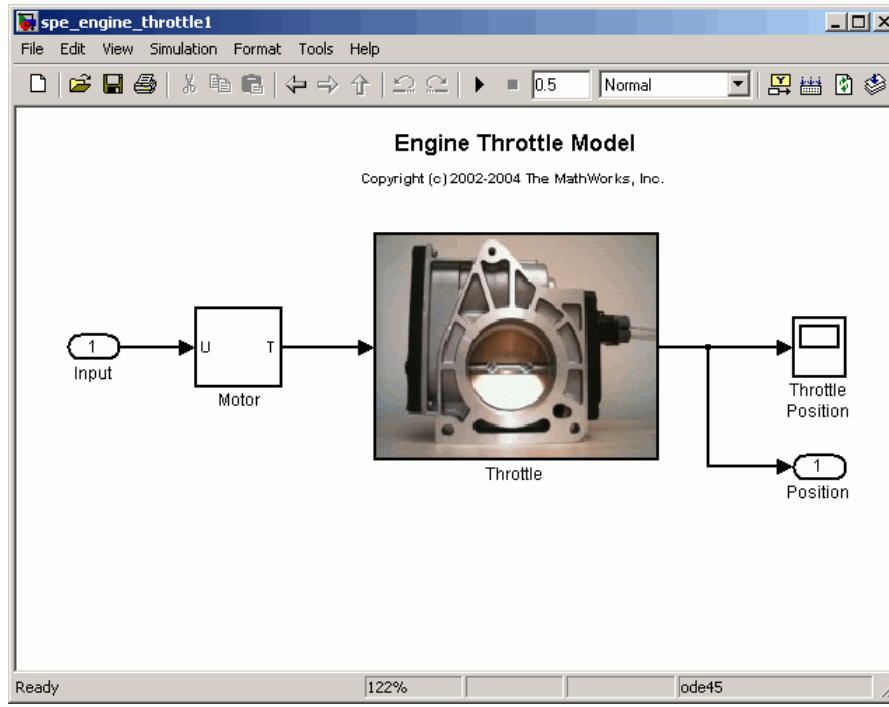
Note Simulink Design Optimization software estimates parameters from real, time-domain data only.

You learn to perform the following tasks using the GUI:

- Load a saved project containing data.
- Estimate model parameters using default settings.
- Validate the model, and refine the estimation results.

About the Model

In this tutorial, you use the `spe_engine_throttle1` Simulink model. This model represents an engine throttle system, as shown in the next figure.

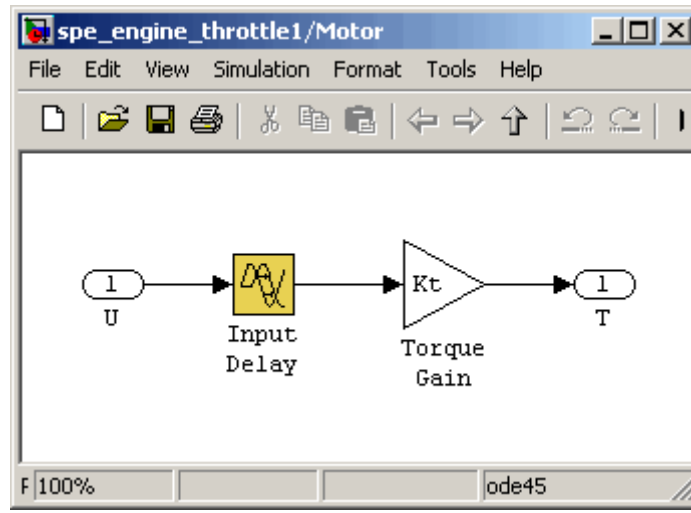


The throttle system controls the flow of air and fuel mixture to the engine cylinders. The throttle body contains a butterfly valve which opens when a driver presses the accelerator pedal. Opening this valve increases the amount of fuel mixture entering the cylinders, which increases the engine speed. A DC motor controls the opening angle of the butterfly valve in the throttle system. The models for these components are described in “Motor Subsystem” on page 6-4, and “Throttle Subsystem” on page 6-5.

The input to the throttle system is the motor current (in amperes), and the output is the angular position of the butterfly valve (in degrees).

Motor Subsystem

The `Motor` subsystem contains the DC motor model. To open the model, double-click the corresponding block.

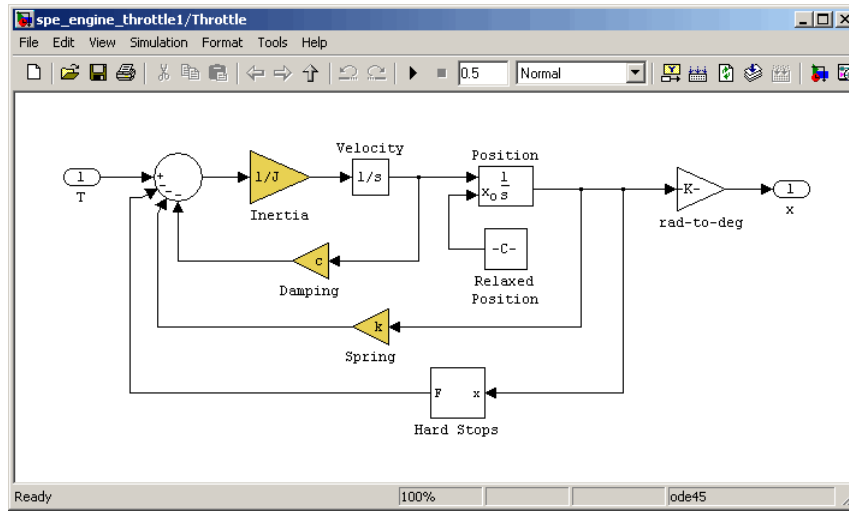


The following table describes the variables, parameters, equation, input, and output of the `Motor` subsystem.

Variables	U is the input current to the motor. T is the torque applied by the motor.
Parameters	K_t is the torque gain of the motor, represented by `Kt` in the model. t_d is the input time delay of the motor, represented by `input_delay` in the model.
Equation	The torque applied by the motor is described in the following equation: $T(t) = K_t U(t - t_d)$ where t is time.
Input	U
Output	T

Throttle Subsystem

The `Throttle` subsystem contains the butterfly valve model. To open the model, right-click the corresponding block, and select **Look Under Mask**.



The `Hard Stops` block models the valve angular position limit of 15° to 90°.

The following table describes the variables, parameters, states, differential equations, inputs, and outputs of the `Throttle` subsystem.

Variables	T is the torque applied by the DC motor. θ is the angular position of the valve, represented by `x` in the model. $T_{hardstop}$ is the torque applied by the hard stop.
Parameters	`J` is the inertia. `c` is the viscous friction. `k` is the spring constant.
States	θ is the angular position. $\dot{\theta}$ is the angular velocity.

Equations	The mathematical system for the butterfly valve is described in the following equation: $$J\ddot{\theta} + c\dot{\theta} + k\theta = T + T_{hardstop}$$ where $15^\circ \le \theta \le 90^\circ$, with initial conditions $\theta_0 = 15^\circ$, and $\dot{\theta}_0 = 0$. The torque applied by the `Hard Stops` block is described in the following equation: $$T_{hardstop} = \left\{ \begin{array}{ll} 0, & 15^\circ \le \theta \le 90^\circ \\ K(90^\circ - \theta), & \theta > 90^\circ \\ K(15^\circ - \theta), & \theta < 15^\circ \end{array} \right\}$$ where `K` is the gain of the `Hard Stops` block.
Input	T
Output	θ

Estimating Model Parameters Using Default Estimation Settings

In this section...
"Overview of the Estimation Process" on page 6-7
"Specifying Parameters and Estimation Data" on page 6-8
"Validating Model Parameters" on page 6-13

Overview of the Estimation Process

Simulink Design Optimization software uses optimization techniques to estimate model parameters. In each optimization iteration, the model is simulated with the current parameter values. The error between the simulated and measured output is computed and minimized. The estimation is complete when the optimization method finds a local minimum.

You perform the following tasks to estimate the model parameters:

- "Specifying Parameters and Estimation Data" on page 6-8
- "Validating Model Parameters" on page 6-13

Note The Simulink model must remain open to perform parameter estimation tasks.

Specifying Parameters and Estimation Data

To specify parameters and estimation data:

1. Load the preconfigured project `spe_engine_throttle1p.mat`.

   This MAT-file contains the estimation data. To learn more about the data and how to prepare it, see Chapter 5, "Tutorial — Preparing Data for Parameter Estimation Using the GUI".

   To load the preconfigured project:

   a. Open the engine throttle system Simulink model by typing the following at the MATLAB prompt:

      ```
      spe_engine_throttle1
      ```

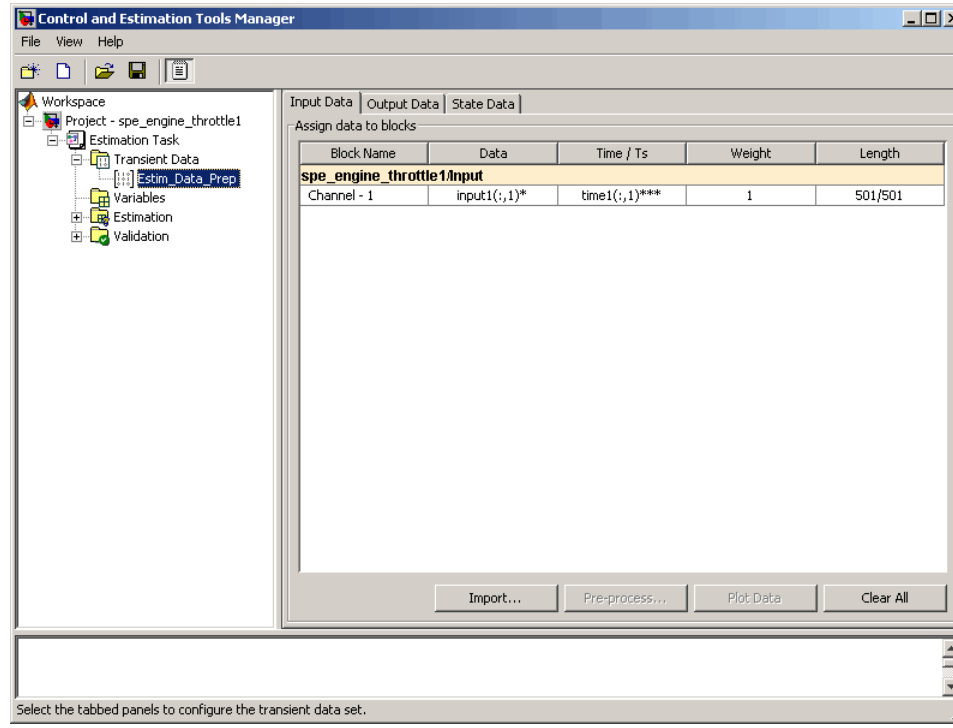
   b. In the Simulink model window, select **Tools > Parameter Estimation**.

      This action opens a new project named **Project - spe_engine_throttle1** in the Control and Estimation Tools Manager GUI.

   c. In the Control and Estimation Tools Manager GUI, select **File > Load**. Browse to the *matlabroot*`\toolbox\sldo\sldodemos\estim` directory, and select `spe_engine_throttle1p.mat`.

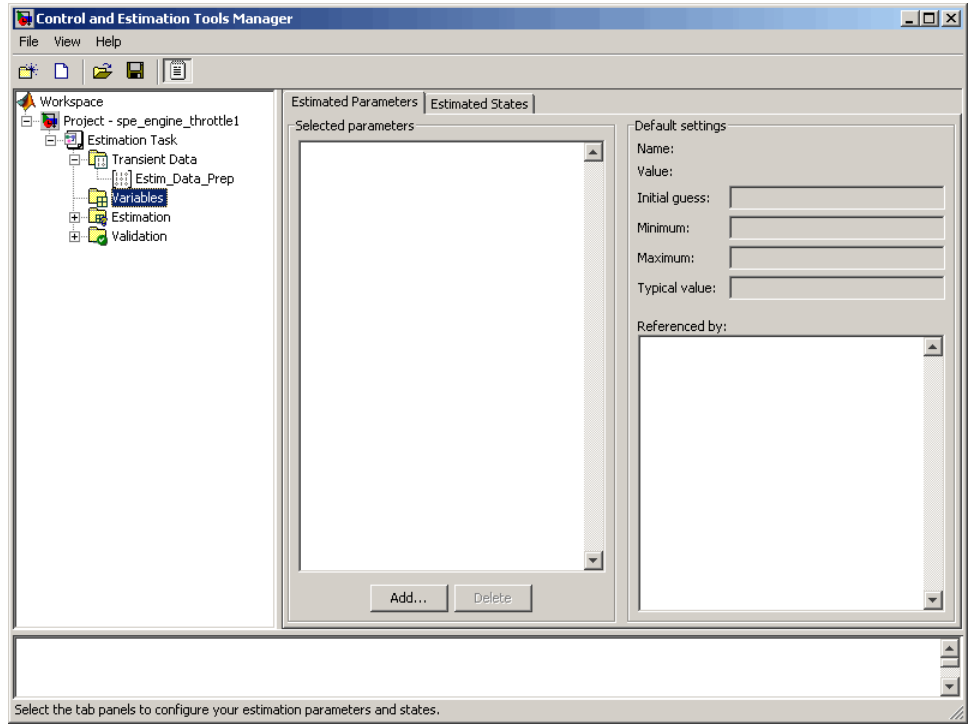
      The Confirm Node Replace dialog box appears. Click **Yes** to load the project into the Control and Estimation Tools Manager GUI.

The Control and Estimation Tools Manager GUI opens as shown in the next figure.



2 Specify parameters for estimation.

a Select the **Variables** node under the **Estimation Task** node.

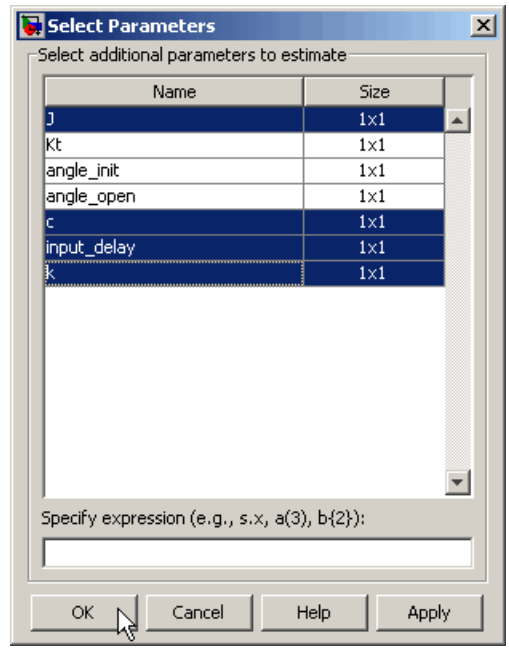


b Click **Add**.

This action opens the Select Parameters dialog box, which shows the model parameters for the Simulink model.

c Select the parameters `J`, `c`, `input_delay`, and `k` by pressing the **Ctrl** key while clicking each name, and then click **OK**.

This action adds the selected parameters to the **Estimated Parameters** tab.



Tip When estimating a large number of parameters, you can select a subset of parameters to estimate. To learn more, see the Inverted Pendulum Parameter Estimation demo.

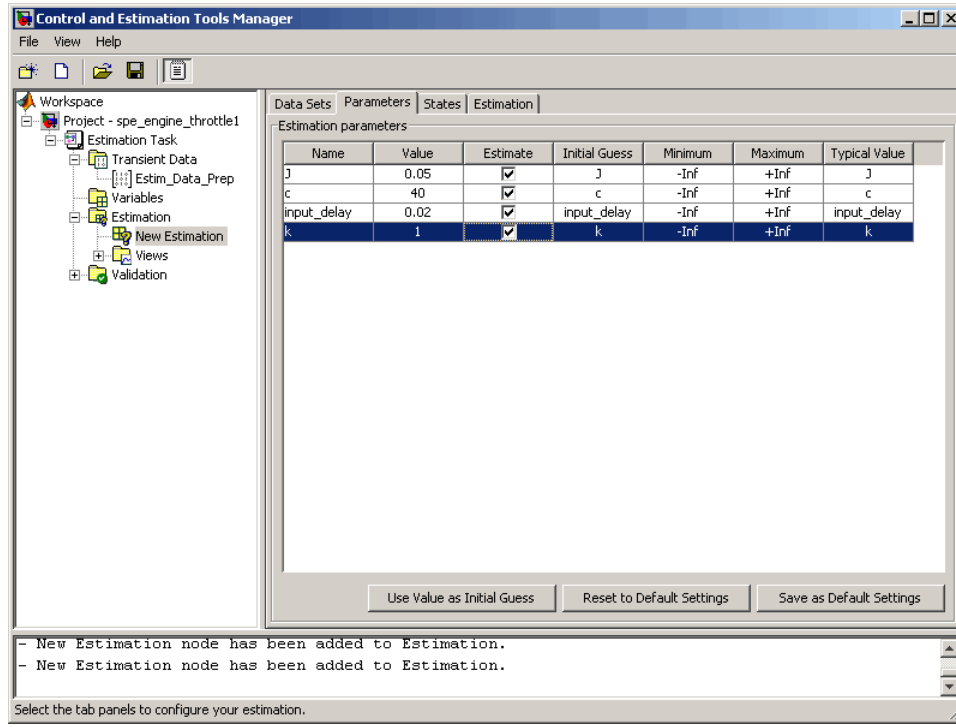
d Select the **Estimation** node, and click **New**.

This action adds a **New Estimation** node under the **Estimation** node.

e Select the **New Estimation** node.

f In the **Parameters** tab, select the **Estimate** check box for all the parameters.

This action specifies the selected parameters for estimation.



The **Parameters** tab, as shown in the previous figure, displays the following information for each parameter:

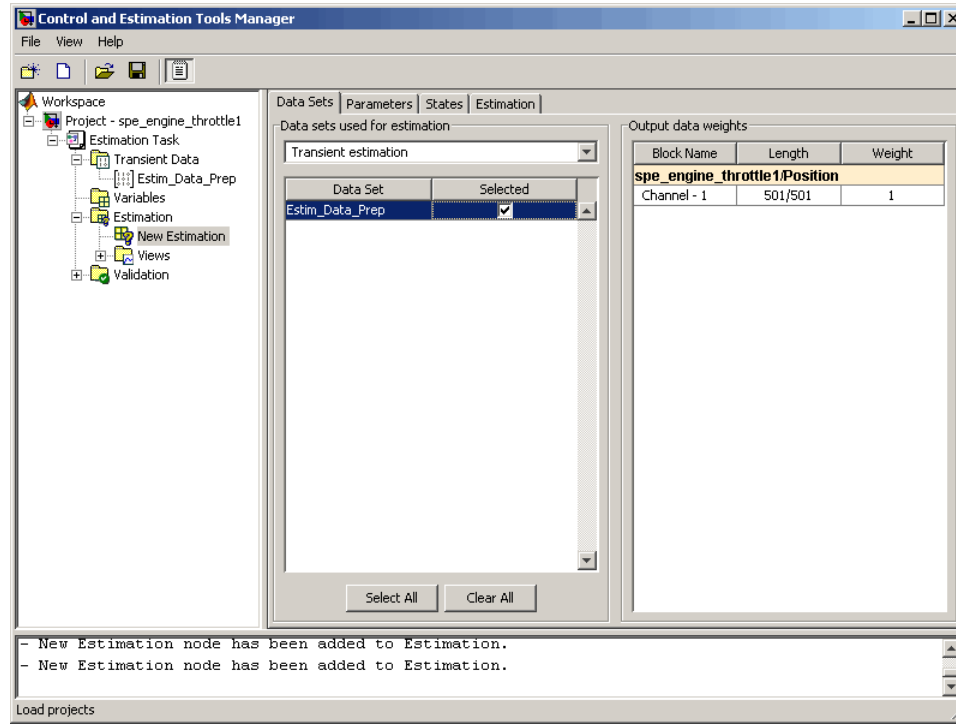
- **Value**: Current parameter value. By default, it is equal to the value specified in the Simulink model.

  During estimation, the optimization method might change a parameter value to minimize the error between the measured and simulated output.

- **Initial Guess**: Initial parameter value. By default, it is equal to the value in the **Value** field and is used at the start of estimation.

- **Minimum** and **Maximum**: Bounds on the parameter value. By default, they are set to `-Inf`, and `+Inf`, respectively. This means that the optimization method searches for a solution in the range `[-Inf, +Inf]`.
- **Typical Value**: Order of magnitude of the parameter value. By default, it is equal to the value in the **Initial Guess** field.

3 Specify data for estimation by checking the **Selected** check box for **Estim_Data_Prep** in the **Data Sets** tab.



Validating Model Parameters

After you specify parameters and data for estimation, as described in “Specifying Parameters and Estimation Data” on page 6-8, estimate the parameters and analyze the results to determine if the model needs to be refined.

1 Create plots to view the estimation results.

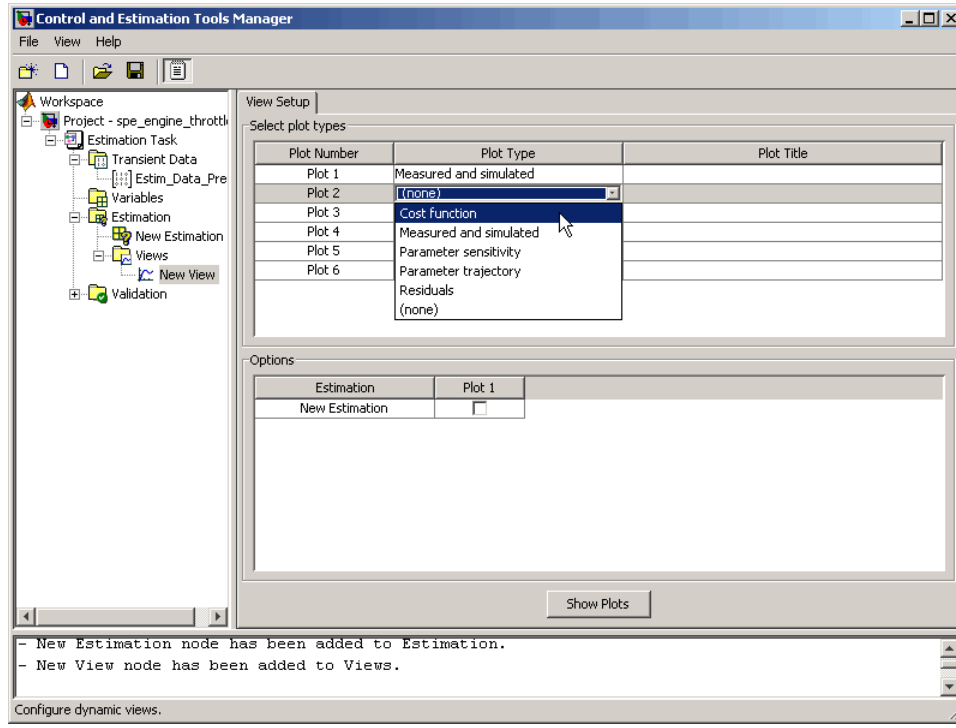
   a Select the **Views** node under the **Estimation** node, and click **New**.

   This action creates a **New View** node under the **Views** node.

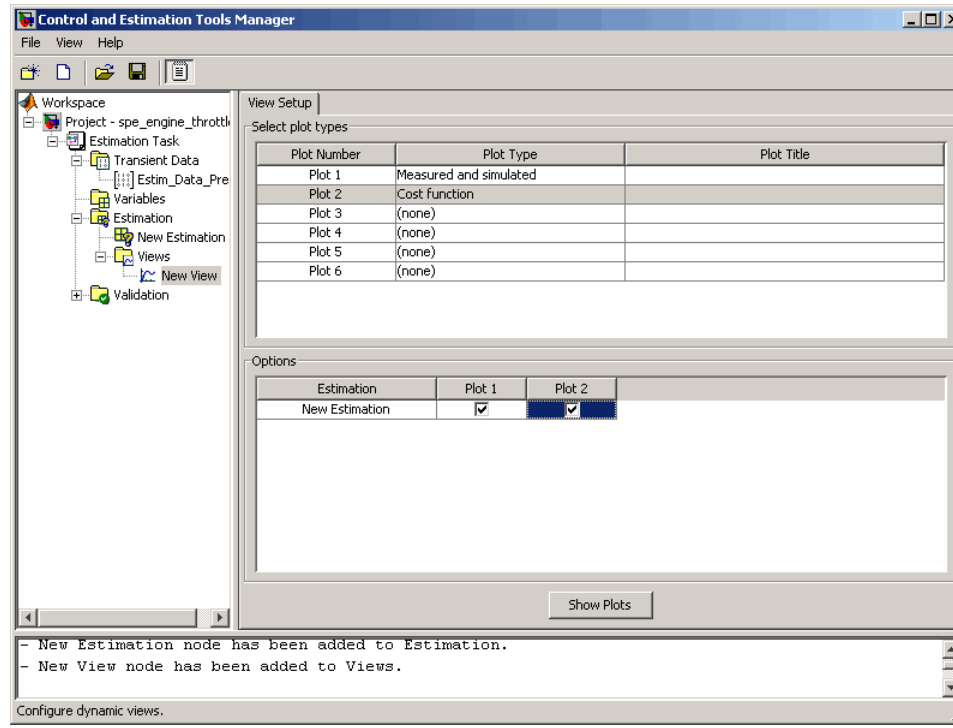
   b Select the **New View** node.

   c In the **View Setup** tab, select the following plots from the drop-down list in the **Plot Type** column:

   - For **Plot 1**, select `Measured and simulated`.
   - For **Plot 2**, select `Cost function`.



d In the **Options** area, select the **Plot 1** and **Plot 2** check boxes, and click **Show Plots**.

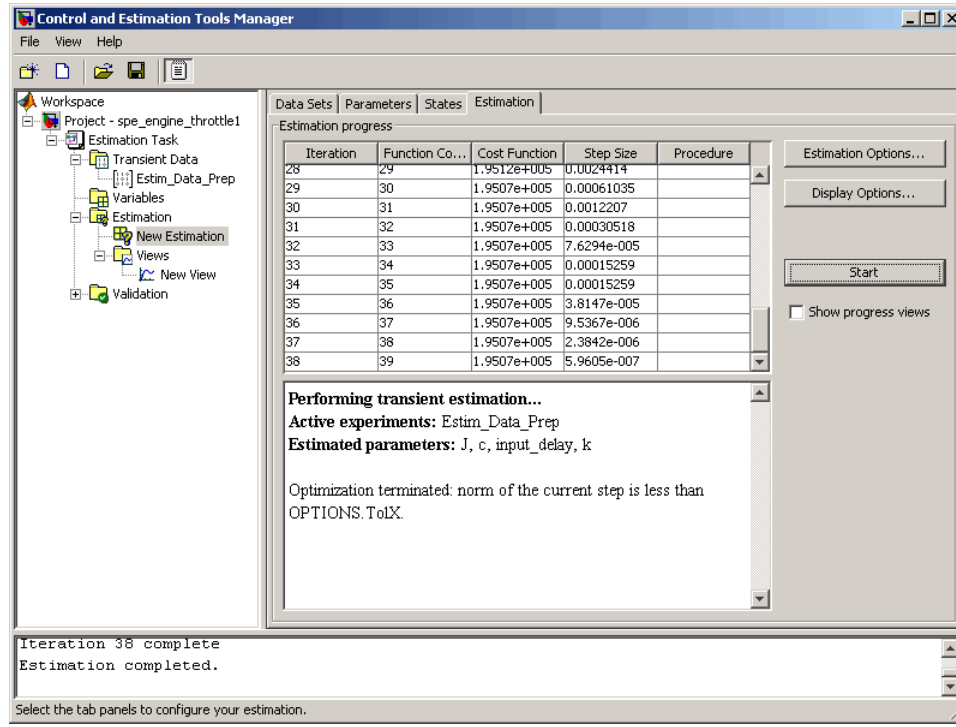


Clicking **Show Plots** opens the following figures:

- `New View - Plot 1 (Measured and simulated)`: Displays the measured output data `Estim_Prep_Data`. During estimation, this plot updates to display the simulated response at each iteration.
- `New View - Plot 2 (Cost function)`: Displays the error between the measured and simulated output, computed at each iteration. By default, the error is computed using *sum- of- squared- errors*, which is a least-squares method.

2 Estimate the parameters by selecting the **Estimation** tab of the **New Estimation** node, and clicking **Start**.

The **Estimation** tab updates at each iteration, and provides information about the estimation progress. When the estimation completes, the **Estimation** tab resembles the next figure.



Note The results of the optimization may differ slightly because of different numerical precision across platforms.

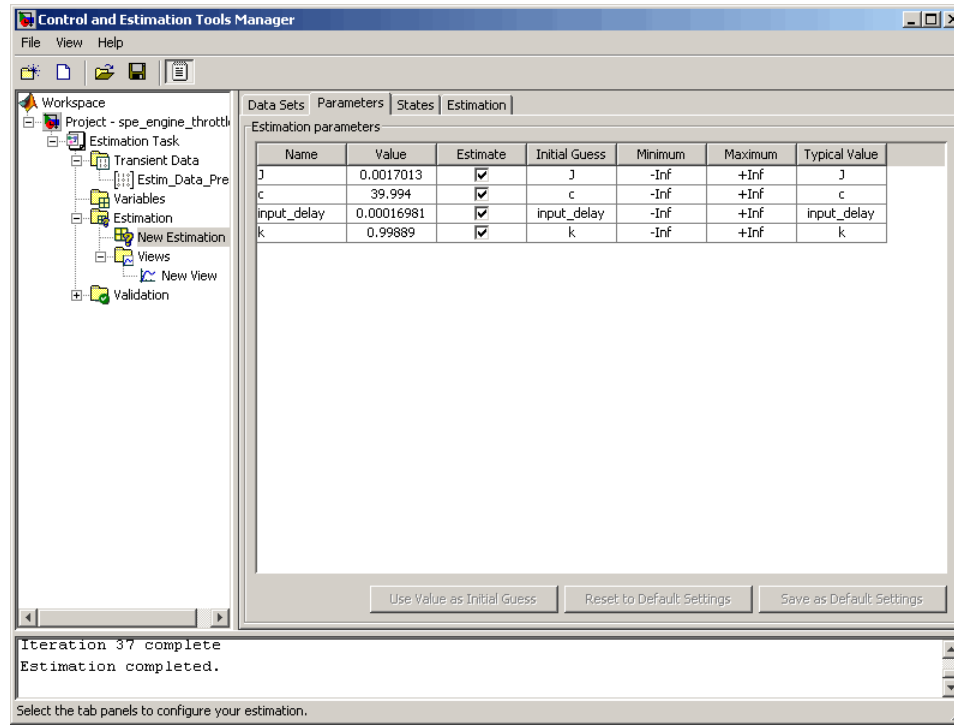
The table displays the following information for each iteration:

- **Cost Function**: Error between the simulated and measured output.

  During estimation, the default optimization method `Nonlinear least squares`, lsqnonlin, minimizes the cost function by changing the parameter values. As shown in the previous figure, the cost function decreases by only 27% from 2.69e5 to 1.95e5.

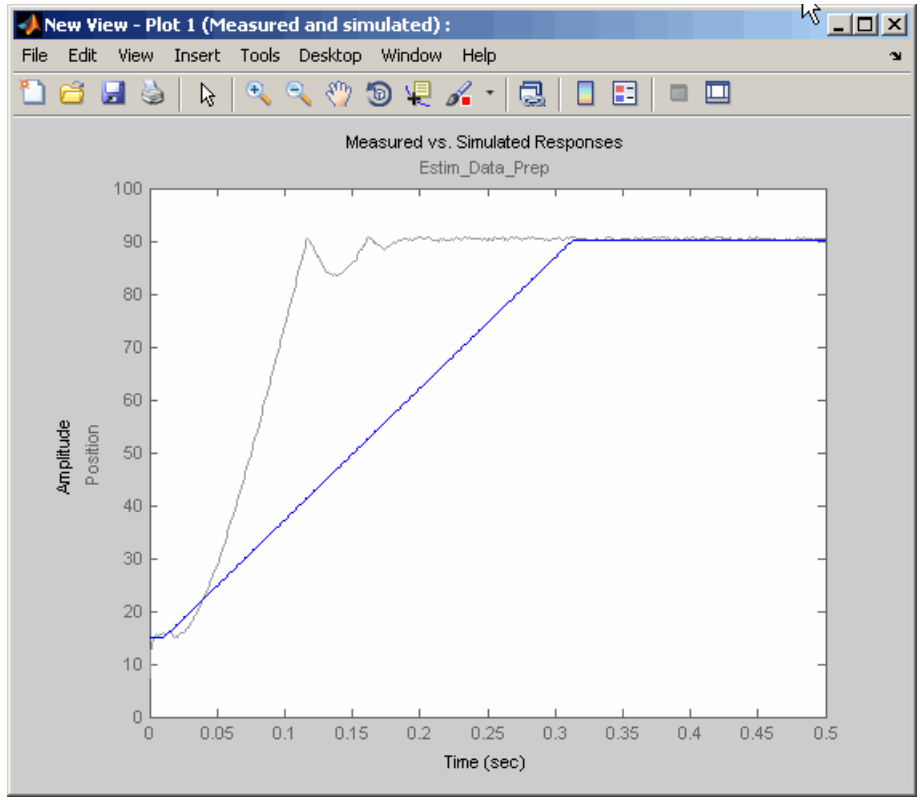
- **Step Size**: Displacement of the optimization method in the current search direction when minimizing the cost function. A final value close to zero indicates that the method has found a local minimum. As shown in the previous figure, the final step size in the last iteration is 5.96e-7.

3 View the estimated parameter values in the **Value** column of the **Parameters** tab.



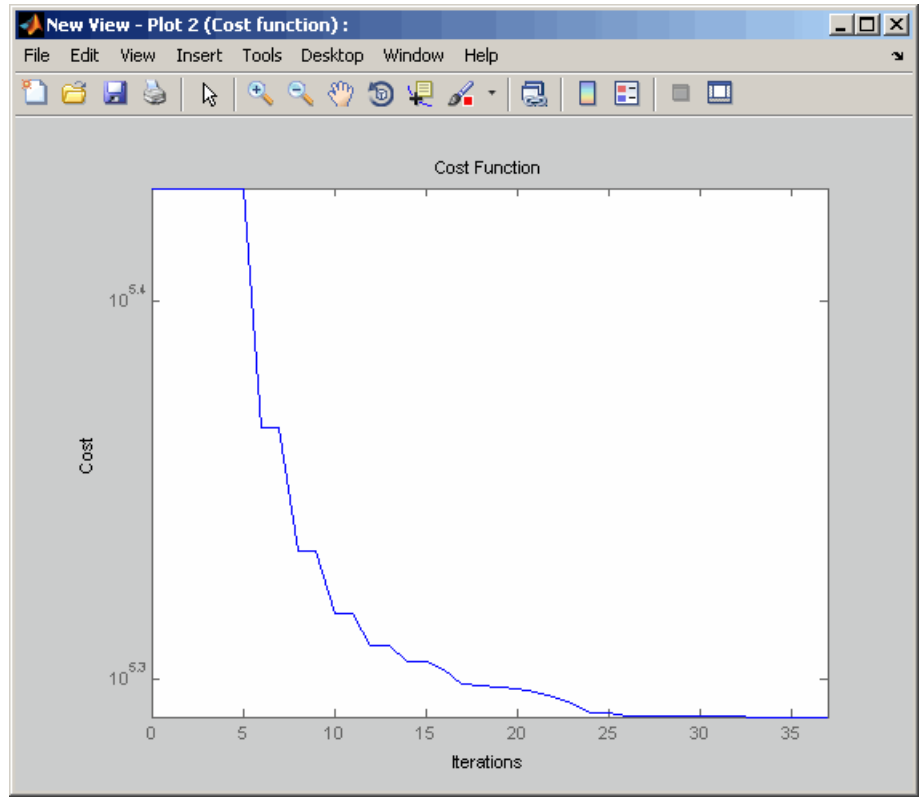
4 Examine the simulated response plot to see how well the simulated output matches the measured output.

The simulated response with the estimated parameters is shown in blue on the `New View - Plot 1 (Measured and simulated)` plot, and significantly differs from the measured data. This difference indicates that the estimated parameters are not accurate.



5 Examine the cost function plot to see how the cost function changes during the estimation.

The cost function values shown in step 2 are plotted on the `New View - Plot 2(Cost function)` plot. The cost function decreases by only 27% from its initial value, and in conjunction with the measured and simulated output plot, indicates that the estimated parameters are not accurate.



To improve the estimation results, you apply bounds on the parameter values, as described in "Improving Estimation Results Using Parameter Bounds" on page 6-20.

Improving Estimation Results Using Parameter Bounds

In this section...

"Strategy for Improving the Estimation Results" on page 6-20

"How to Specify Parameter Bounds" on page 6-20

Strategy for Improving the Estimation Results

Analyzing the `Measured and Simulated` and `Cost function` plots, as described in "Validating Model Parameters" on page 6-13, you see that the model parameters estimated using the default estimation settings are not accurate. You must run additional estimations to improve the accuracy of the model. There are several techniques to improve the accuracy of the estimated parameters. For more information, see "How to Specify Estimation Options in the GUI".

In this portion of the tutorial, you improve the results by specifying bounds on parameter values. This technique restricts the region in which the optimization method searches for a local minima.

Based on physical insight, you know the following characteristics of the engine throttle system:

- All the parameter values are positive.
- Maximum time delay of the system, represented by `input_delay`, is 0.1 s.

Therefore, you specify 0 as the minimum value for all parameters, and 0.1 as the maximum value of `input_delay`.

After you estimate the parameters, analyze the results using the `Measured and Simulated` and `Cost function` plots.

How to Specify Parameter Bounds

You must have already estimated the parameters using default settings, as described in "Estimating Model Parameters Using Default Estimation Settings" on page 6-7.

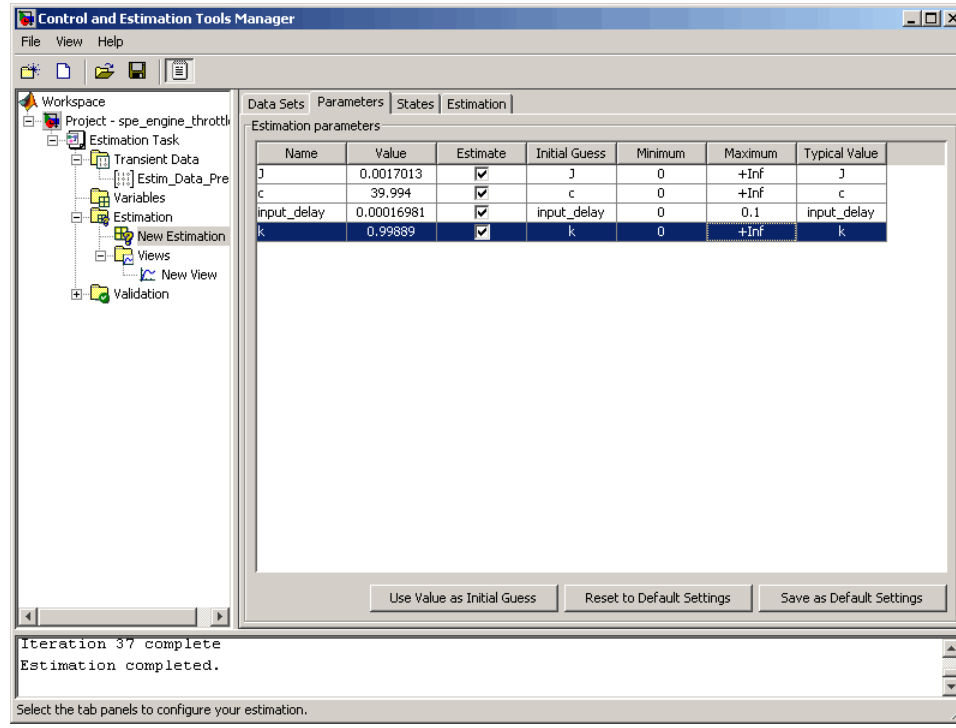
To improve the estimation results by specifying parameter bounds:

1. Select the **Parameters** tab of the **New Estimation** node.

2. Specify the minimum value for each parameter by double-clicking the corresponding **Minimum** cell, and replacing `-Inf` with `0`.

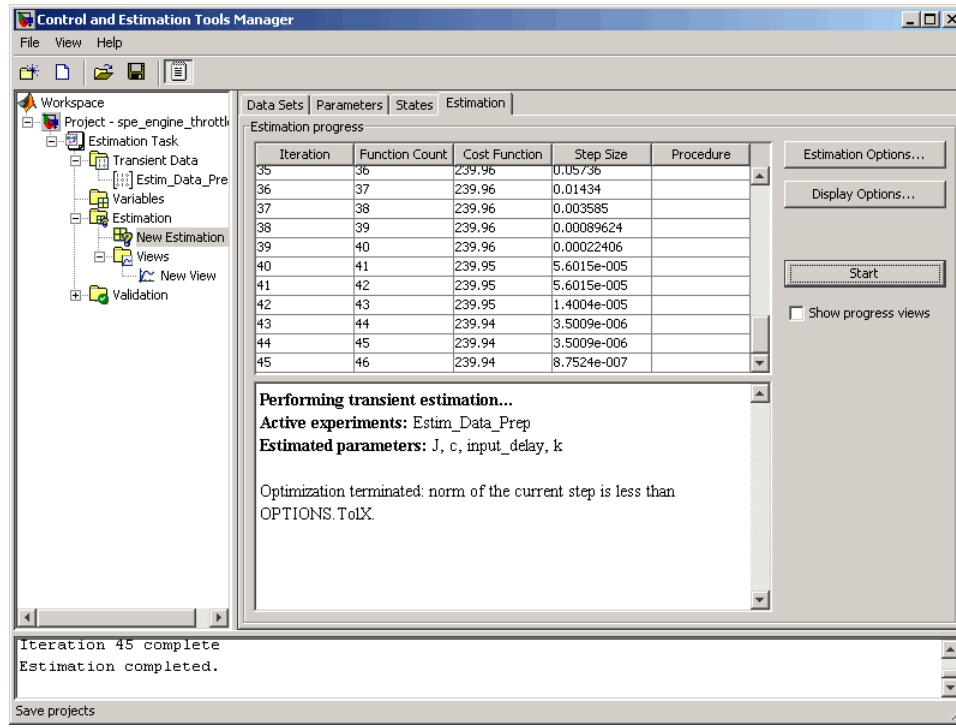
3. Specify the maximum value for `input_delay` by double-clicking the corresponding **Maximum** cell and replacing `+Inf` with `0.1`.

   The **Parameters** tab resembles the next figure.



4 Select the **Estimation** tab, and click **Start**.

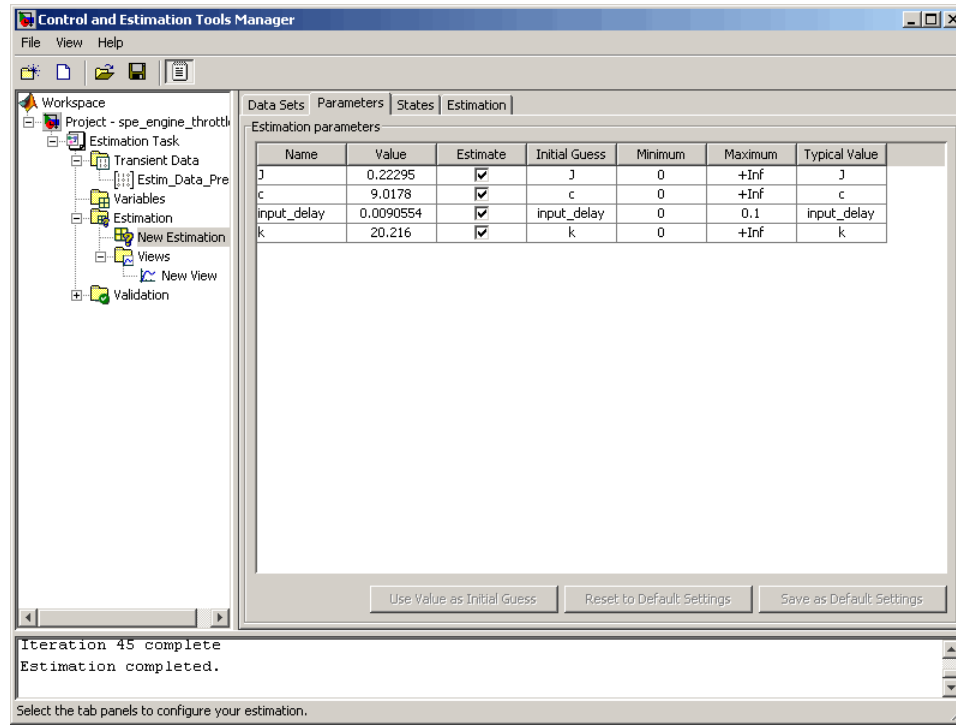
The **Estimation** tab updates at each iteration, and provides information about the estimation progress. When the estimation completes, the **Estimation** tab resembles the next figure.



Note The results of the optimization may differ slightly because of different numerical precision across platforms.

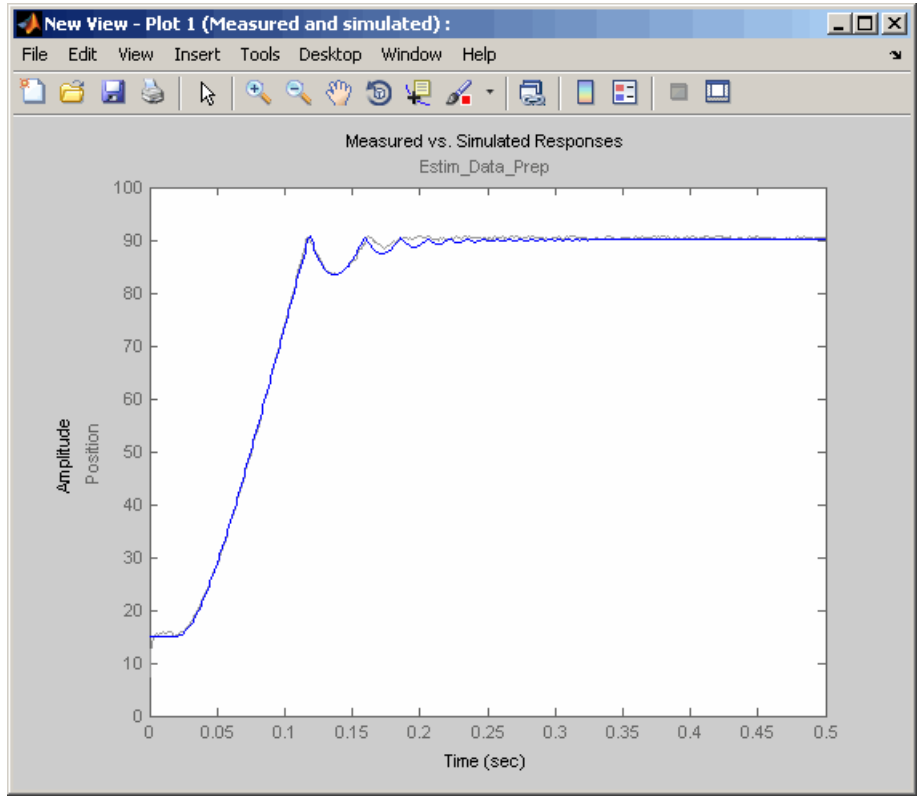
The cost function decreases by 99% from 1.95e5 to 239.94. The final step size 8.75e-7 is close to 0, which indicates that the optimization method has found a local minimum.

5 View the estimated parameter values in the **Value** column of the **Parameters** tab.



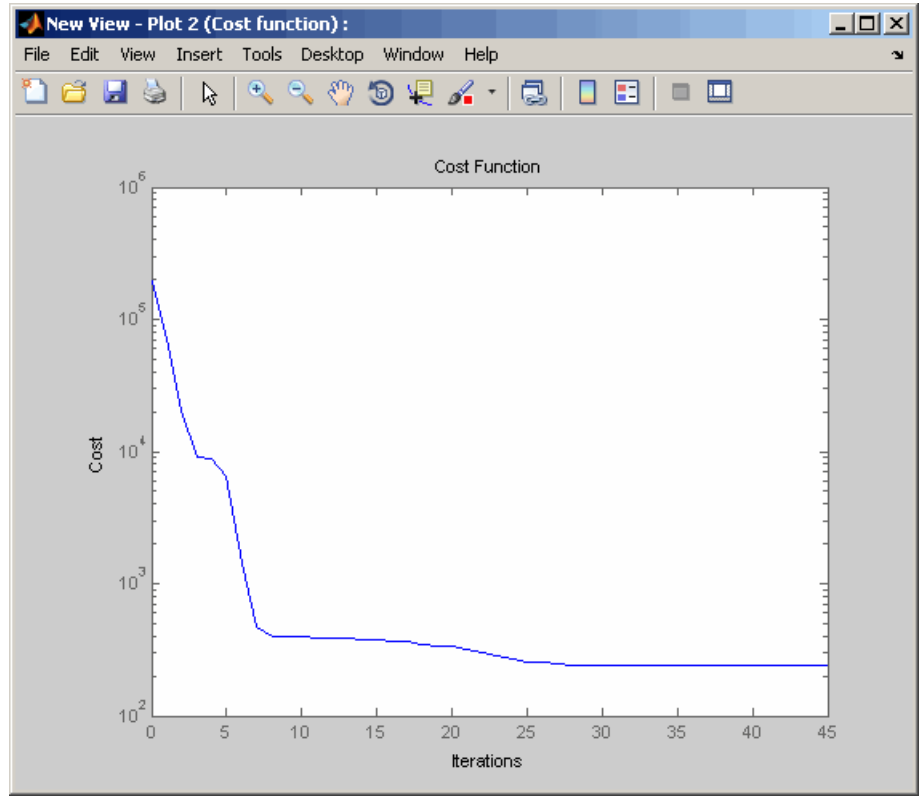
6 Examine the simulated response plot to see how well the simulated output matches the measured output.

The simulated response plot, shown in blue on the `New View - Plot 1 (Measured and simulated)` plot, is overlaid on the measured output data. The simulated output closely matches the measured data.



7 Examine the cost function plot to see how the cost function changes during the estimation.

The cost function values shown in step 4 are plotted on the `New View - Plot 2(Cost function)` plot. The cost function also decreases by 99% from its initial value, and in conjunction with the measured and simulated response plot indicates a good fit of the simulated response with the measured data.



Validating Estimated Model Parameters

After you estimate the model parameters, as described in "Improving Estimation Results Using Parameter Bounds" on page 6-20, validate the model using another data set *(validation data)*. A good match between the simulated response and the validation data indicates that you have not overfitted the model.

To validate the estimated parameters using a validation data set:

1. Load the preconfigured project `spe_engine_throttle1_importValData1.mat`.

   This MAT-file contains a validation data set already imported into the GUI, and the estimated parameters as described in "Improving Estimation Results Using Parameter Bounds" on page 6-20.

   Tip To learn how to import data and time vector into the Control and Estimation Tools Manager GUI, see "Importing Data into the GUI" on page 5-6.

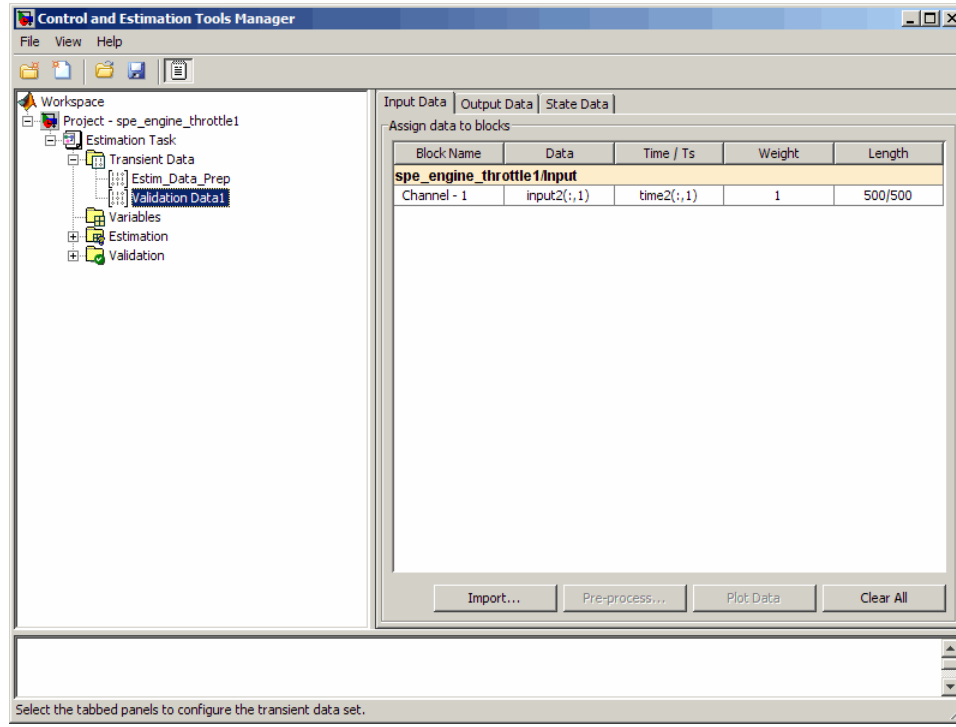
   To load the preconfigured project:

   a. In the Control and Estimation Tools Manager GUI, select **File > Load**.

   b. Browse to the *matlabroot*`\toolbox\sldo\sldodemos\estim` directory, and select `spe_engine_throttle1_importValData1.mat`.

      The Confirm Node Replace dialog box appears. Click **Yes** to load the project into the Control and Estimation Tools Manager GUI.

The Control and Estimation Tools Manager GUI now has a node that contains the validation data, as shown in the next figure.



The validation data contains the input data, output data and time vector in the MATLAB variables `input2`, `position2` and `time2` respectively.

2 Plot the measured and simulated response, and residuals.

a Select the **Validation** node, and click **New**.

This action creates a **New Validation** node.

b Select the **New Validation** node.

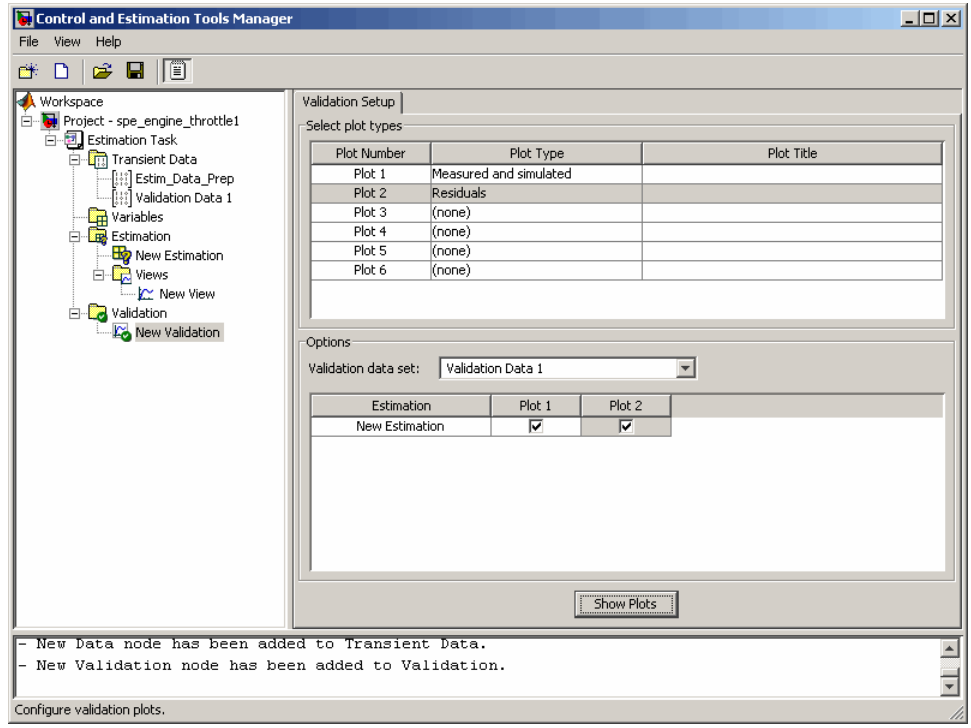
c In the **Validation Setup** tab, select the following plots in the **Plot Type** column.

- For **Plot 1**, select `Measured and simulated`.
- For **Plot 2**, select `Residuals`.

d In the **Options** area, select the **Plot 1**, and **Plot 2** check boxes.

e Select `Validation Data 1` from the **Validation data set** drop-down list.

The **Validation Setup** tab resembles the next figure.



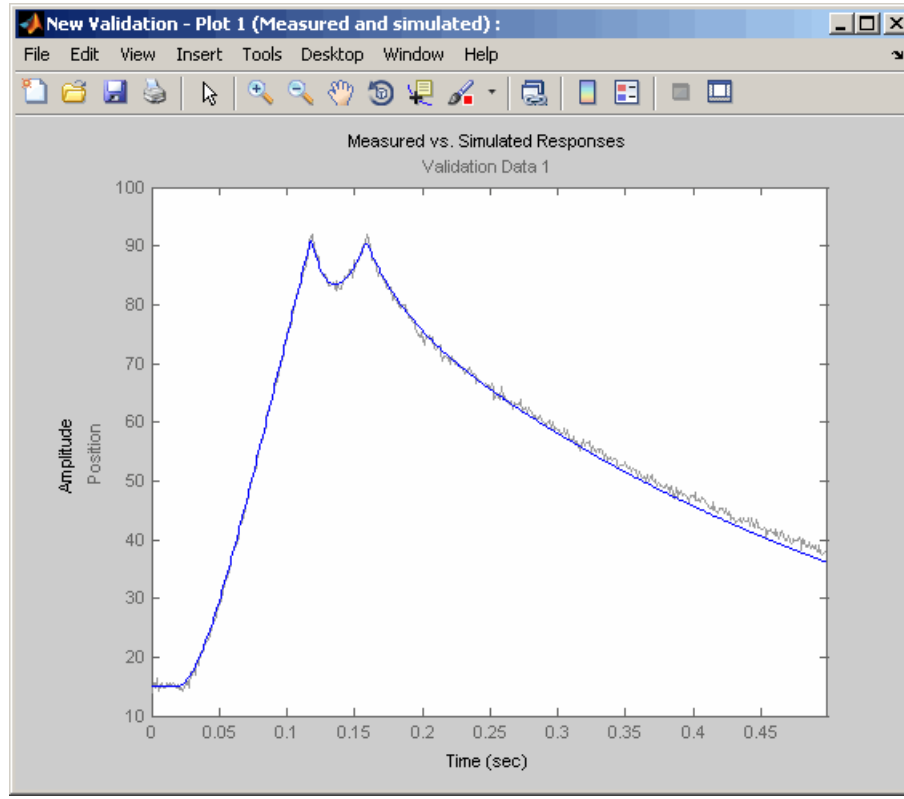
f Click **Show Plots**.

This action opens the following figures:

- `New Validation - Plot 1 (Measured and simulated)`: Displays the validation data and simulated response.
- `New Validation - Plot 2(Residuals)`: Displays the difference between the measured data and simulated response.

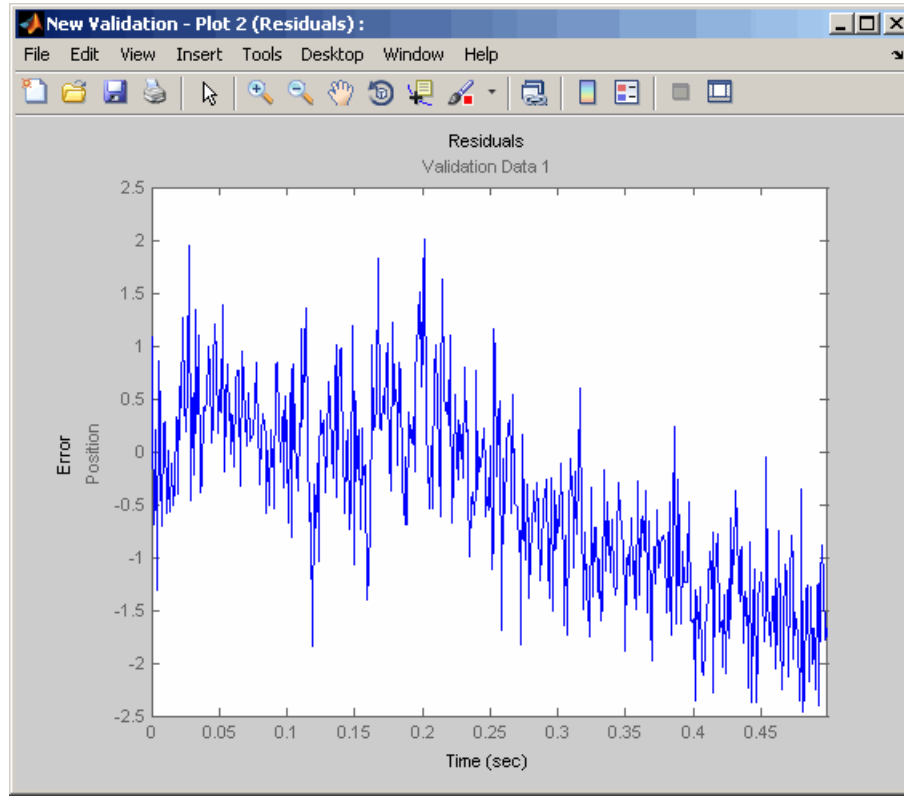
3 Examine the simulated response plot to see how well the simulated output matches the validation data.

The simulated response, as shown in blue on the `New Validation - Plot 1 (Measured and simulated)` plot, is overlaid on the measured output data, and closely matches the measured data `Validation Data 1`.



4 Examine the residuals plot to compare the difference between the simulated response and measured data.

The difference between the simulated and measured data, as shown in the `New Validation - Plot 2(Residuals)` plot, varies between 2 and -2.5. The residuals lie within 6% of the maximum output variation, and do not display any systematic patterns. This indicates a good fit between the simulated output and measured data.



5 Save the Control and Estimation Tools Manager project.

a In the Control and Estimation Tools Manager GUI, select **File > Save**.

This action opens the Save Projects dialog box.

b In the Save Projects dialog box, click **OK**.

c In the Save Projects window, specify the name of the project in the **File name** field, and click **Save**.

This action saves the project as a MAT-file.

Tip You can load a project by selecting **File > Load** in the Control and Estimation Tools Manager GUI. To view the estimated parameter values, select the **Parameters** tab of **New Estimation** node.

7

Tutorial — Optimizing Parameters to Meet Time-Domain Requirements Using the GUI

- "About This Tutorial" on page 7-2
- "Configuring a Model for Optimizing Parameters" on page 7-5
- "Optimizing Model Parameters to Meet Step Response Requirements" on page 7-8
- "Refining Model Parameters to Track a Reference Signal" on page 7-25
- "Saving the Project" on page 7-31

About This Tutorial

In this section...

"Objectives" on page 7-2

"About the Model" on page 7-2

"Design Requirements" on page 7-4

Objectives

In this tutorial, you learn how to use the GUI to optimize the parameters of a Simulink model to meet time-domain design requirements.

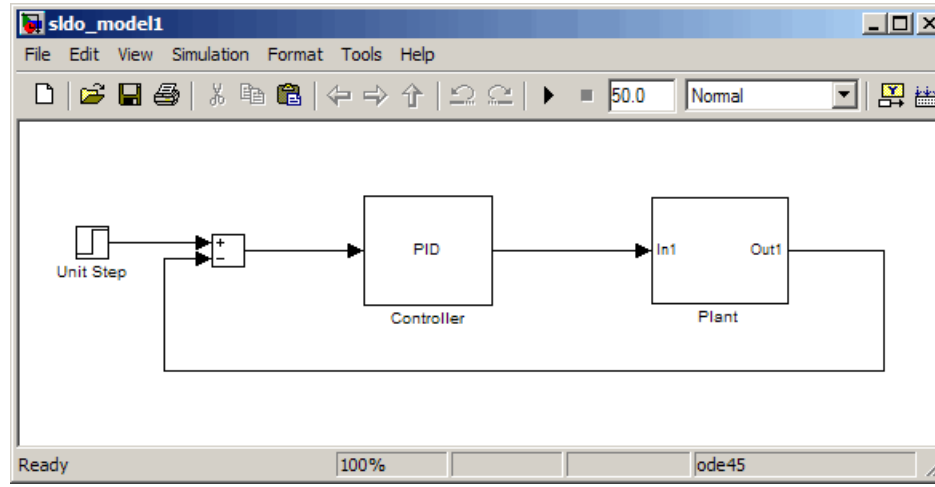
You accomplish the following tasks using the GUI:

- Specify step response requirements on the model's output.
- Specify a reference signal for the model's output to track.
- Optimize and refine the parameters to meet the design requirements.

Tip To learn how to optimize model parameters to meet time-domain design requirements using command line functions, see Chapter 8, "Tutorial — Optimizing Parameters to Meet Time-Domain Requirements Using the Command Line".

About the Model

In this tutorial, you use the Simulink model named `sldo_model1`, as shown in the next figure.

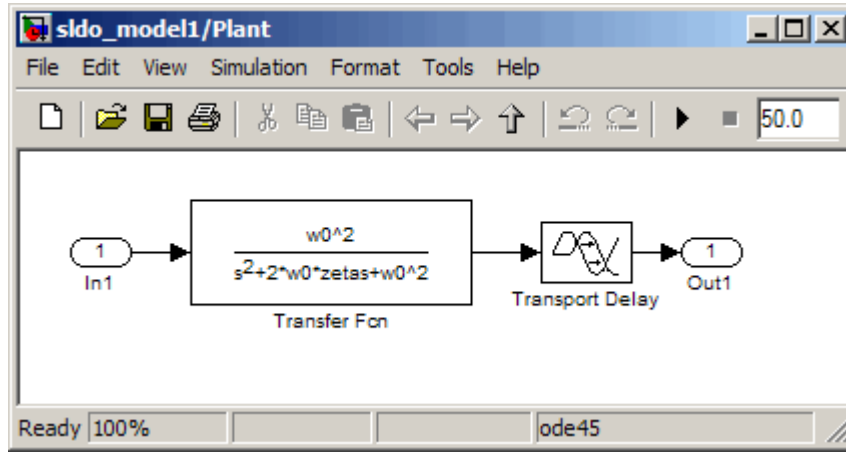


Tip To open the model, type `sldo_model1` at the MATLAB command prompt.

The model contains a `Controller` block, which is a PID controller. This block controls the output of the `Plant` subsystem. The `Unit Step` block applies a step input to the system.

Note This tutorial uses a step input to produce the model's response and optimize the model parameters to meet step response requirements. You can also use other types of inputs, such as ramp, and optimize parameters to meet requirements on the model's response to these inputs.

Double-click the `Plant` subsystem to open it. The plant is a second-order system with delay. It contains Transfer Function and Transport Delay blocks, as shown in the next figure.



To learn more about the blocks, see Transfer Fcn and Transport Delay block reference pages.

Design Requirements

The model's output must meet the following design requirements:

- Overshoot less than 10%
- Rise time less than 10 seconds
- Settling time less than 30 seconds
- Track a reference signal specified by 1–e(-0.3*t), where t is time

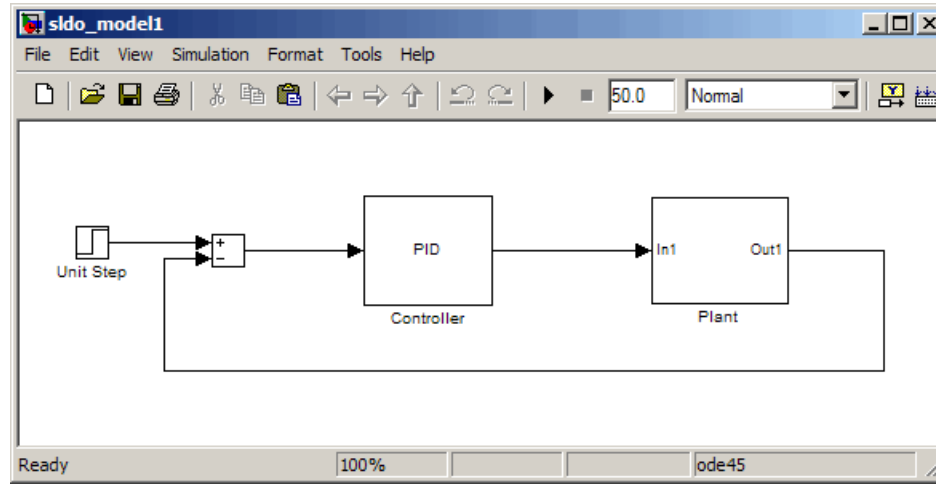
Configuring a Model for Optimizing Parameters

In this portion of the tutorial, you configure a Simulink model for optimizing parameters:

1. Open the `sldo_model1` model, if it is not already open, by typing the model name at the MATLAB prompt:

   ```
   sldo_model1
   ```

   The Simulink model opens, as shown in the next figure.

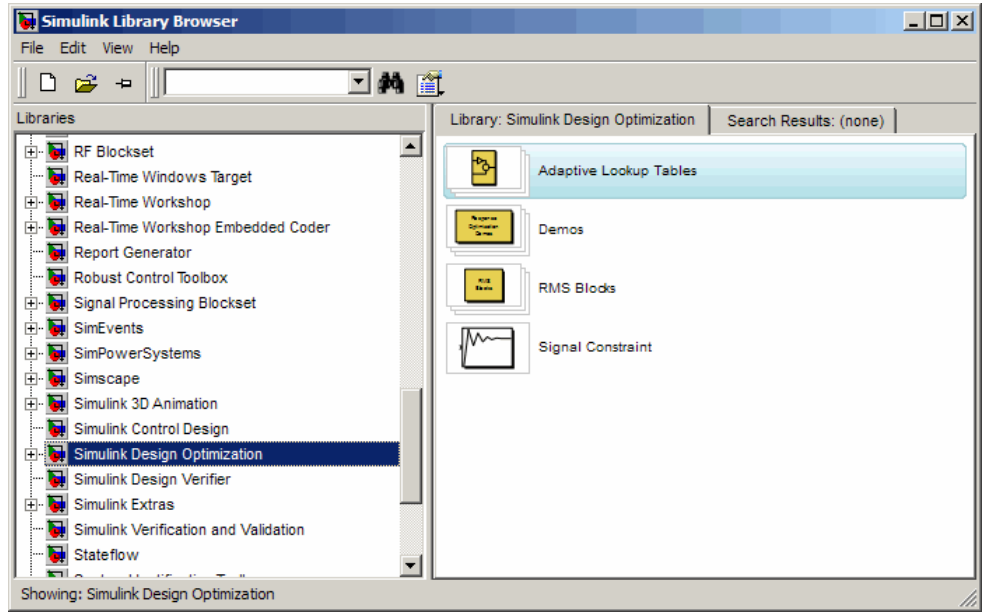


   To learn more about the model, see “About the Model” on page 7-2.

2. Add a Signal Constraint block to the model.

   a. In the Simulink model, select **View > Library Browser** to open the Simulink Library Browser.

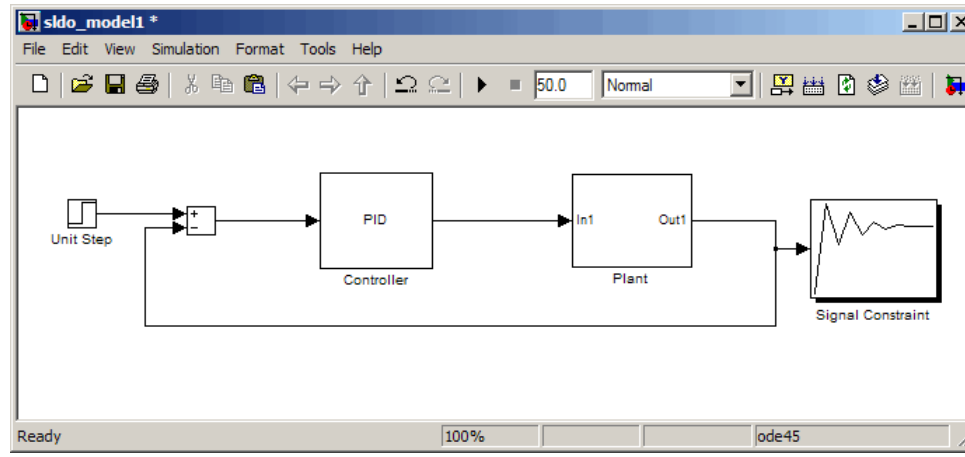
b In the **Libraries** pane, select **Simulink Design Optimization**.



c Drag and drop the Signal Constraint block into the Simulink model window.

To learn more about the block, see the Signal Constraint block reference page.

d Connect the Signal Constraint block to the model's output, as shown in the next figure.



Note You must connect the Signal Constraint block to the signal to which you want to add design requirements. To learn more, see "Constraining Model Signals" in the *Simulink Design Optimization User's Guide*.

Optimizing Model Parameters to Meet Step Response Requirements

In this section...

"Specify Time-Domain Design Requirements" on page 7-8

"Specifying Parameters to Optimize" on page 7-18

"Optimizing the Parameters" on page 7-21

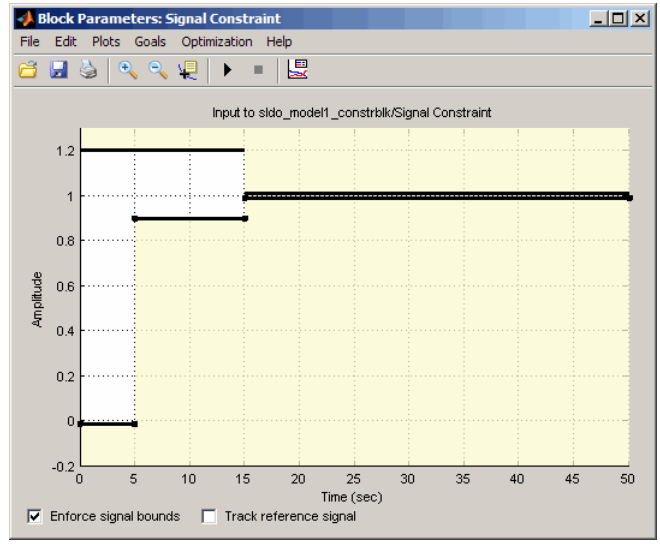
Specify Time-Domain Design Requirements

In this portion of the tutorial, you specify the rise time, setting time and overshoot requirements that the model's response should satisfy.

You must have already configured the Simulink model, as described in "Configuring a Model for Optimizing Parameters" on page 7-5.

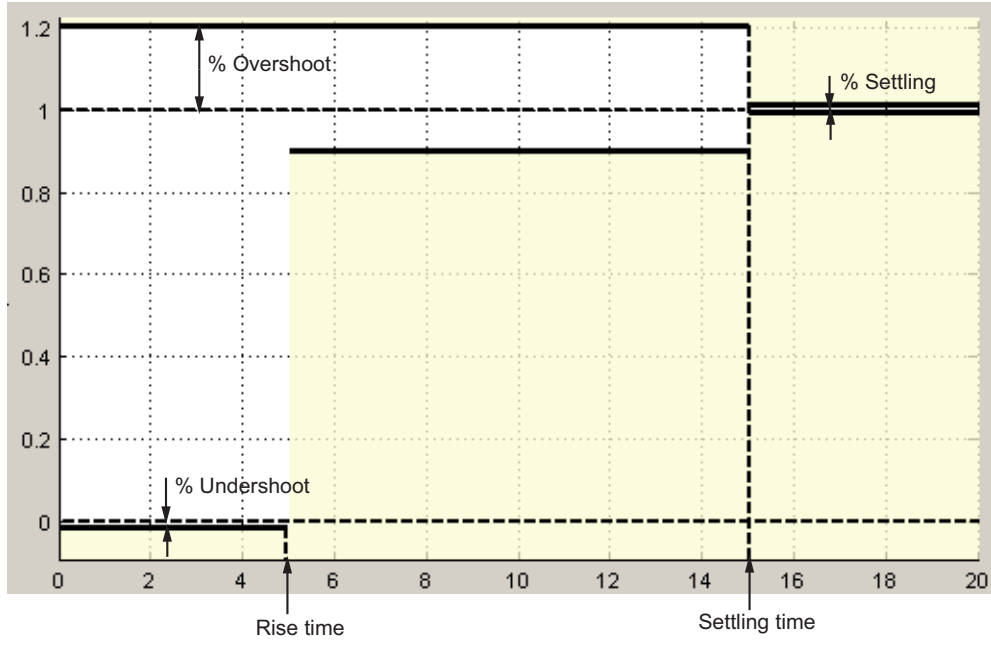
To specify the design requirements:

1 Double-click the Signal Constraint block to open the Block Parameters: Signal Constraint window.



Tip To view the line segment that extends from 0 to 15 seconds, set the upper limit of the **Amplitude** axis to 1.3. To do so, right-click in the plot, and select **Axes Limits**.

The Block Parameters window shows an amplitude versus time plot. The line segments map to the step response requirements shown in the next figure.



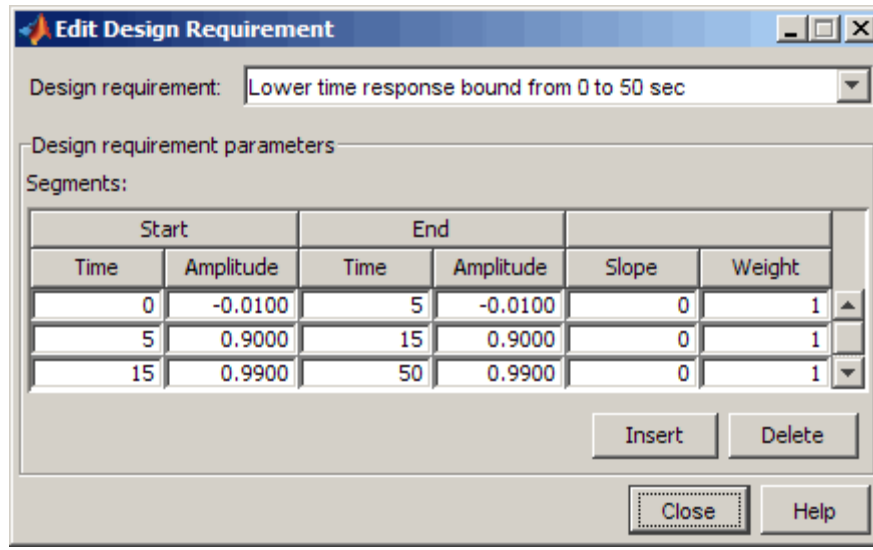
By default, the line segments represent the following step response requirements:

- Amplitude less than or equal to –0.01 up to the rise time of 5 seconds for 1% undershoot
- Amplitude between 0.9 and 1.2 up to the settling time of 15 seconds
- Amplitude equal to 1.2 for 20% overshoot up to the settling time of 15 seconds
- Amplitude between 0.99 and 1.01 beyond the settling time for 2% settling

To specify the design requirements described in “Design Requirements” on page 7-4, you must modify the time and amplitude values of the line segments, as described in the next steps.

2 Double-click the lower yellow region shown on the plot.

This action opens the Edit Design Requirement dialog box, as shown in the following figure.

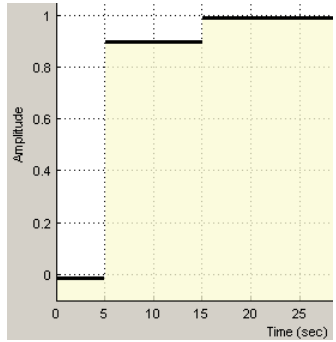


The rows in the dialog box define the *lower bound* on the signal. The time span, displayed in the **Design requirement** drop-down list, is same as the simulation time specified in the Simulink model.

The rows correspond to the following three line segments in the Block Parameters window:

- First row — Lower-line segment that extends from 0 to 5 seconds.
- Second row — Lower-line segment that extends from 5 to 15 seconds.
- Third row — Lower-line segment that extends from 15 to 50 seconds.

The default amplitude and time values specified in the rows appear in the Block Parameters window, as shown in the next figure.



3 Specify the rise time requirement:

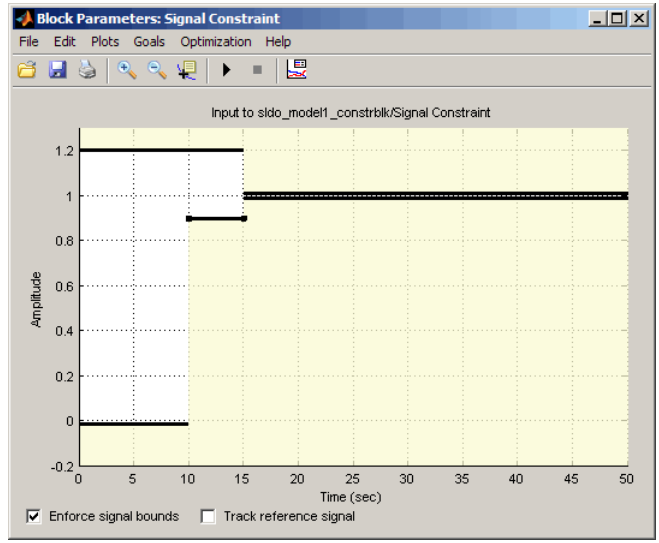
a In the first row of the Edit Design Requirement dialog box, double-click the **Time** field in the **End** column.

b Type 10, and press **Enter**.

The first row now looks like the following figure.

Start		End			
Time	Amplitude	Time	Amplitude	Slope	Weight
0	-0.0100	10	-0.0100	0	1

This action also updates the Block Parameters window to show the rise time requirement. This requirement is represented by the line segment at the bottom left of the window that extends from 0 to 10 seconds, as shown in the next figure.



4 Specify the settling time requirement:

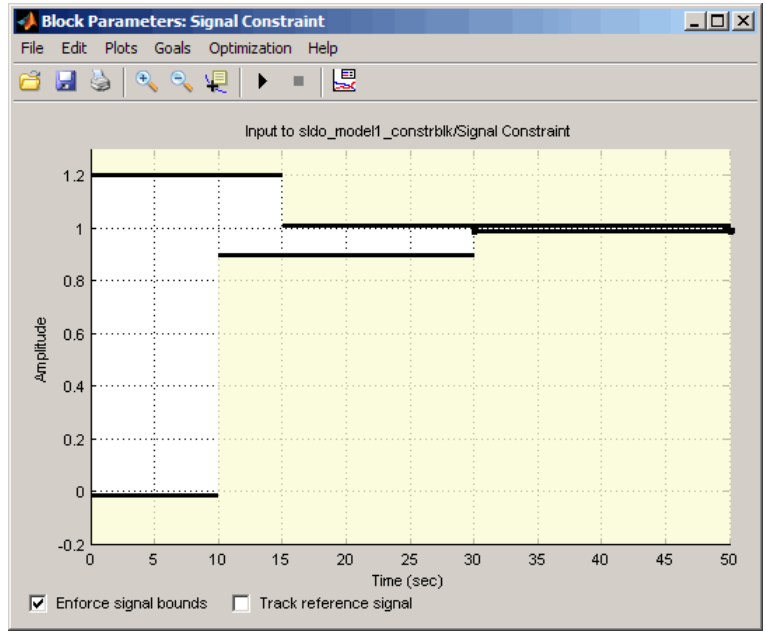
a In the second row of the Edit Design Requirement dialog box, double-click the **Time** field in the **End** column.

b Type 30, and press **Enter**.

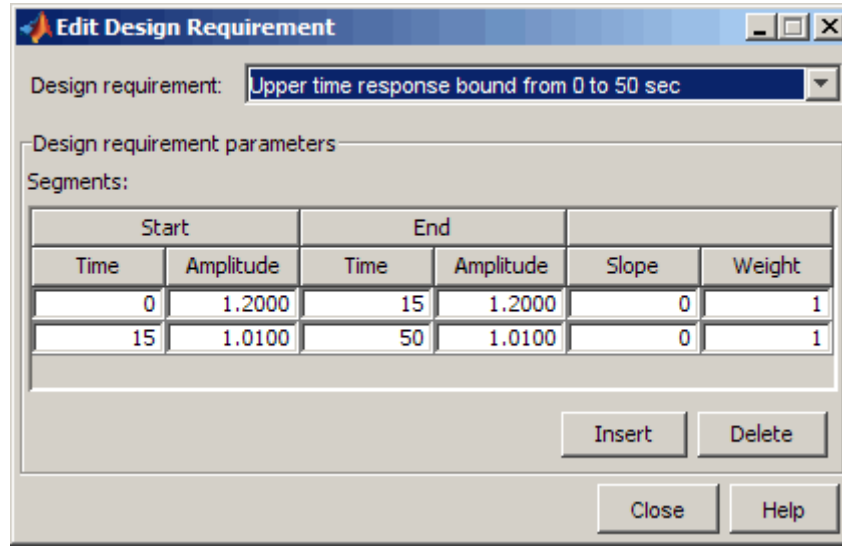
The second row looks like the following figure.

Start		End			
Time	Amplitude	Time	Amplitude	Slope	Weight
0	-0.0100	10	-0.0100	0	1
10	0.9000	30	0.9000	0	1

This action also updates the Block Parameters window to show the settling time requirement on the lower bound. This requirement is represented by the line segment that extends from 10 to 30 seconds, as shown in the next figure.



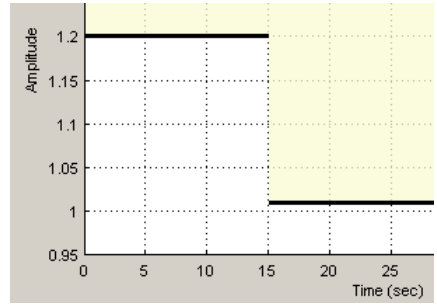
c In the Edit Design Requirements dialog box, select `Upper time response bound from 0 to 50 sec` in the **Design Requirement** drop-down list.



The rows in the dialog box define the *upper bound* on the signal, and correspond to the following two line segments in the Block Parameters window:

- First row — Upper-line segment that extends from 0 to 15 seconds.
- Second row — Upper-line segment that extends from 15 to 50 seconds.

The default amplitude and time values specified in the rows appear in the Block Parameters window, as shown in the next figure.



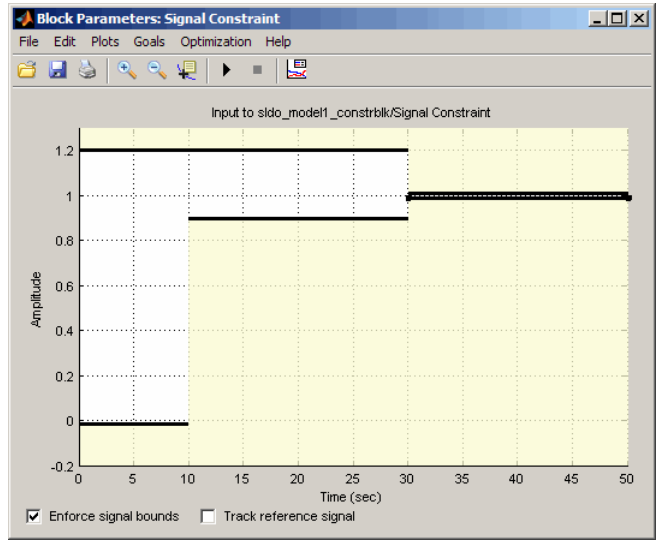
d In the first row, double-click the **Time** field in the **End** column.

e Type `30`, and press **Enter**.

The first row looks like the following figure.

Start		End			
Time	Amplitude	Time	Amplitude	Slope	Weight
0	1.2000	30	1.2000	0	1

This action also updates the Block Parameters window to show the settling time requirement on the upper-bound. This requirement is represented by the line segment that extends from 0 to 30 seconds, as shown in the next figure.



5 Specify the overshoot requirement:

a In the Edit Design Requirement dialog box, verify that `Upper time response bound from 0 to 50 sec` is selected in the **Design Requirement** drop-down list.

b In the first row, double-click the **Amplitude** field in the **Start** column.

c Type `1.1`, and press **Enter**.

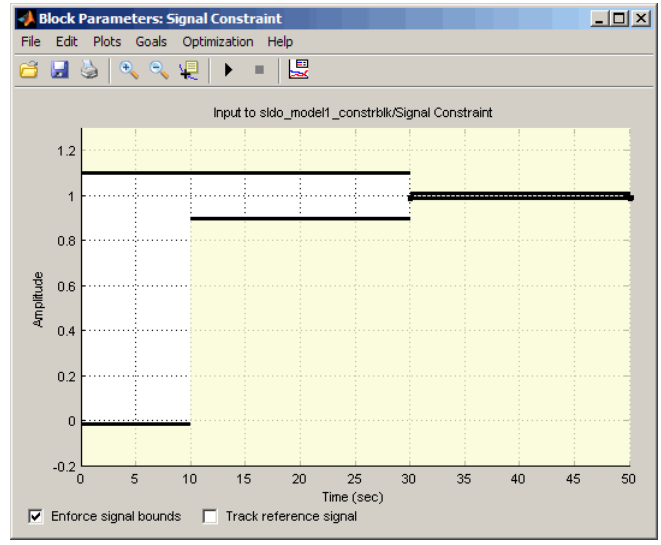
d Double-click the **Amplitude** field of the **End** column.

e Type `1.1`, and press **Enter**.

The first row now looks like the following figure.

Start		End			
Time	Amplitude	Time	Amplitude	Slope	Weight
0	1.1000	30	1.1	0	1

The Block Parameters window updates to show the overshoot requirement. This requirement is represented by the amplitude value of `1.1` for the line segment that extends from 0 to 30 seconds, as shown in the following figure.



When you optimize the model parameters, the model's final response must lie in the white region bounded by the line segments in order to meet the design requirements.

6 In Edit Design Requirement dialog box, click **Close**.

Specifying Parameters to Optimize

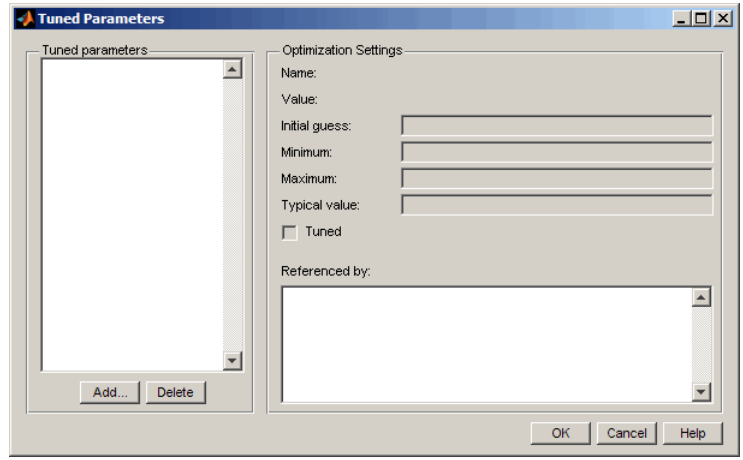
In this portion of the tutorial, you specify the model parameters to optimize. When you optimize the model parameters, the software modifies the parameter values to meet the specified design requirements.

Before you specify the parameters to optimize, you can specify the design requirements, as described in "Specify Time-Domain Design Requirements" on page 7-8.

To specify the parameters to optimize:

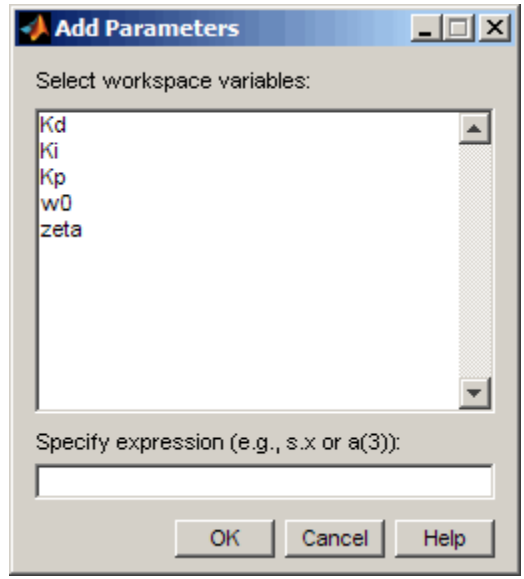
1. In the Block Parameters window, select **Optimization > Tuned Parameters**.

   This action opens the Tuned Parameters dialog box.



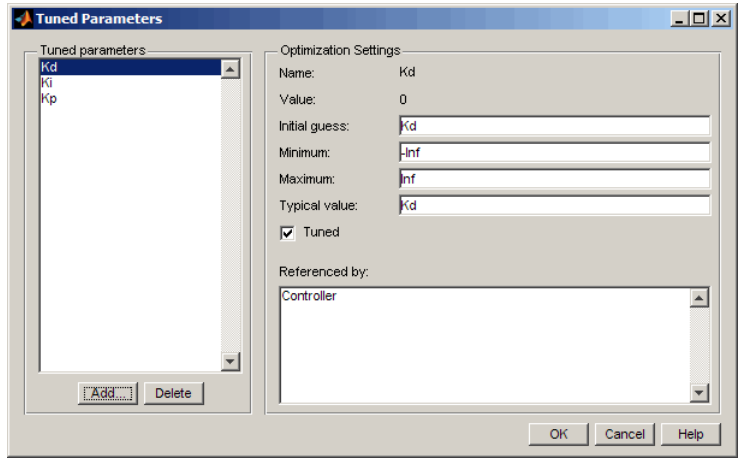
2 In the Tuned Parameters window, click **Add**.

This action opens the Add Parameters dialog box.



3 In the Add Parameters dialog box, select the PID controller parameters `Kd`, `Ki` and `Kp`, and click **OK**.

This action adds `Kd`, `Ki` and `Kp` to the Tuned Parameters window.



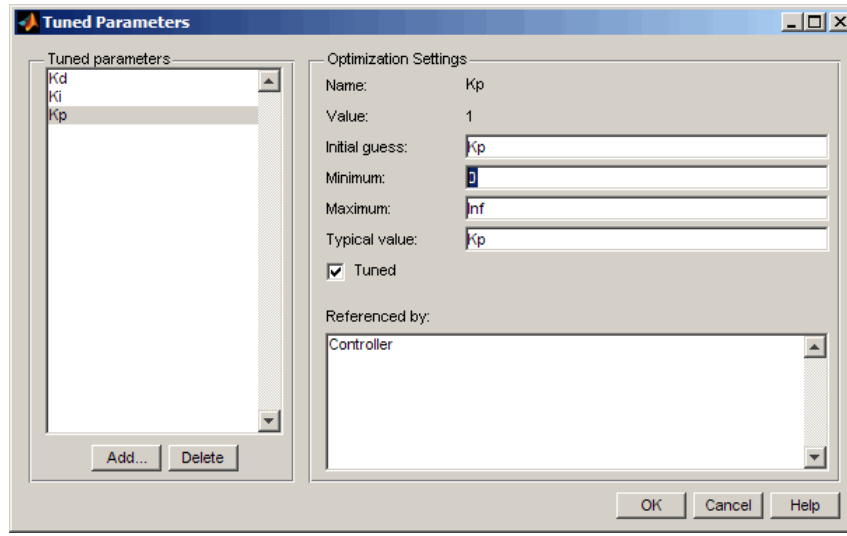
The **Optimization Settings** area displays the following parameter settings:

- **Value** — Current parameter value
- **Initial Guess** — Initial parameter value
- **Minimum** and **Maximum** — Parameter bounds
- **Typical Value** — Scaling factor for the parameter

To learn more about the parameter settings, see "Changing Tuned Parameter Specifications" in the *Simulink Design Optimization User's Guide*.

4 To limit the parameters to positive values, enter the minimum value of each parameter as `0` in the corresponding **Minimum** field.

After you enter these values, the Tuned Parameters window resembles the next figure.



5 Click **OK** to apply the parameter settings and close the Tuned Parameters window.

Optimizing the Parameters

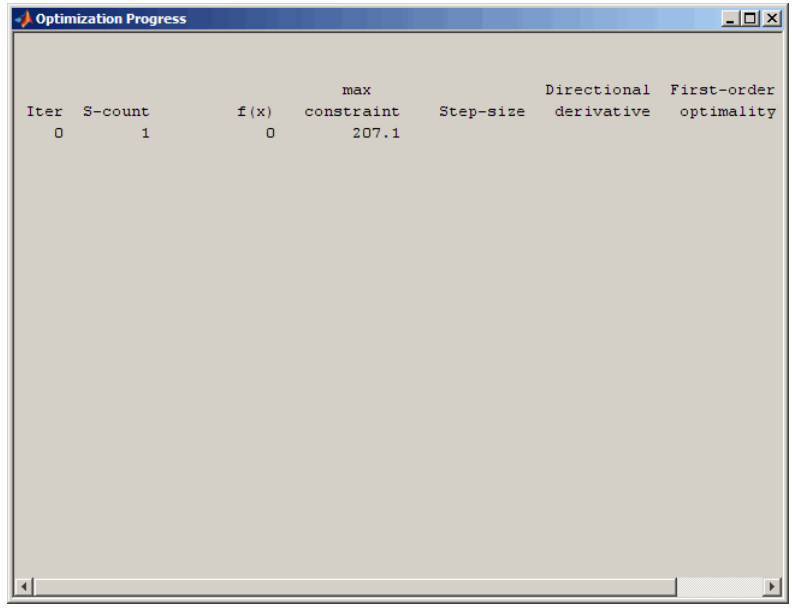
In this portion of the tutorial, you optimize the model parameters to meet the specified design requirements.

You must have already specified the design requirements and the parameters to optimize, as described in "Specify Time-Domain Design Requirements" on page 7-8, and "Specifying Parameters to Optimize" on page 7-18, respectively.

To optimize the model parameters:

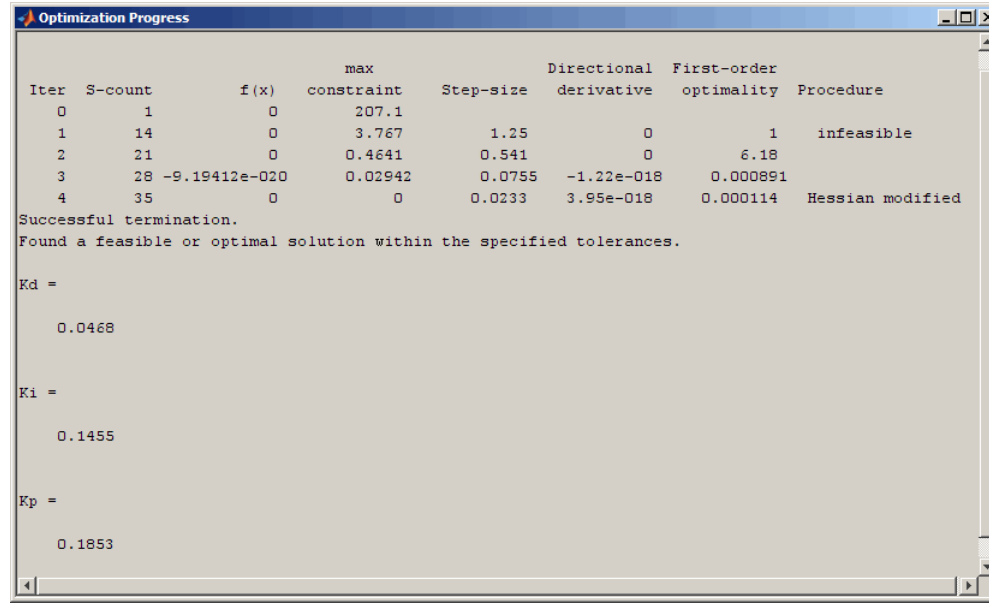
1 In the Block Parameters window, click **Optimization > Start** to start the optimization.

This action opens the optimization progress window as shown in the next figure.



- During each optimization iteration, the software simulates the model, and the default optimization method `Gradient descent` modifies the controller parameters to reduce the distance between the simulated response and the design requirement line segments. To learn more about the optimization method, see "Selecting Optimization Methods" in the *Simulink Design Optimization User's Guide*.

- After the optimization completes, the Optimization Progress window resembles the next figure.

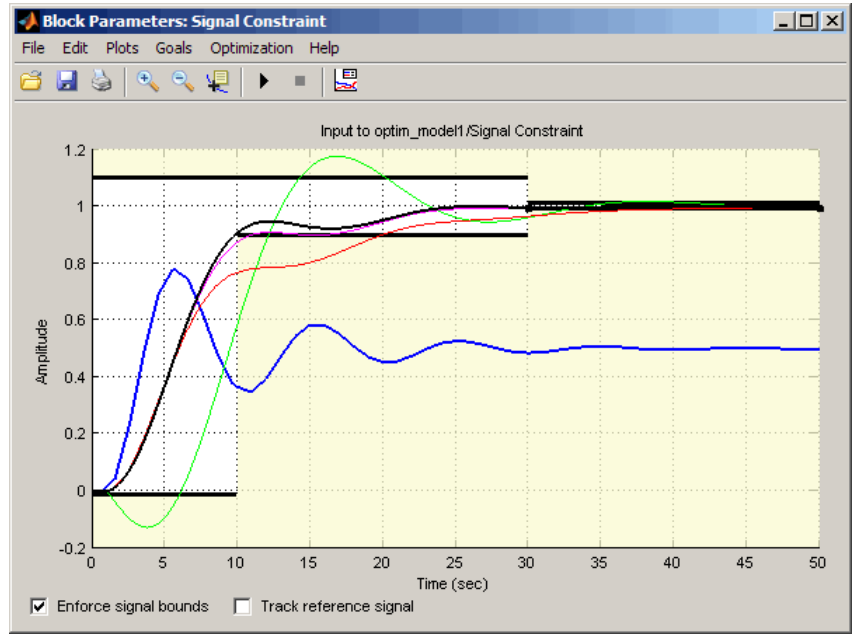


  The message `Successful termination` indicates that the optimization method found a solution that meets the design requirements within the parameter bounds. For more information about the outputs displayed in the optimization progress window, see "Displaying Iterative Output" in the Optimization Toolbox documentation.

  The optimized parameter values are displayed in the Optimization Progress window, and also written into the Simulink model.

2 In the Block Parameters window, examine the model's response to see if the response meets the step response requirements.

  The final response of the system with the optimized parameter values appears in black, as shown in the next figure.



The plot also displays the initial response of the system in blue. The remaining responses show the simulated response at each optimization iteration.

The optimized response of the system lies in the white region bounded by the design requirement line segments and thus meets the specified requirements.

Next, you refine the model parameters by specifying a reference signal that the system response must track, as described in “Refining Model Parameters to Track a Reference Signal” on page 7-25.

Refining Model Parameters to Track a Reference Signal

In this portion of the tutorial, you optimize the model parameters to track a reference signal specified by the equation

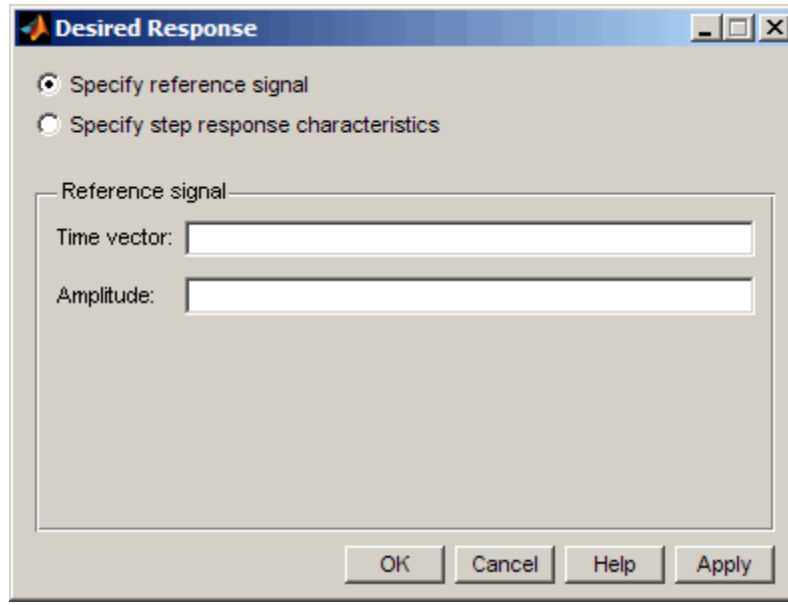
$$y = 1 - \exp(-0.3 \times t)$$

You must have already optimized the model parameters to meet the step response requirements, as described in "Optimizing Model Parameters to Meet Step Response Requirements" on page 7-8.

To optimize the parameters to track a reference signal:

1. In the Block Parameters window, select **Goals > Desired Response**.

   This action opens the Desired Response dialog box.

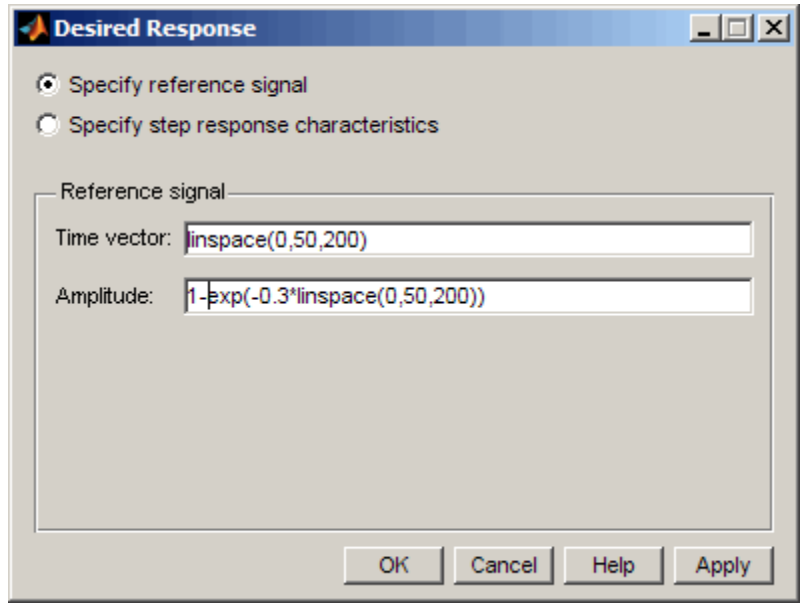


2. Specify the reference signal:

a In the **Time vector** field of the Desired Response dialog box, enter `linspace(0,50,200)`.

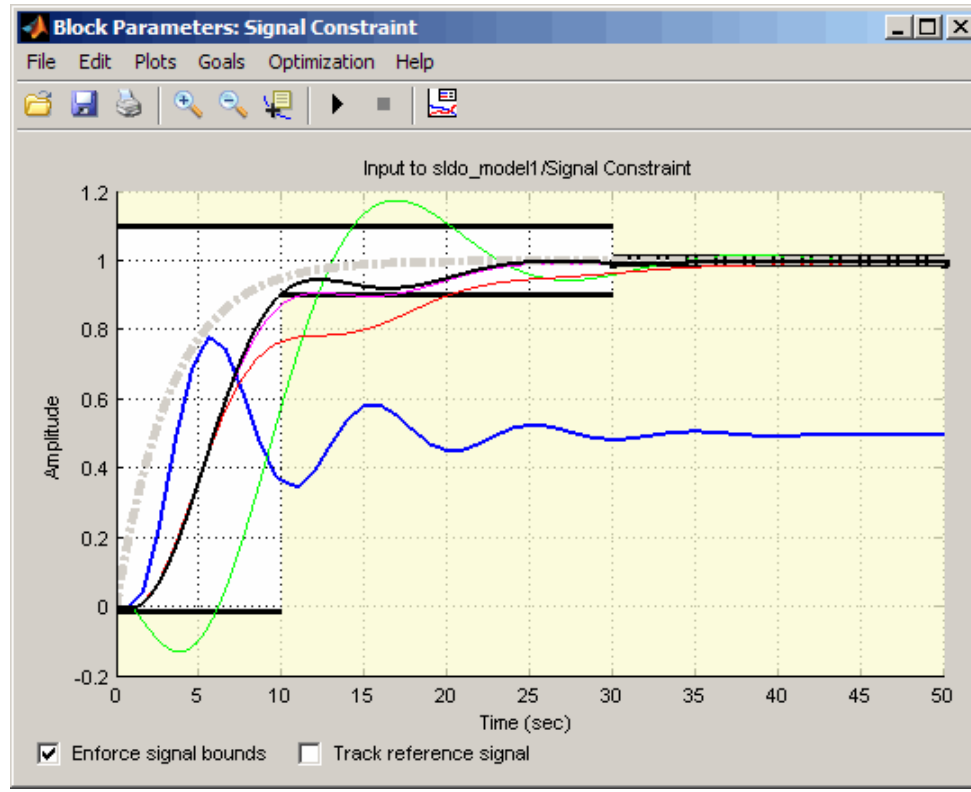
b In the **Amplitude** field, enter `1-exp(-0.3*linspace(0,50,200))`.

After you make these changes, the Desired Response dialog box resembles the following figure.



c Click **OK** to add the reference signal to the block and close the Desired Response dialog box.

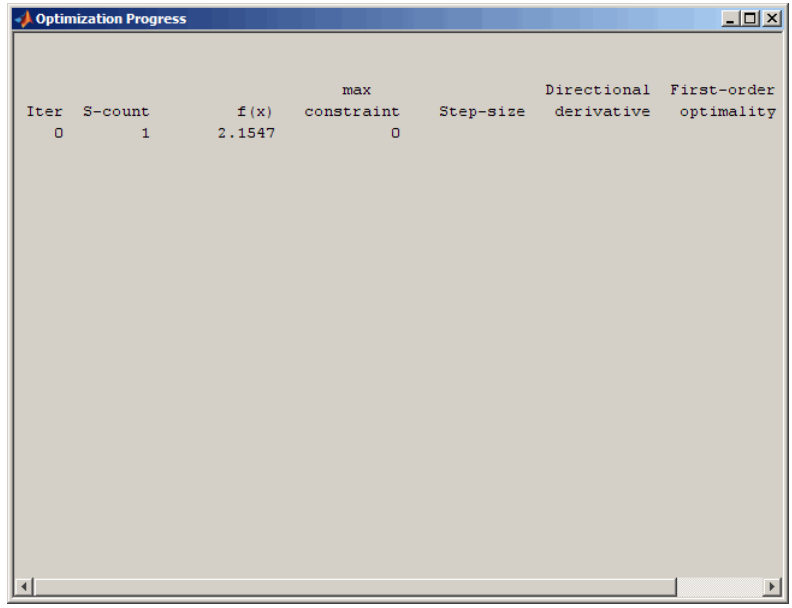
The Block Parameters window updates to display the reference signal, shown as the gray dashed curve, in the next figure.



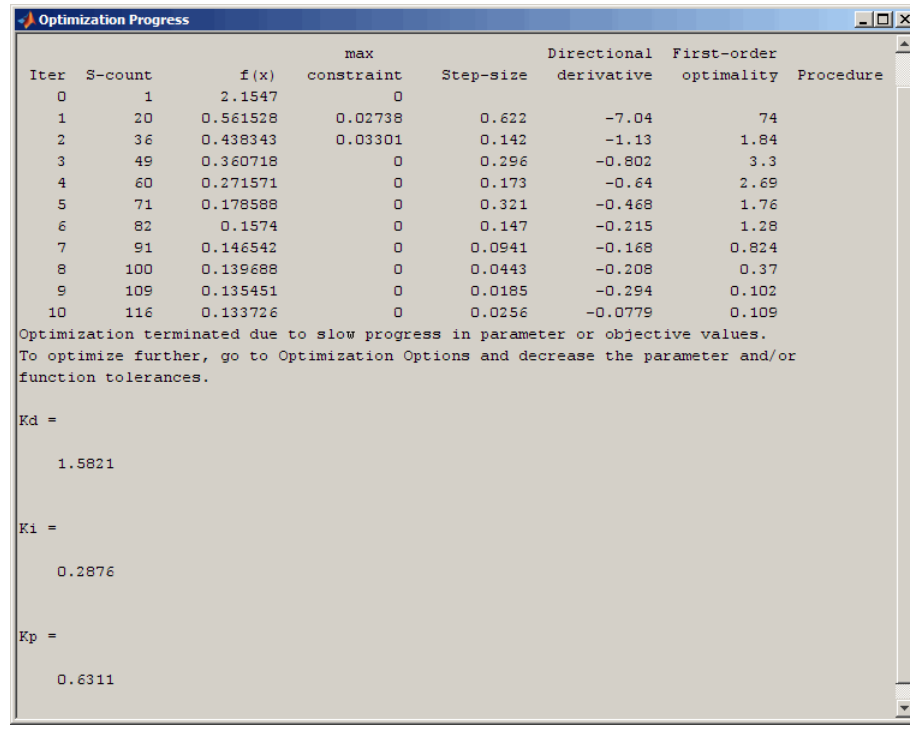
3 In the Block Parameters window, select the **Track reference signal** check box.

4 In the Signal Constraint block window, select **Optimization > Start** to start the optimization.

This action restarts the optimization using the optimized parameters values computed in "Optimizing Model Parameters to Meet Step Response Requirements" on page 7-8. The Optimization Progress window opens, as shown in the next figure.



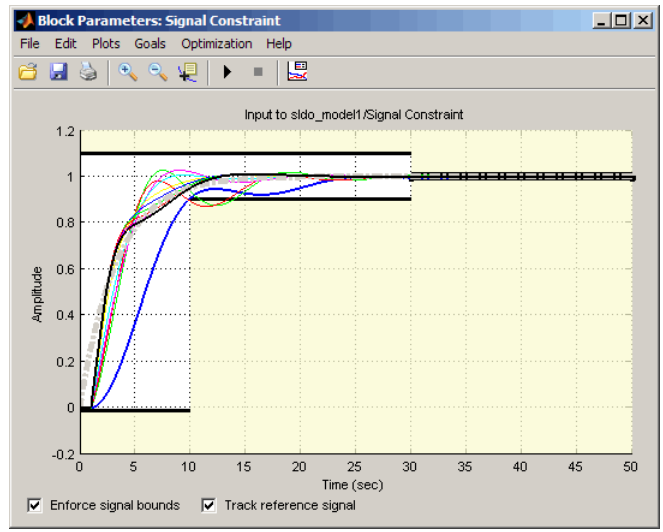
- At each iteration, the optimization method modifies the controller parameters to minimize the error between the simulated response and the reference signal. Simultaneously, the method satisfies the step response requirements by reducing the distance between the simulated response and the design requirement line segments.
- After the optimization completes, the Optimization Progress window resembles the following figure.



The message, `Optimization terminated due to slow progress in parameter or objective values` appears in the Optimization Progress window. This message indicates that the optimization method cannot minimize the error between the simulated response and the reference signal any further. You must examine the optimized response of the model to check if the response tracks the reference signal, as described in the next step.

5 In the Block Parameters window, examine the model's response to see how well the response tracks the reference signal.

The model's response before you began the optimization appears as the blue curve. After the optimization completes, the final response, shown as the black curve, tracks the reference signal.



Note To refine the parameters further to track the reference signal more closely, change the parameter and function tolerances, as described in "Selecting Optimization Termination Options" in the *Simulink Design Optimization User's Guide*.

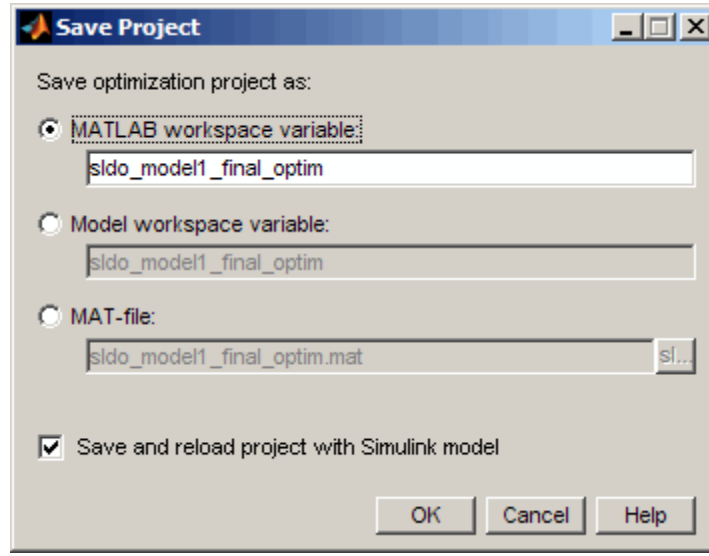
Saving the Project

After you optimize the model parameters, you can save the model with the design requirements and optimized parameter values in a response optimization project.

To save a response optimization project:

1. In the Signal Constraint block window, select **File > Save As**.

   This action opens the Save Project dialog box.



2. In the Save Project dialog box, select the **Model workspace variable** option, and enter a variable name.

3. Click **OK** to save the project.

Tip You can load the response optimization project by loading the Simulink model.

8

Tutorial — Optimizing Parameters to Meet Time-Domain Requirements Using the Command Line

- "About This Tutorial" on page 8-2
- "Configuring a Model for Optimizing Parameters" on page 8-5
- "Optimizing Model Parameters to Meet Step Response Requirements" on page 8-8
- "Refining Model Parameters to Track a Reference Signal" on page 8-15

About This Tutorial

In this section...

"Objectives" on page 8-2

"About the Model" on page 8-2

"Design Requirements" on page 8-4

Objectives

In this tutorial, you learn how to use Simulink Design Optimization functions to optimize the parameters of a Simulink model to meet time-domain design requirements.

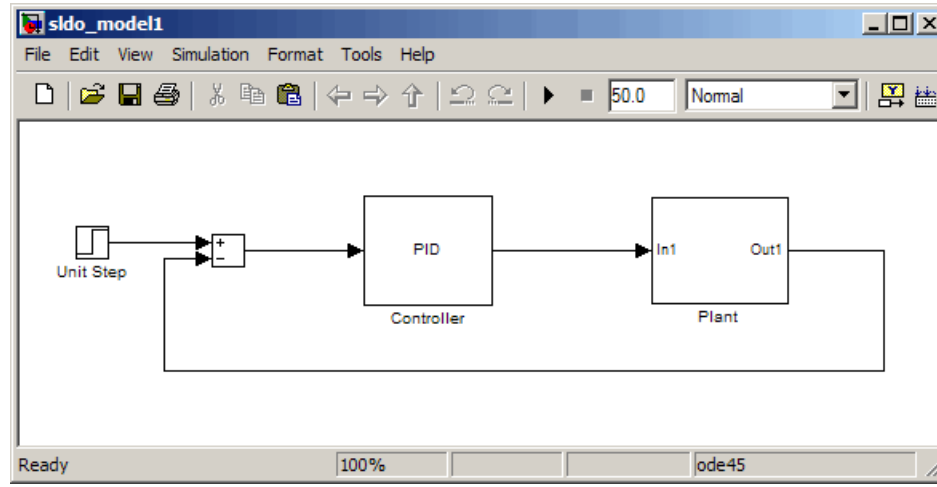
You accomplish the following tasks using the command line:

- Specify step response requirements on the model's output.
- Specify a reference signal for the model's output to track.
- Optimize the parameters to meet the design requirements.

Tip To learn how to optimize model parameters to meet time-domain design requirements using the GUI, see Chapter 7, "Tutorial — Optimizing Parameters to Meet Time-Domain Requirements Using the GUI".

About the Model

In this tutorial, you use the Simulink model named `sldo_model1`, as shown in the next figure.

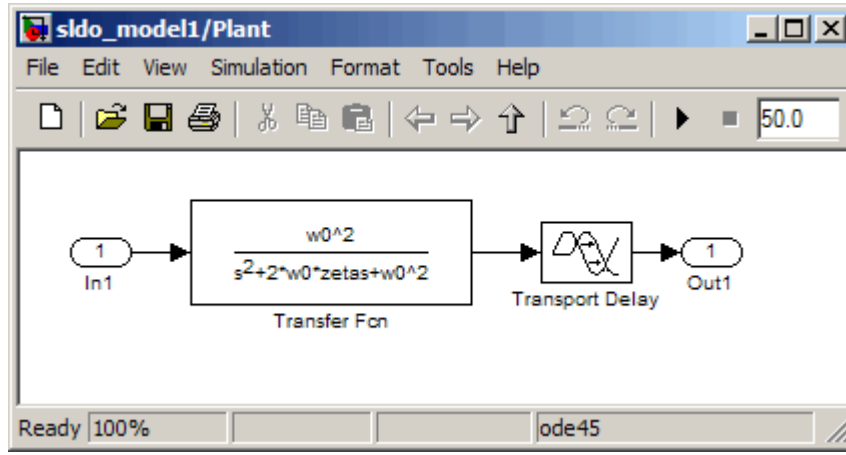


Tip To open the model, type `sldo_model1` at the MATLAB command prompt.

The model contains a `Controller` block, which is a PID controller. This block controls the output of the `Plant` subsystem. The `Unit Step` block applies a step input to the system.

Note This tutorial uses a step input to produce the model's response and optimize the model parameters to meet step response requirements. You can also use other types of inputs, such as ramp, and optimize parameters to meet requirements on the model's response to these inputs.

Double-click the `Plant` subsystem to open it. The plant is modeled as a second-order system with delay. It contains Transfer Function and Transport Delay blocks, as shown in the next figure.



To learn more about the blocks, see Transfer Fcn, and Transport Delay block reference pages.

Design Requirements

The models' output must meet the following design requirements:

- Overshoot less than 10%
- Rise time less than 10 seconds
- Settling time less than 30 seconds
- Track a reference signal specified by $y = 1 - \exp(-0.3 \times t)$, where t is time

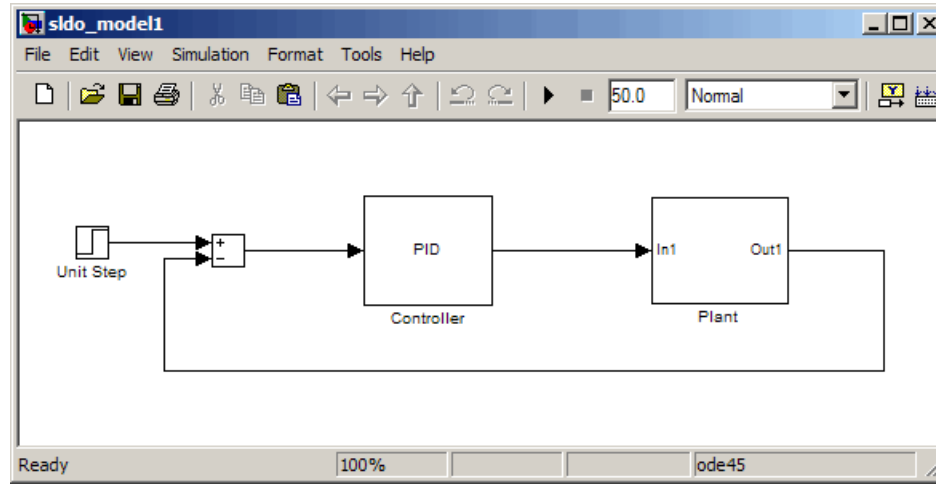
Configuring a Model for Optimizing Parameters

To optimize parameters of a Simulink model, you must first configure the model:

1. Open the `sldo_model1` model, if it is not already open, by typing the model name at the MATLAB prompt:

   ```
   sldo_model1
   ```

   The Simulink model opens, as shown in the next figure.

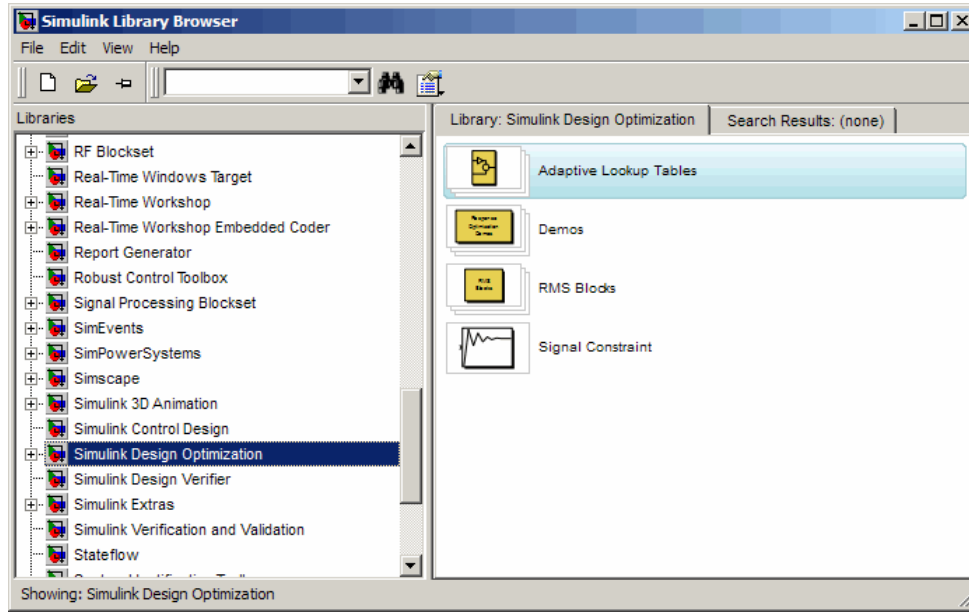


   To learn more about the model, see “About the Model” on page 8-2.

2. Add a Signal Constraint block to the model.

   a. In the Simulink model, select **View > Library Browser** to open the Simulink Library Browser.

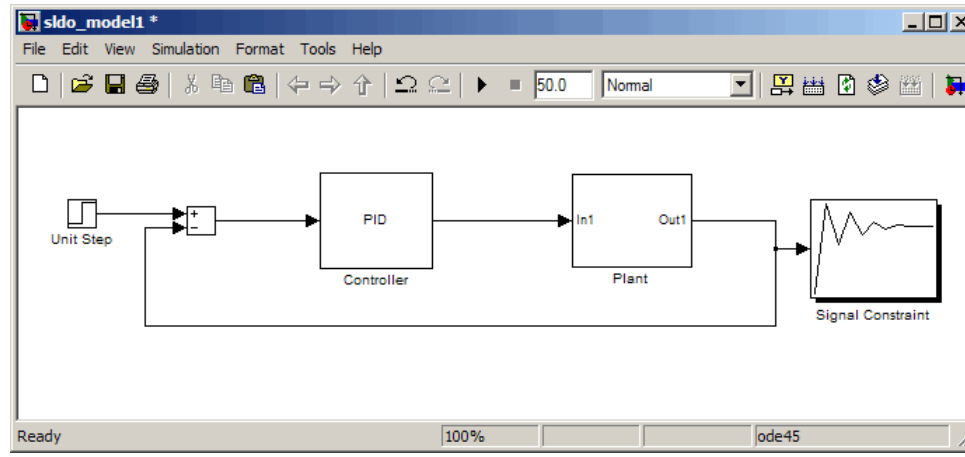
b In the **Libraries** pane, select **Simulink Design Optimization**.



c Drag and drop the Signal Constraint block into the Simulink model window.

To learn more about the block, see Signal Constraint block reference page.

d Connect the Signal Constraint block to the model's output, as shown in the next figure.



Note You must connect the Signal Constraint block to the signal to which you want to add design requirements. To learn more, see "Constraining Model Signals" in the *Simulink Design Optimization User's Guide*.

Optimizing Model Parameters to Meet Step Response Requirements

In this portion of the tutorial, you use Simulink Design Optimization commands to optimize the model parameters to meet step response requirements.

You must have already configured your model for optimization, as described in "Configuring a Model for Optimizing Parameters" on page 8-5.

To optimize the model parameters to meet step response requirements:

1. Create a new project using the `newsro` command.

   ```
   proj=newsro('sldo_model1',{'Kd','Ki','Kp'})
   ```

   This command creates a new project object, `proj`, that contains instructions to optimize the parameters `Kp`, `Ki`, and `Kd`.

   ```
                Name: 'sldo_model1'
          Parameters: [3x1 ResponseOptimizer.Parameter]
        OptimOptions: [1x1 sroengine.OptimOptions]
               Tests: [1x1 ResponseOptimizer.SimTest]
               Model: 'sldo_model1'

   Response Optimization Project.
   ```

2. Retrieve the signal constraint object for the project, `proj`, using the `findconstr` command.

   ```
   constr=findconstr(proj,'sldo_model1/Signal Constraint')
   ```

   This command retrieves the following signal constraint object:

   ```
             ConstrEnable: 'on'
               isFeasible: 1
               CostEnable: 'off'
                   Enable: 'on'
                     Name: 'Signal Constraint'
               SignalSize: [1 1]
              LowerBoundX: [3x2 double]
   ```

```
          LowerBoundY: [3x2 double]
      LowerBoundWeight: [3x1 double]
          UpperBoundX: [2x2 double]
          UpperBoundY: [2x2 double]
      UpperBoundWeight: [2x1 double]
            ReferenceX: []
            ReferenceY: []
       ReferenceWeight: []

  Signal Constraint.
```

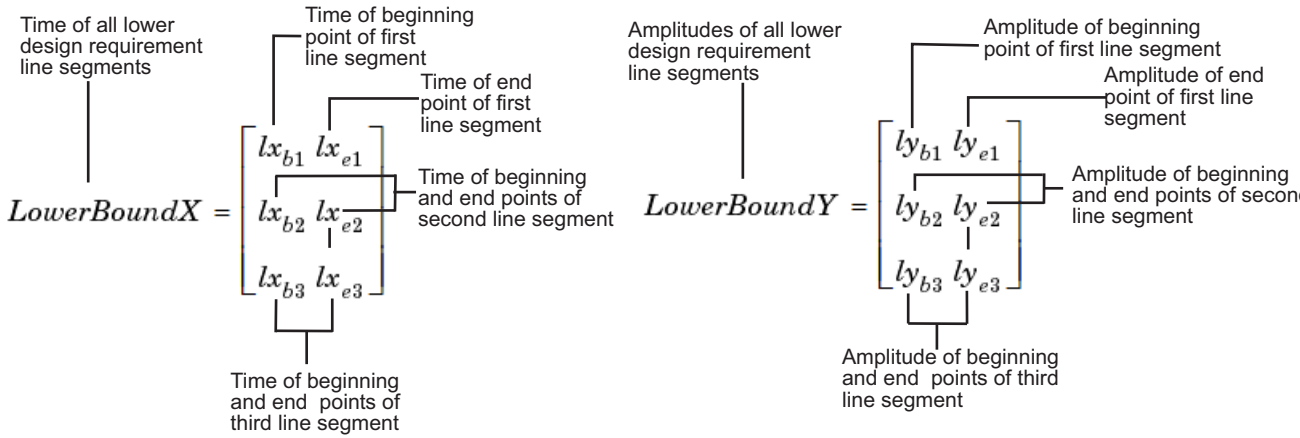
The `LowerBoundX`, `LowerBoundY`, `UpperBoundX`, and `UpperBoundY` properties of the `constr` object represent the numeric values of design requirements on a signal.

Tip You can view the default design requirements by double-clicking the Signal Constraint block in the `sldo_model1` Simulink model.

The `LowerBoundX`, and `LowerBoundY` properties define the *lower bound*, and the `UpperBoundX` and `UpperBoundY` define the *upper bound* on the signal. These properties are matrices, whose elements represent the values of the bounds as shown in the following table.

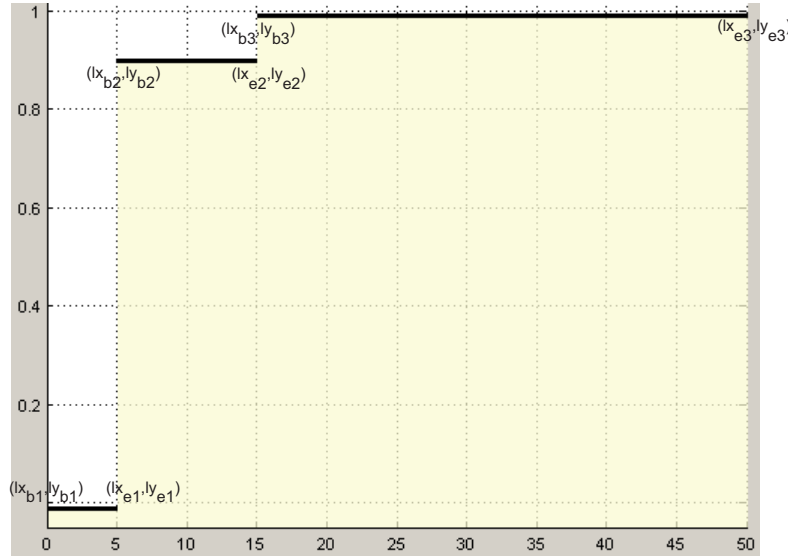
Property Name	Purpose
`LowerBoundX`	Time values of the lower-bound line segments
`LowerBoundY`	Amplitude values of the lower-bound line segments
`UpperBoundX`	Time values of the upper-bound line segments
`UpperBoundY`	Amplitude values of the upper-bound line segments

For example, the elements of the `LowerBoundX`, and `LowerBoundY` matrices map to the following values.



$$LowerBoundX = \begin{bmatrix} lx_{b1} & lx_{e1} \\ lx_{b2} & lx_{e2} \\ lx_{b3} & lx_{e3} \end{bmatrix} \qquad LowerBoundY = \begin{bmatrix} ly_{b1} & ly_{e1} \\ ly_{b2} & ly_{e2} \\ ly_{b3} & ly_{e3} \end{bmatrix}$$

The line segments that correspond to the elements of `LowerBoundX`, and `LowerBoundY` matrices appear in the next figure.



Similarly, the elements of `UpperBoundX` and `UpperBoundY` map to the time and amplitude values of the upper-bound line segments, respectively. To specify the step response requirements, as described in "Design Requirements" on page 8-4, you must edit these properties as shown in the next step.

Note When you add or modify design requirements using functions, the Block Parameters window does not update to display the new requirements.

To view the values of the `LowerBoundY` matrix, type the following command:

```
constr.LowerBoundY
```

3 Specify the step response requirements using the following commands:

```
% Rise time requirement
constr.LowerBoundX=[0 10;10 15;15 50];
% Settling time requirement on lower line segments;
constr.LowerBoundX=[0 10;10 30;30 50];
% Settling time requirement on upper line segments
constr.UpperBoundX=[0 30;30 50];
% Overshoot requirement on upper bound
constr.UpperBoundY=[1.1 1.1;1.01 1.01];
```

Tip You can type `constr.`*`PropertyName`* to view the updated values of the properties.

---

4 Specify the parameters to optimize:

a Retrieve the vector of parameter objects from the project object, `proj`, using the `findpar` command.

```
param(1)=findpar(proj,'Kd');
param(2)=findpar(proj,'Ki');
param(3)=findpar(proj,'Kp');
```

b To view the properties of a parameter object, type the object name. For example:

```
param(1)
```

This command returns the properties of `param(1)` object.

```
              Name: 'Kd'
             Value: 0
      InitialGuess: 0
           Minimum: -Inf
           Maximum: Inf
      TypicalValue: 0
      ReferencedBy: {0x1 cell}
       Description: ''
             Tuned: 1

Tuned parameter.
```

c To limit the parameter to positive values, enter the minimum value for each parameter as `0` using the following commands:

```
param(1).Minimum=0;
param(2).Minimum=0;
param(3).Minimum=0;
```

Tip To view the updated value of the `Minimum` property for `Kd`, type `param(1).Minimum`.

---

5 Optimize the parameters using the `optimize` command:

```
opt=optimize(proj)
```

This command returns the following results:

Iter	S-count	f(x)	max constraint	Step-size	Directional derivative	First-order optimality	Procedure
0	1	0	207.1				
1	14	0	3.767	1.25	0	1	infeasible
2	21	0	0.4641	0.541	0	6.18	
3	28	-9.19412e-020	0.02942	0.0755	-1.22e-018	0.000891	
4	35	0	0	0.0233	3.95e-018	0.000114	Hessian modified

```
Successful termination.
Found a feasible or optimal solution within the specified tolerances.

Kd =

    0.0468


Ki =

    0.1455


Kp =

    0.1853


ans =

         Cost: 0
            X: [4x1 double]
     ExitFlag: 1
    Iteration: 4
```

- At each optimization iteration, the software simulates the model and the default optimization method `Gradient descent` modifies the controller parameters to reduce the distance between the simulated response and the design requirements. To learn more about the optimization method, see "Selecting Optimization Methods" in the *Simulink Design Optimization User's Guide*.
- After the optimization completes, you see the messages `Successful termination`, and `ExitFlag:1` at the MATLAB prompt. These messages indicate that the optimization method found a solution that meets the design requirements within the parameter bounds. For more information about the outputs displayed during the optimization, see "Displaying Iterative Output" in the Optimization Toolbox documentation.

  The optimized values of the parameters, `Kp`, `Ki` and `Kd`, are displayed at the command line, and are also written into the Simulink model.

Next, you refine the parameters by specifying a reference signal, as described in “Refining Model Parameters to Track a Reference Signal” on page 8-15.

Refining Model Parameters to Track a Reference Signal

In this portion of the tutorial, you optimize the model parameters to track a reference signal specified by the equation

$$y = 1 - \exp(-0.3 \times t)$$

You must have already optimized the model parameters to meet the step response requirements, as described in "Optimizing Model Parameters to Meet Step Response Requirements" on page 8-8.

To optimize the parameters to track a reference signal:

1. Specify the reference signal to track using the following commands:

```
% Enable reference tracking
constr.CostEnable='on';
% Specify the time values of the reference signal
constr.ReferenceX=linspace(0,50,200)';
% Specify the amplitude values of the reference signal
constr.ReferenceY=1-exp(-0.3*linspace(0,50,200))';
```

Tip To verify that the reference signal has been added to the signal constraint object, type `constr=findconstr(proj,'sldo_model1/Signal Constraint')`. The `ReferenceX` and `ReferenceY` properties now contain the reference signal.

---

2. Initialize the parameters in `proj` for optimization.

   a. Set the initial guess of the parameters using the `initpar` command.

```
% Copy parameter values from Simulink model to the project
initpar(proj)
```

   b. Update the scaling factor of the parameters with the initial parameter values using the following commands.

```
% Update scaling factor with the initial parameter value
param(1).TypicalValue = param(1).InitialGuess;
```

```
param(2).TypicalValue = param(2).InitialGuess;
param(3).TypicalValue = param(3).InitialGuess;
```

To learn more about the scaling factor, see "Changing Tuned Parameter Specifications" in the *Simulink Design Optimization User's Guide*.

When you optimize the model parameters after this initialization, the optimization method uses the updated parameter settings to recompute new parameter values.

3 Start the optimization using the `optimize` command.

```
opt=optimize(proj)
```

This command returns the following results:

```
                                   max                  Directional  First-order
 Iter  S-count         f(x)   constraint    Step-size   derivative   optimality  Procedure
    0        1       2.1547            0
    1       20     0.561528      0.02738        0.622        -7.04           74
    2       36     0.438343      0.03301        0.142        -1.13         1.84
    3       49     0.360718            0        0.296       -0.802          3.3
    4       60     0.271571            0        0.173        -0.64         2.69
    5       71     0.178588            0        0.321       -0.468         1.76
    6       82       0.1574            0        0.147       -0.215         1.28
    7       91     0.146542            0       0.0941       -0.168        0.824
    8      100     0.139688            0       0.0443       -0.208         0.37
    9      109     0.135451            0       0.0185       -0.294        0.102
   10      116     0.133726            0       0.0256      -0.0779        0.109
Optimization terminated due to slow progress in parameter or objective values.
To optimize further, go to Optimization Options and decrease the parameter and/or
function tolerances.

Kd =
    1.5821
Ki =

    0.2876


Kp =
```

```
    0.6311


ans =

         Cost: 0.1337
            X: [4x1 double]
     ExitFlag: 5
    Iteration: 10
```

- At each iteration, the optimization method modifies the controller parameters to minimize the error between the simulated response and the reference signal. Simultaneously, the method satisfies the step response requirements by reducing the distance between the simulated response and the design requirement line segments.
- After the optimization completes, you see the message `Optimization terminated due to slow progress in parameter or objective values`. This message indicates that the optimization method cannot minimize the error between the simulated response and the reference signal any further. If you want to refine the parameters further to track the reference signal more closely, change the parameter and function tolerances, as described in "Selecting Optimization Termination Options" in the *Simulink Design Optimization User's Guide*.

Tip To view the updated response of the model, connect a `Scope` block to the model and simulate the model to view the response.

---

4 Save the project by typing the following command:

```
save('proj');
```

9

Tutorial — Designing a PID Controller Using Optimization-Based Tuning

- "About This Tutorial" on page 9-2
- "Configuring a Project for Optimization-Based Control Design" on page 9-5
- "Designing an Initial PID Controller to Meet Bode Magnitude and Phase Margins Requirements" on page 9-11
- "Refining the Controller Design to Meet Controller Output Bounds" on page 9-32
- "Saving the Project" on page 9-48

About This Tutorial

In this section...

"Objectives" on page 9-2

"About the Model" on page 9-2

"Design Requirements" on page 9-4

Objectives

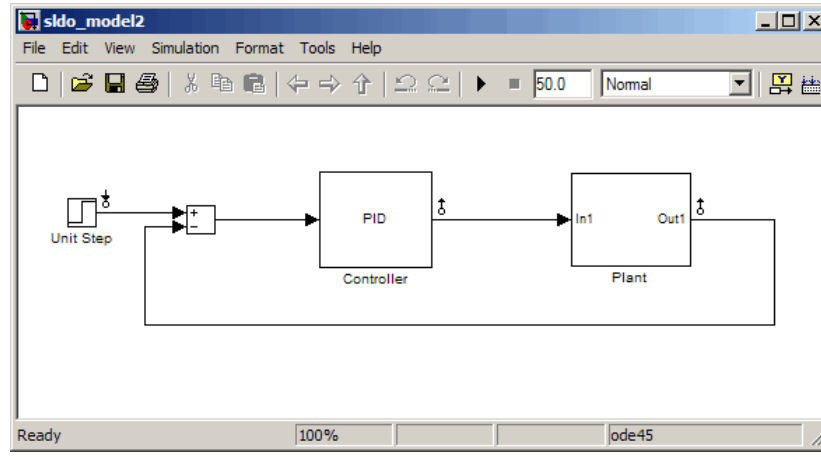
In this tutorial, you learn to use optimization methods to design a PID controller to meet frequency-domain design requirements on a system's response.

You accomplish the following tasks using the GUI:

- Specify frequency-domain Bode magnitude and phase margin requirements.
- Design an initial controller to meet the frequency-domain requirements.
- Refine the initial controller design to limit the controller's output signal.

About the Model

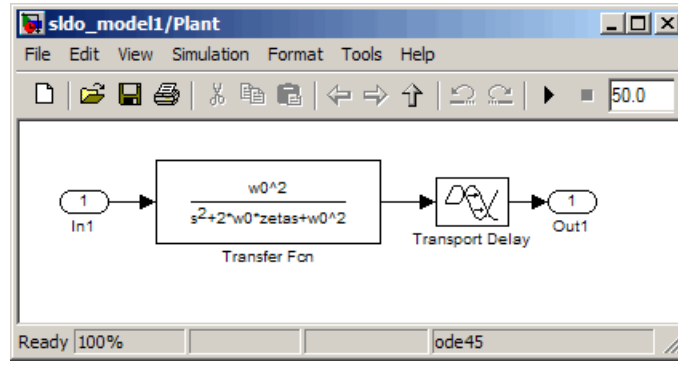
In this tutorial, you use the Simulink model named `sldo_model2`, as shown in the next figure.



The model contains a `Controller` block, which is a PID Controller. This block controls the output of the `Plant` subsystem.

Using the Simulink Control Design software has linearized the Simulink model at the operating point specified in the model. The `sldo_model2.mat` file contains a preconfigured SISO Design Tool session saved after linearizing the model. To learn more about linearizing Simulink models for control design, see "Designing Compensators" in the *Simulink Control Design User's Guide*.

Double-click the `Plant` subsystem to open it. The plant is modeled as a second-order system with delay. It contains Transfer Function and Transport Delay blocks, as shown in the next figure.



To learn more about the blocks, see Transfer Fcn and Transport Delay block reference pages.

Design Requirements

The compensator you design in this tutorial must meet the following design requirements:

- Bode lower magnitude bound of 0 in the frequency range 1e-3 to 1 rad/sec
- Phase margin greater than 60 degrees
- Controller output bounds in the range [-250 550]

Configuring a Project for Optimization-Based Control Design

To design a linear controller for a Simulink model, you must first configure a Control and Estimation Tools Manager project.

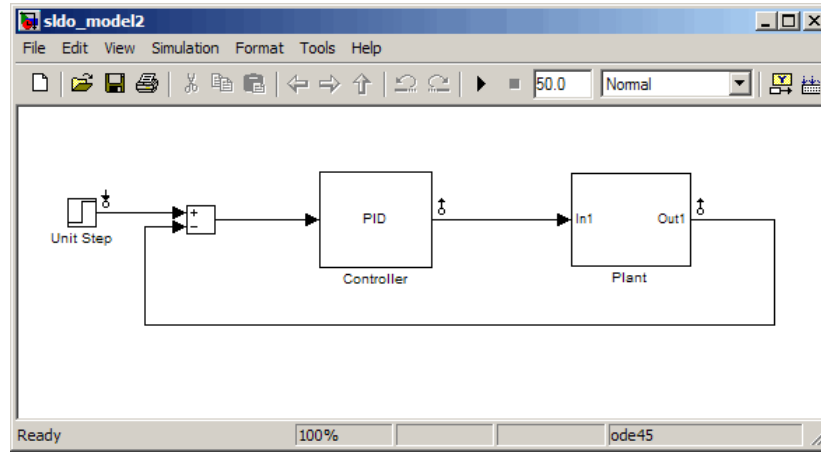
1 Open a SISO Design Tool session for the linearized Simulink model by typing the following command at the MATLAB prompt:

```
sisotool('sldo_model2.mat')
```

Note `sldo_model2.mat` file contains a preconfigured SISO Design Tool session. This session was saved after Simulink Control Design software linearized the `sldo_model2` Simulink model.

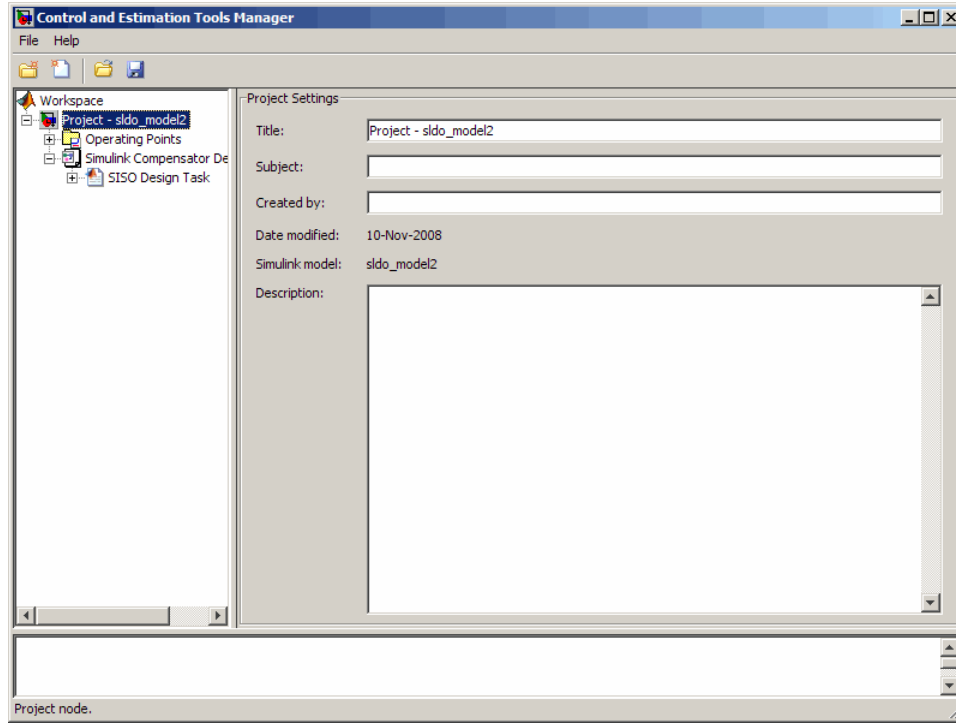
This command opens the following windows:

- Simulink model

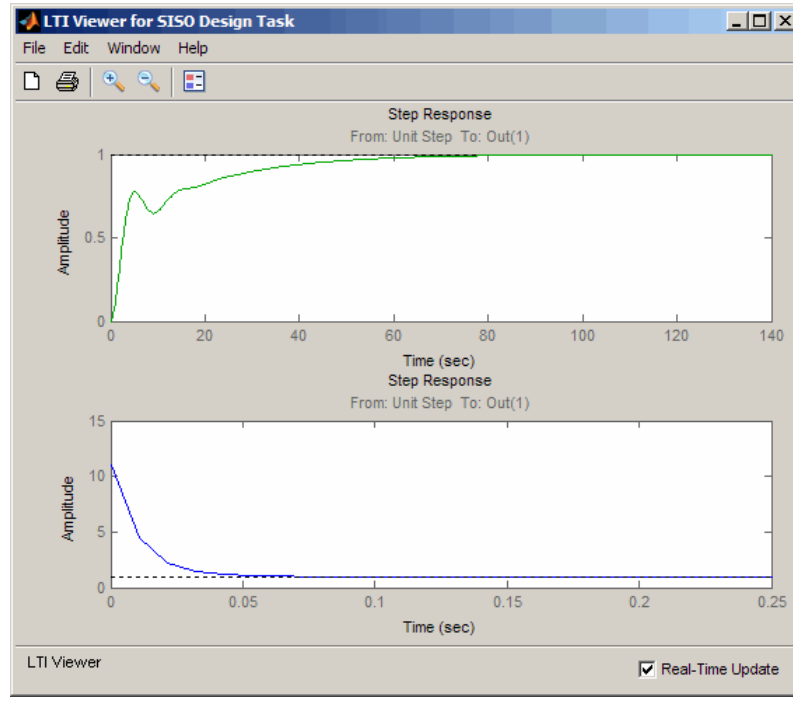


To learn more about the model, see “About the Model” on page 9-2.

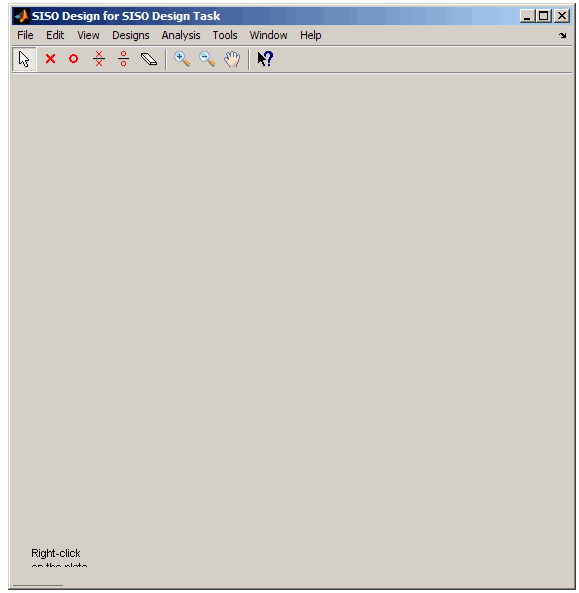
- Control and Estimation Tools Manager GUI, which contains a **SISO Design Task** node under the **Simulink Compensator Design Task** node.



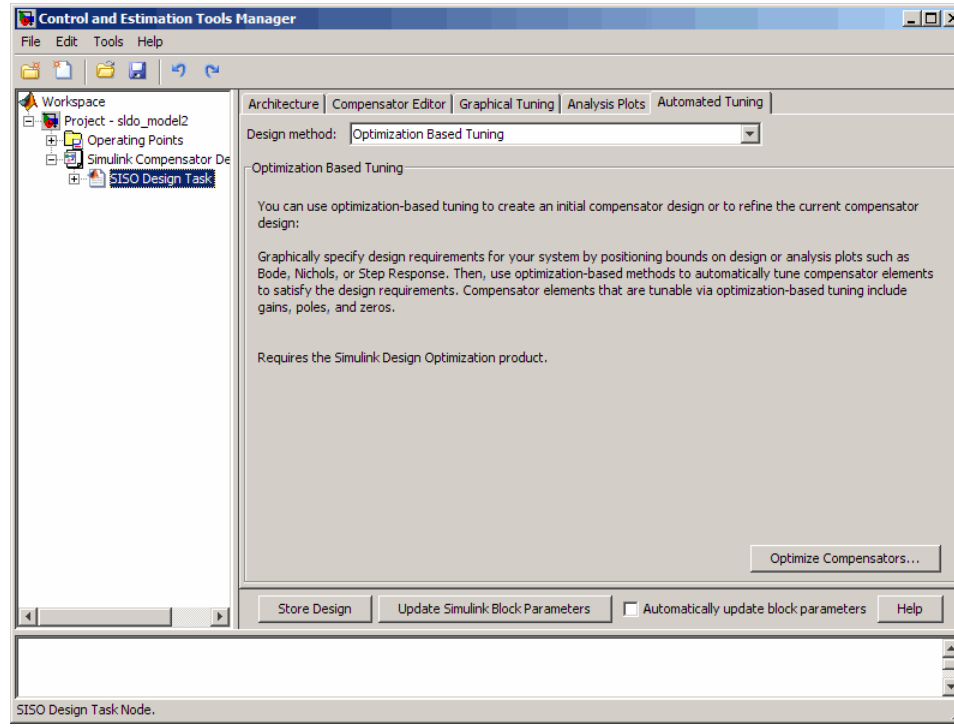
- LTI Viewer for SISO Design Task window, which contains the following plots:
  - Closed-loop step response of the system in the top plot
  - Output of the `Controller` block in the bottom plot



- A blank SISO Design for SISO Design Task window

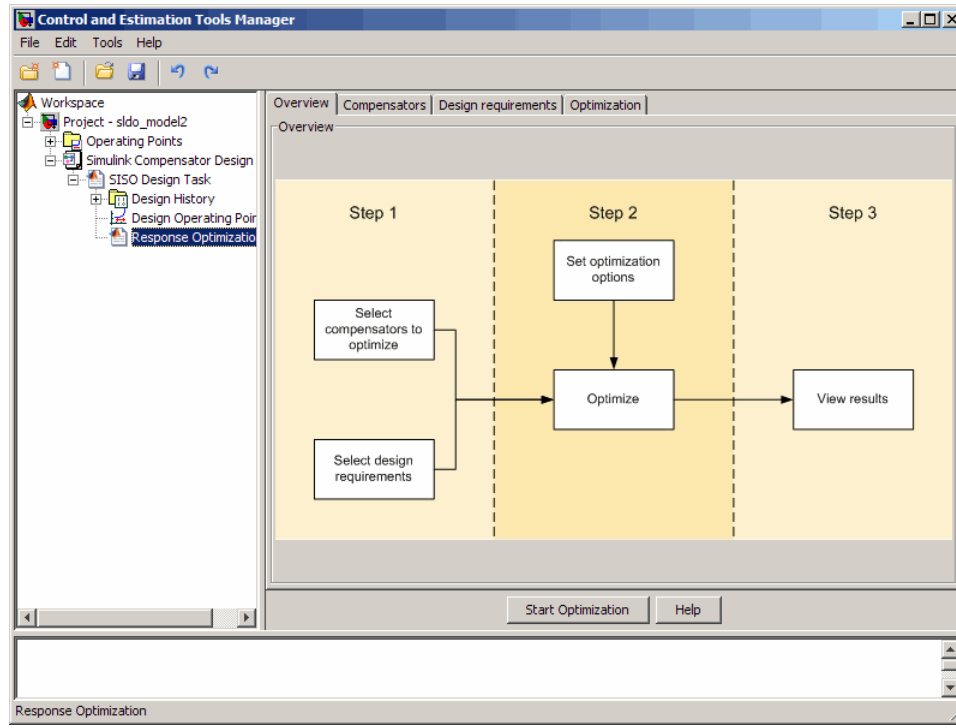


2 In the Control and Estimation Tools Manager GUI, select the **Automated Tuning** tab in the **SISO Design Task** node.



3 Click **Optimize Compensators**.

This action creates a new **Response Optimization** node.



Designing an Initial PID Controller to Meet Bode Magnitude and Phase Margins Requirements

In this section...
"Specifying the Controller Parameters" on page 9-11
"Specifying Bode Magnitude and Phase Margin Design Requirements" on page 9-15
"Designing the Controller" on page 9-25

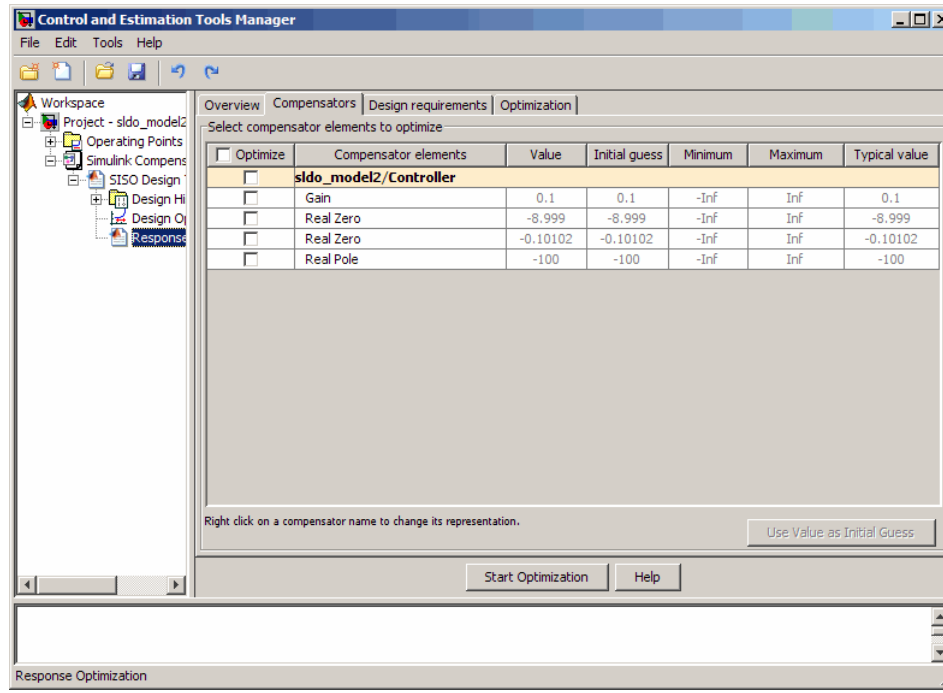
Specifying the Controller Parameters

In this portion of the tutorial, you specify the controller parameters to design.

You must have already configured a project for control design, as described in "Configuring a Project for Optimization-Based Control Design" on page 9-5.

To specify the controller parameters to design:

1 In the Control and Estimation Tools Manager GUI, select the **Compensators** tab in the **Response Optimization** node.



The controller parameters appear as poles and zeros in the **Compensator elements** column and represent the following:

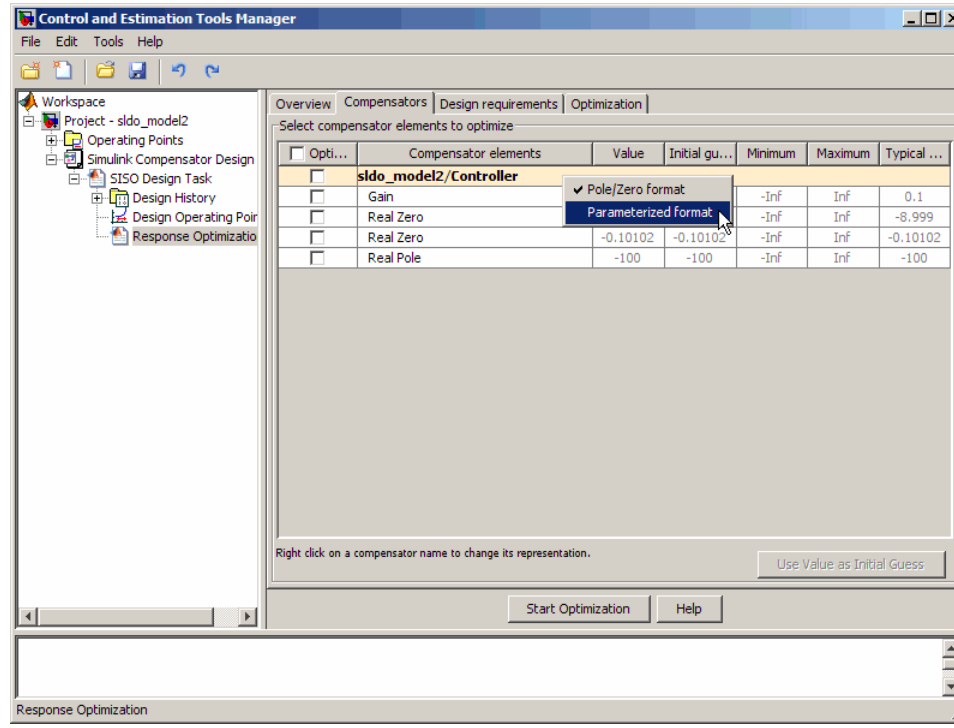
- Gain — Overall gain of the controller
- Real zeros — Zeros resulting from the differentiator and integrator
- Real pole — Pole resulting from the low-pass filter of the differentiator

---

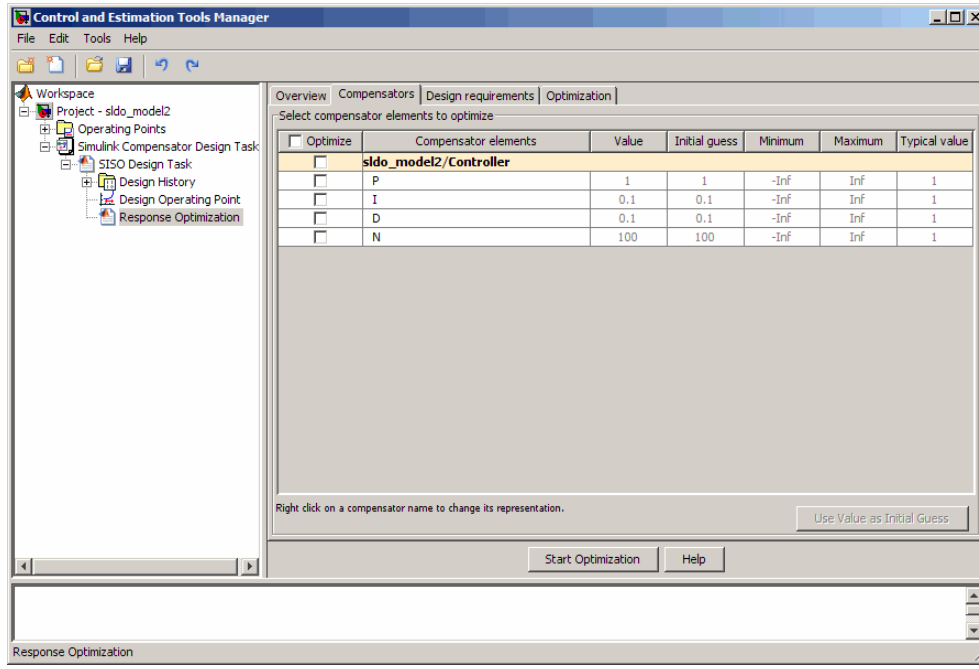
Tip To view the structure of the `Controller` block, right-click the block and select **Look Under Mask**.

---

2 For convenience, change the PID controller parameters to Simulink block mask parameters format. To do so, right-click the **sldo_model2/Controller** column, and select **Parameterized format**.



This action displays the controller parameters as Simulink block mask parameters `P`, `I`, and `D`, as shown in the next figure. To learn more about mask parameters, see "Mask Parameters" in the Simulink documentation.

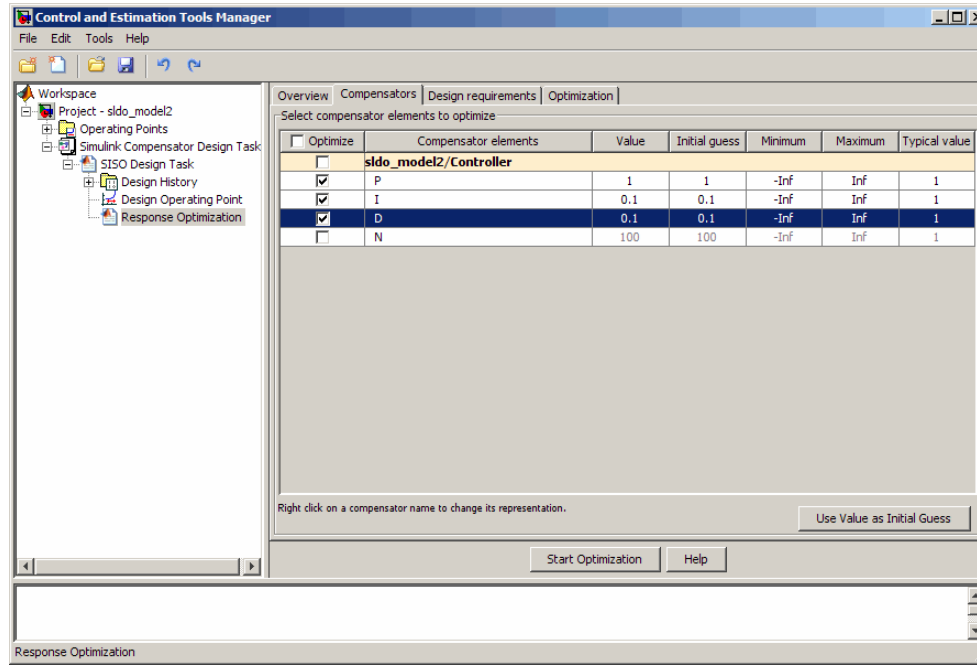


The **Compensators** tab displays the following parameter settings:

- **Value** — Current controller parameter value
- **Initial Guess** — Initial controller parameter value
- **Minimum** and **Maximum** — Controller parameter bounds
- **Typical Value** — Scaling factor for the controller parameter

To learn more about the parameter settings, see "Changing Tuned Parameter Specifications" in the *Simulink Design Optimization User's Guide*.

3 In the **Optimize** column, select the check boxes for `P`, `I`, and `D`.



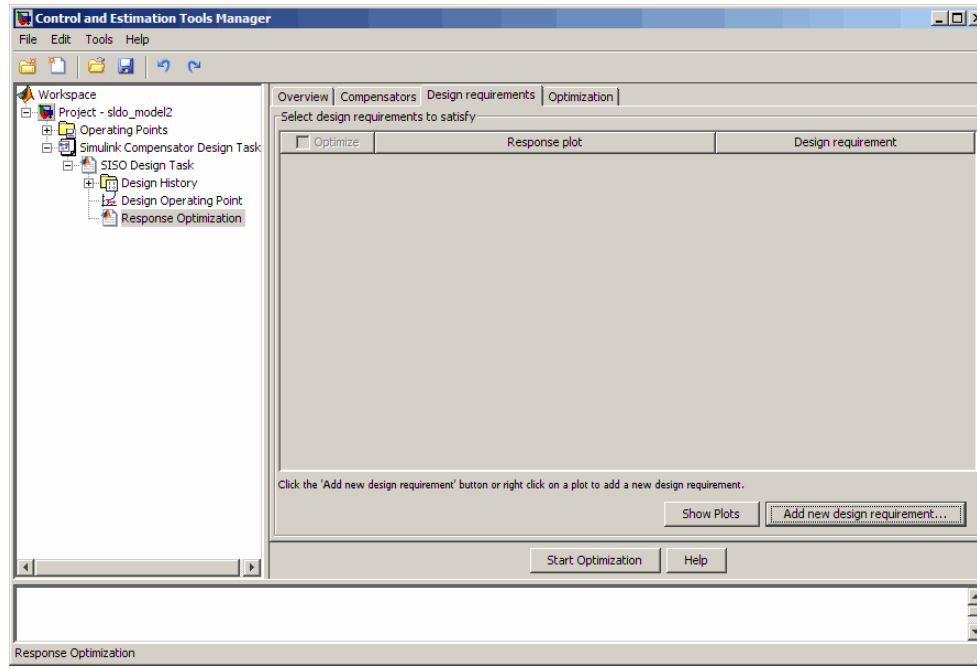
Specifying Bode Magnitude and Phase Margin Design Requirements

In this portion of the tutorial, you specify the Bode magnitude and phase margin requirements that the controller must satisfy.

Before you specify the design requirements, you can specify the controller parameters to design, as described in “Specifying the Controller Parameters” on page 9-11.

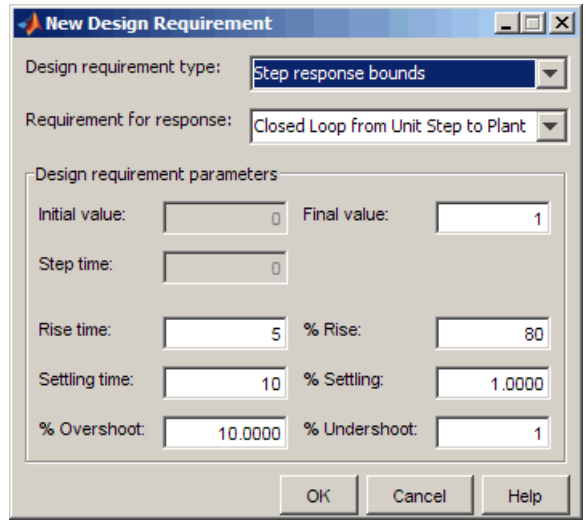
To specify the Bode design requirements:

1 In the **Response Optimization** node of the Control and Estimation Tools Manager GUI, select the **Design requirements** tab.



2 Click **Add new design requirement**.

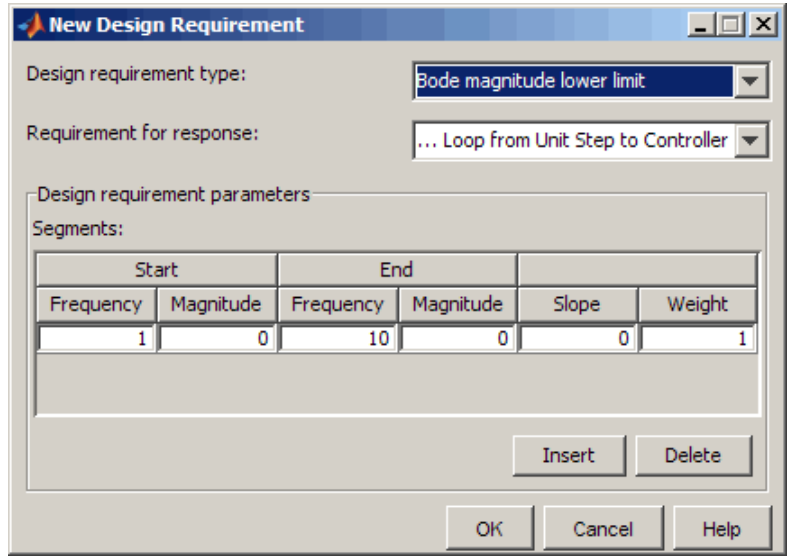
This action opens the New Design Requirement dialog box.



3 Specify the Bode magnitude lower limit requirement:

a In the New Design Requirement dialog box, select `Bode magnitude lower limit` from the **Design requirement type** drop-down list.

This action updates the New Design Requirement dialog box, as shown in the next figure.

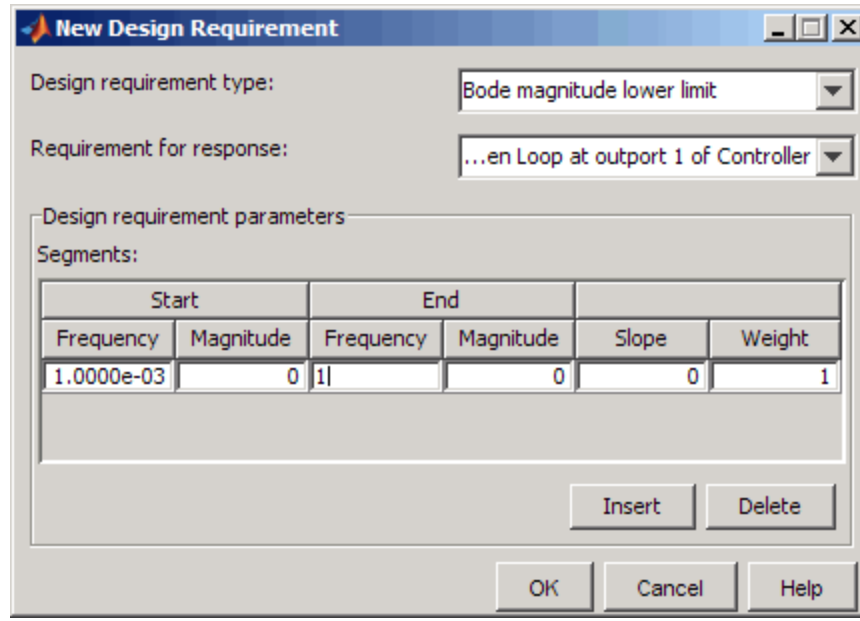


b Select `Open Loop at outport 1 of Controller` from the **Requirement for response** drop-down list.

c In the **Frequency** field of the **Start** column, enter `1e-3`.

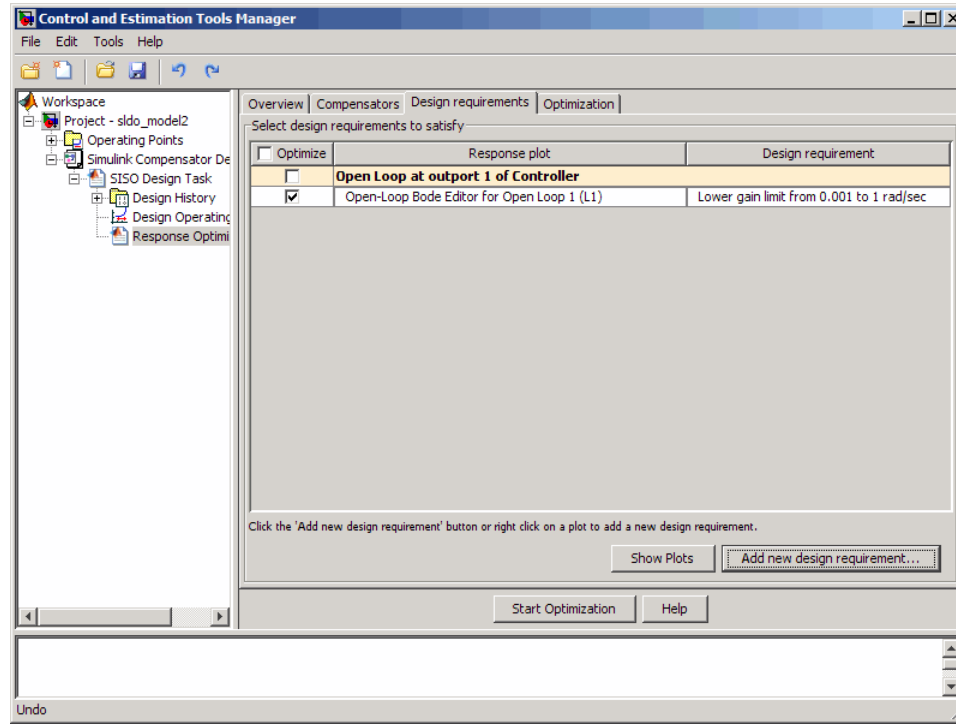
d In the **Frequency** field of the **End** column, enter `1`.

The New Design Requirement dialog box resembles the following figure.

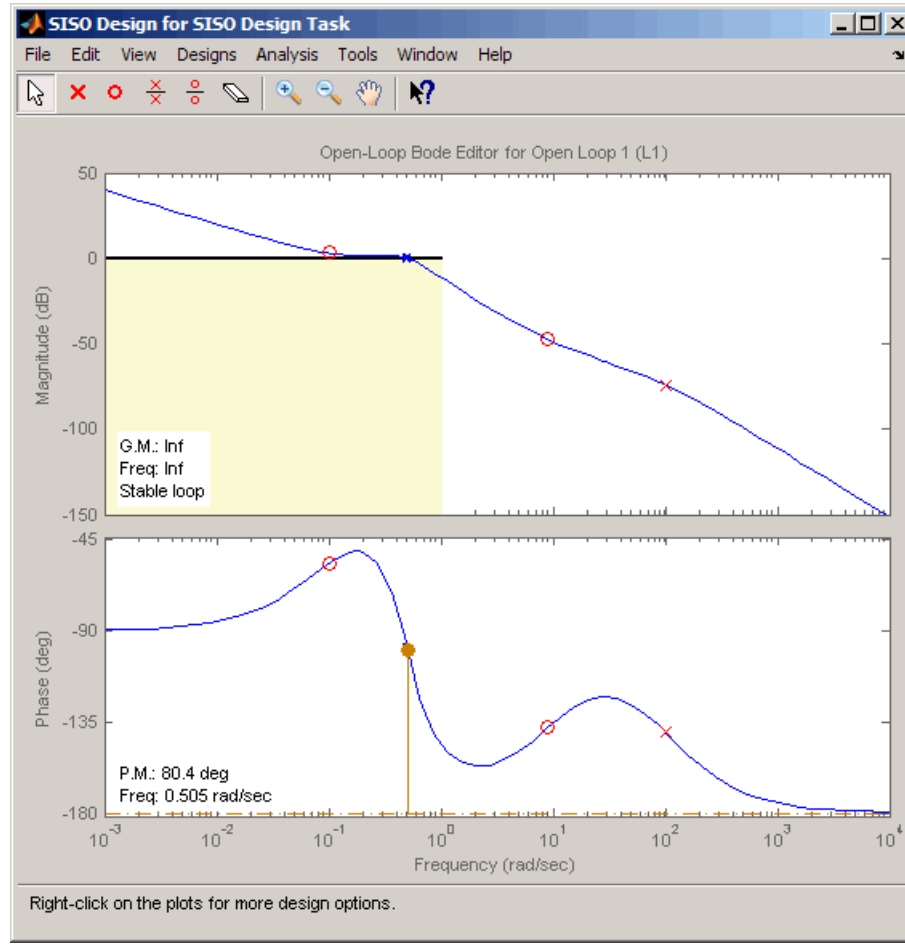


e Click **OK** to close the New Design Requirement dialog box.

The Bode lower magnitude limit is added to the **Design requirements** tab, as shown in the next figure.

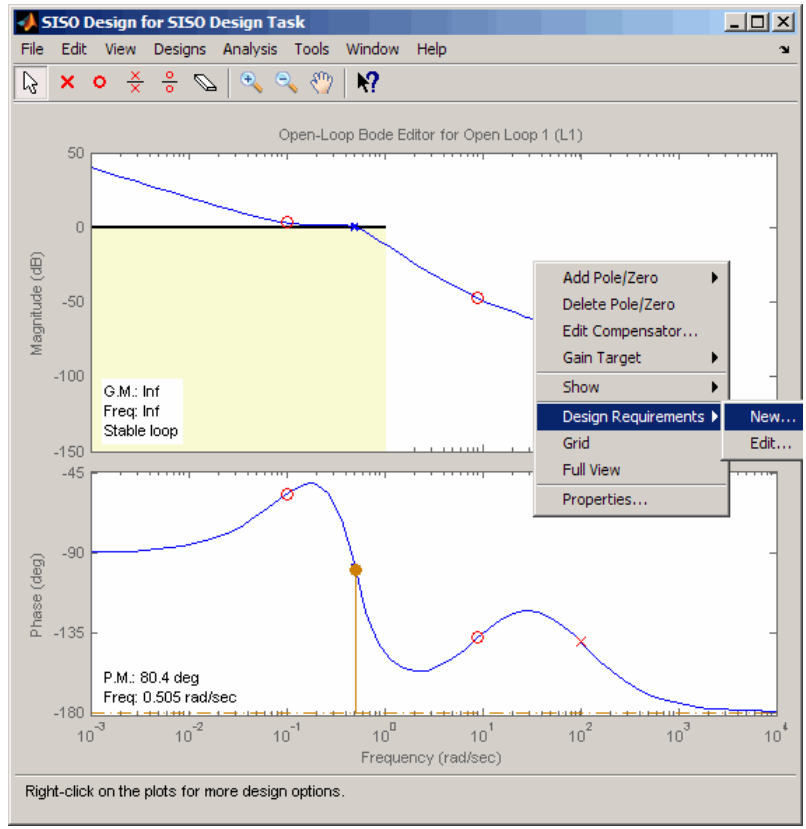


The SISO Design for SISO Design task window also updates to show the Bode plot with the design requirement displayed as the black line segment.

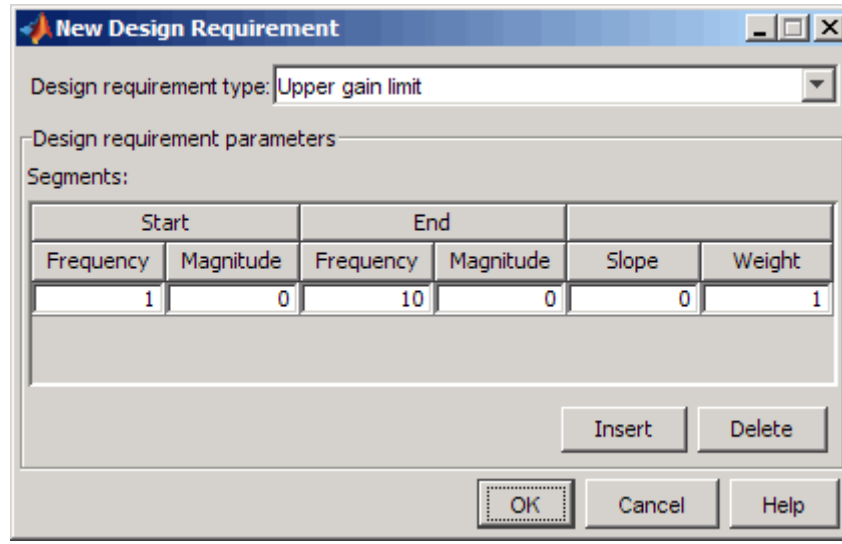


4 Add the phase margin requirement:

a In the SISO Design window, right-click the white area on the top plot, and select **Design requirement > New**.

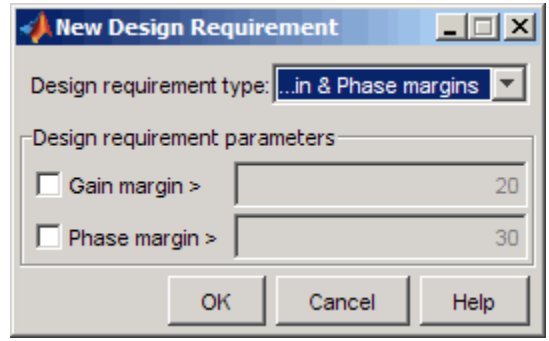


This action opens the New Design Requirement dialog box, as shown in the next figure.

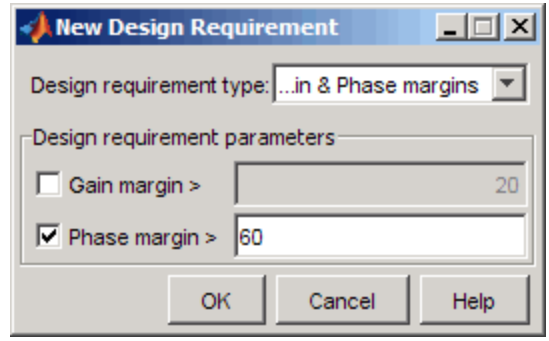


b In the New Design Requirement dialog box, select `Gain & phase margins` from the **Design requirement type** drop-down list.

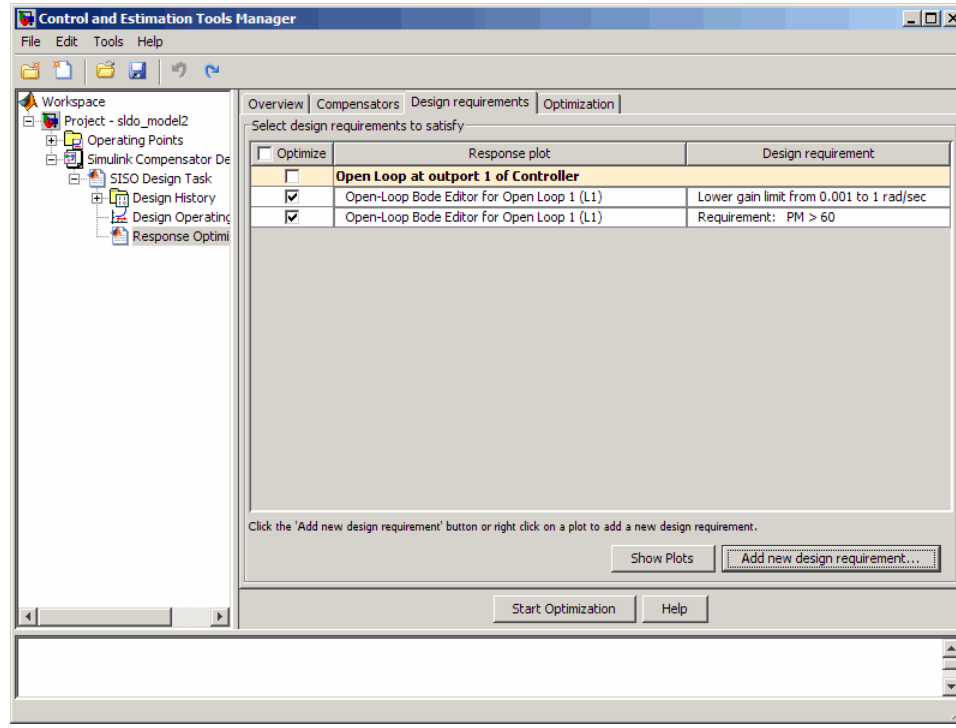
The New Design Requirement dialog box updates to display the **Gain Margin >** and **Phase Margin >** options, as shown in the next figure.



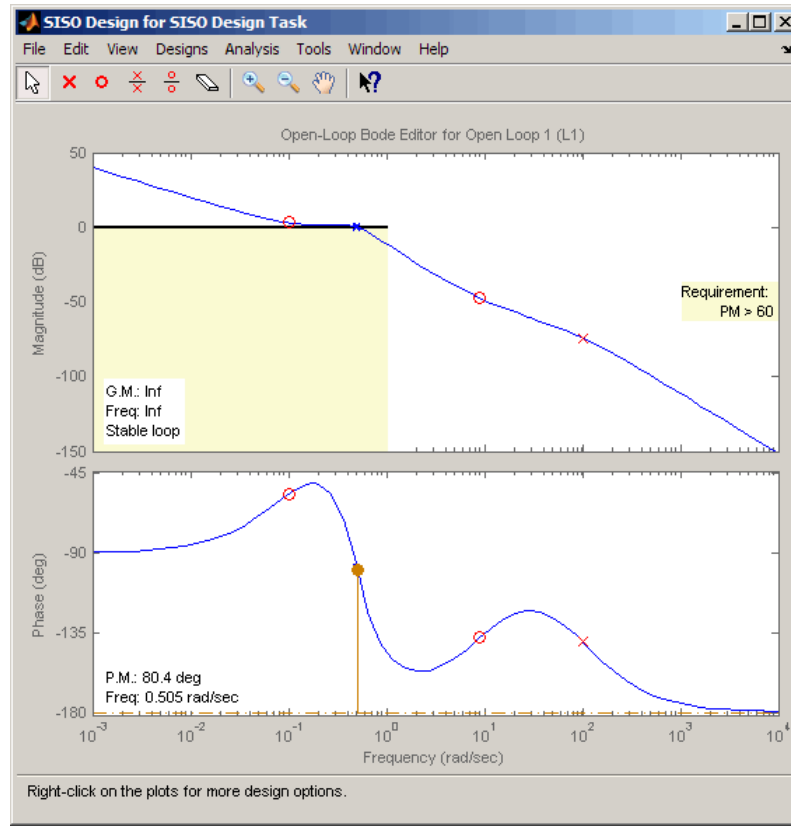
c Select the **Phase margin >** check box, and enter `60` in the adjacent field. Then, click **OK** to close the dialog box.



The **Design requirements** tab in the Control and Estimation Tools Manager GUI updates. It now displays the phase margin requirement, as shown in the next figure.



The SISO Design window also updates to display the phase margin requirement.



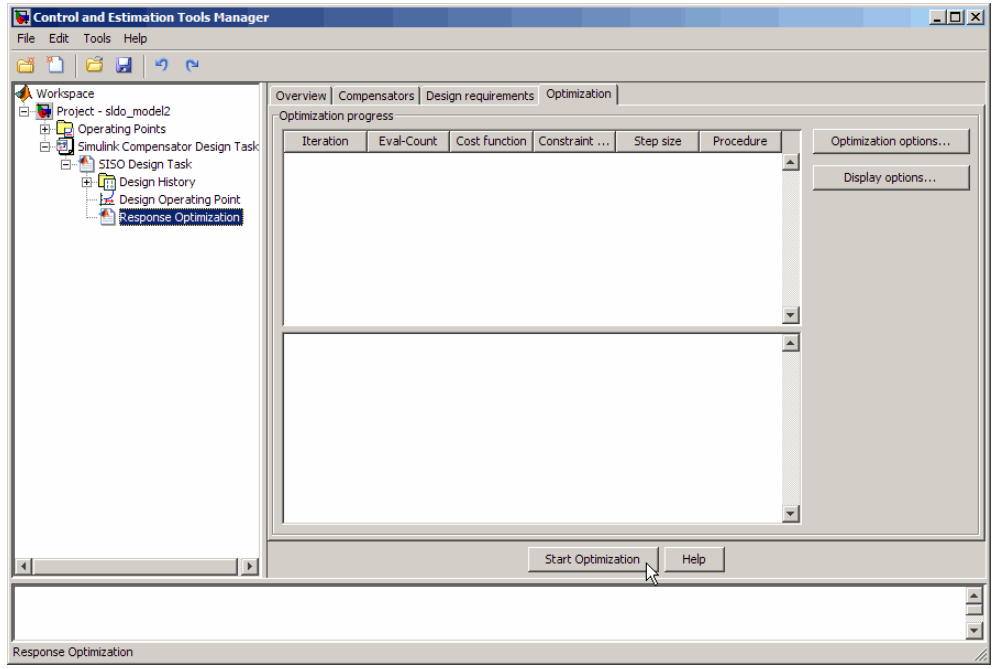
Designing the Controller

In this portion of the tutorial, you design the controller to meet the Bode magnitude and phase requirements.

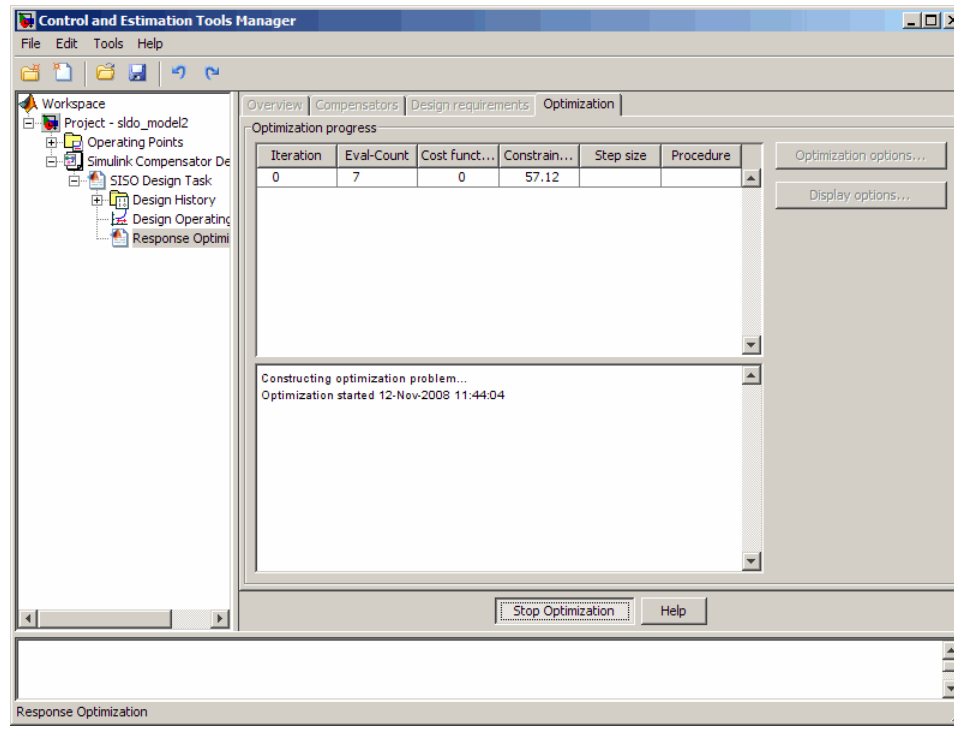
You must have already specified the controller parameters to design, as described in “Specifying the Controller Parameters” on page 9-11, and the design requirements, as described in “Specifying Bode Magnitude and Phase Margin Design Requirements” on page 9-15.

To design the controller:

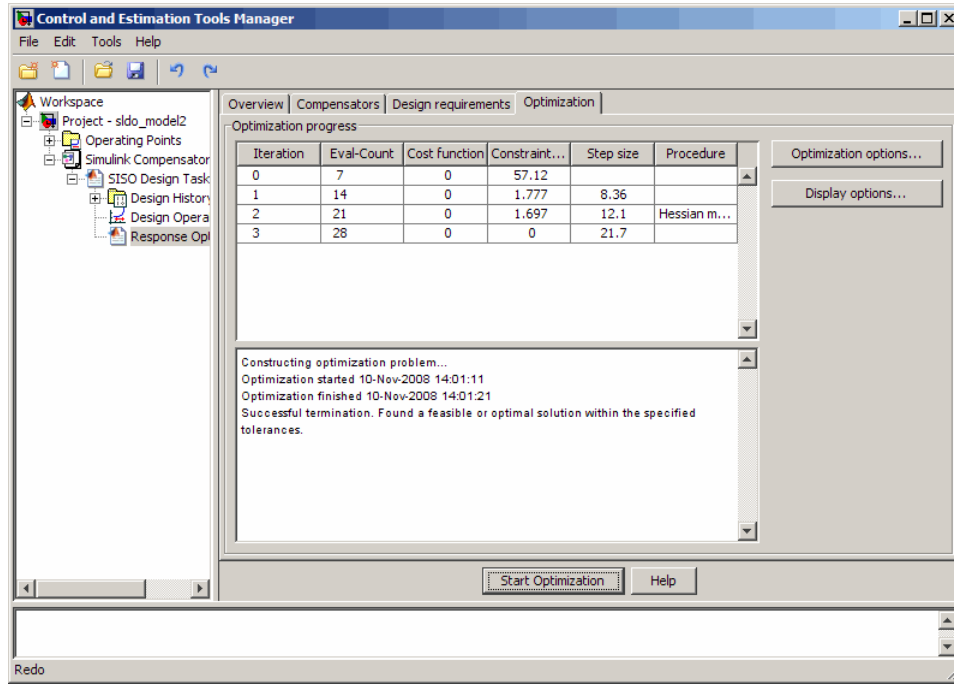
1 In the **Optimization** tab, click **Start Optimization**.



The **Optimization** tab updates as shown in the next figure.



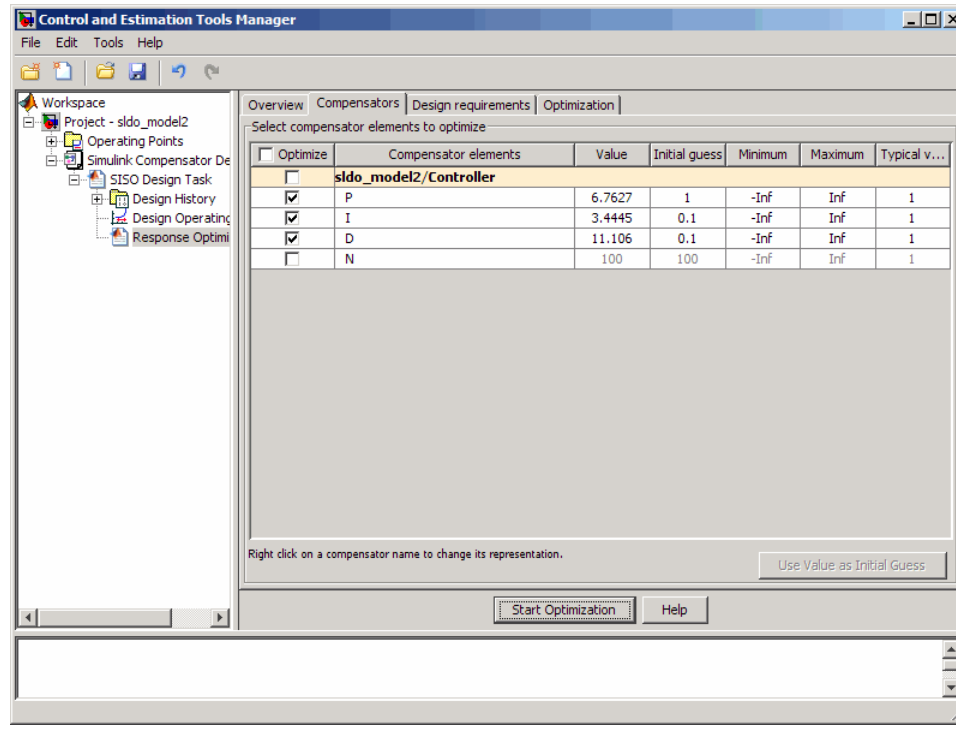
- At every optimization iteration, the default optimization method `Gradient descent` reduces the distance between the current response and the magnitude requirement line segment by modifying the controller parameters. Simultaneously, the software also computes the phase margin and reduces the distance between the current response and the phase margin. To learn more about the optimization method, see "Selecting Optimization Methods" in the *Simulink Design Optimization User's Guide*.
- After the optimization completes, the **Optimization** tab displays the optimization iterations and status, as shown in the next figure.



The message `Successful termination` indicates that the optimization method found a solution that meets the design requirements. For more information about the outputs displayed in the **Optimization progress** table, see "Displaying Iterative Output" in the Optimization Toolbox documentation.

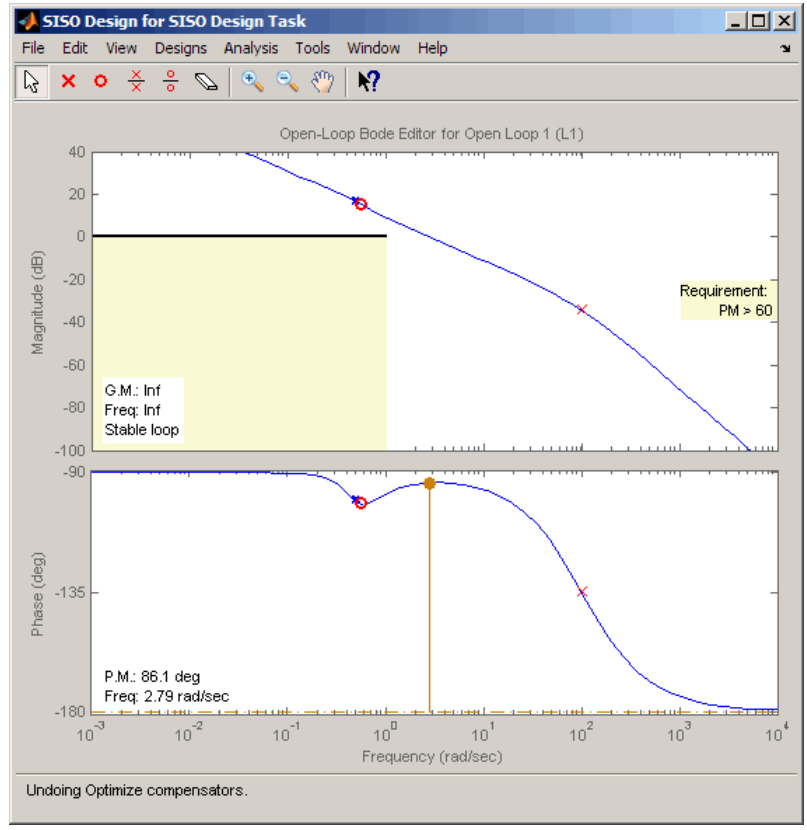
2 Examine the controller parameters and the system's response:

a View the optimized parameter values in the **Value** field of the **Compensator** tab.



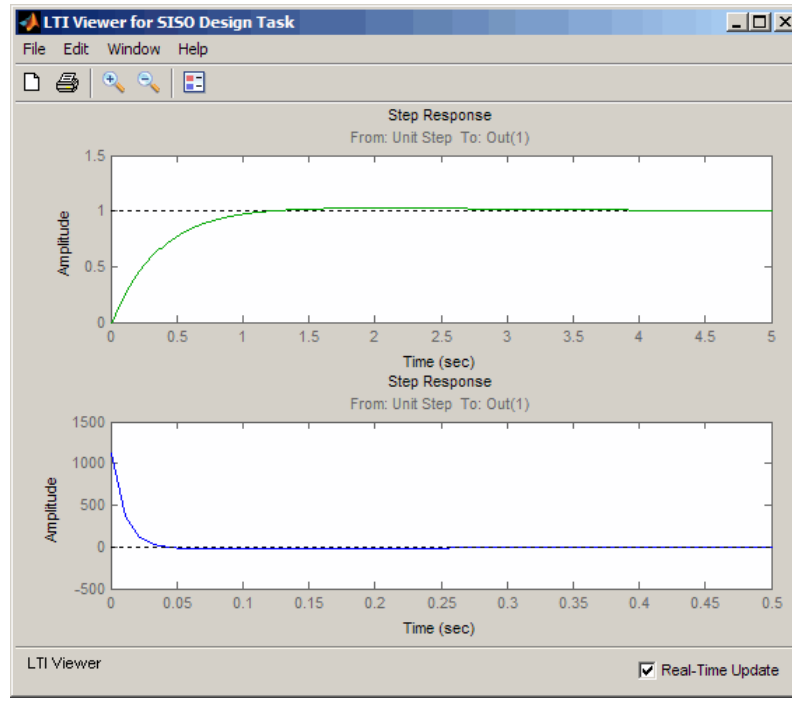
b Examine the system's response on the following plots:

- The SISO Design window



  - The top plot shows that the magnitude of the system, displayed as the blue curve, lies outside the yellow region. This plot indicates that the system has met the Bode magnitude requirement.
  - The bottom plot displays the phase margin (P.M.) value of 86.1 degrees. This indicates that the system has met the phase margin design requirement of >60 degrees.

- The LTI Viewer



  - The top plot shows that the closed-loop response of the system is stable. The system with the designed controller thus meets both the magnitude and phase margin requirements.
  - The bottom plot in the LTI Viewer shows that the peak value of the controller's output is 1000, which is very large and can cause damage to the plant. To limit the controller output, you apply lower and upper bounds on the signal, as described in "Refining the Controller Design to Meet Controller Output Bounds" on page 9-32.

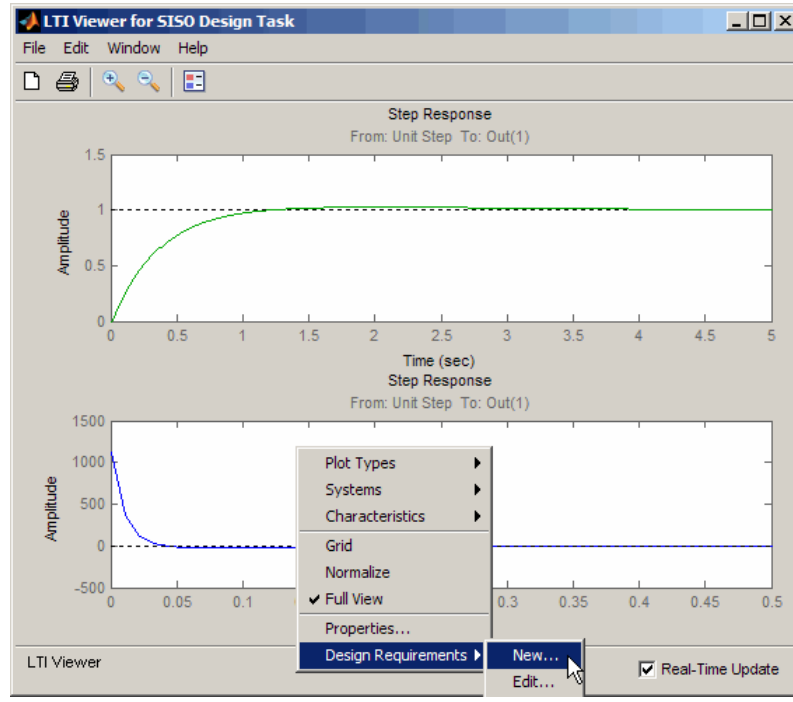
Refining the Controller Design to Meet Controller Output Bounds

In this portion of the tutorial, you refine the controller to satisfy bounds on the controller's output.

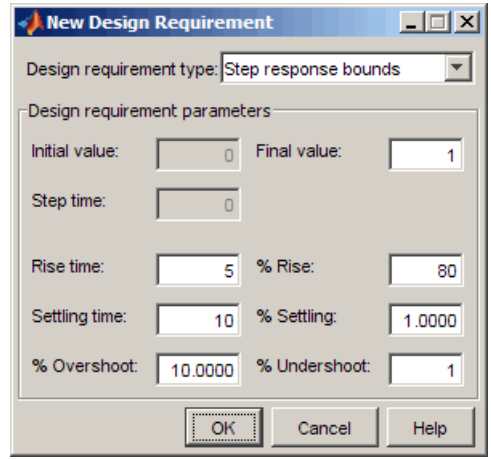
You must have already designed an initial controller, as described in "Designing an Initial PID Controller to Meet Bode Magnitude and Phase Margins Requirements" on page 9-11.

To tune the compensator parameters to meet the bounds on the controller's signal:

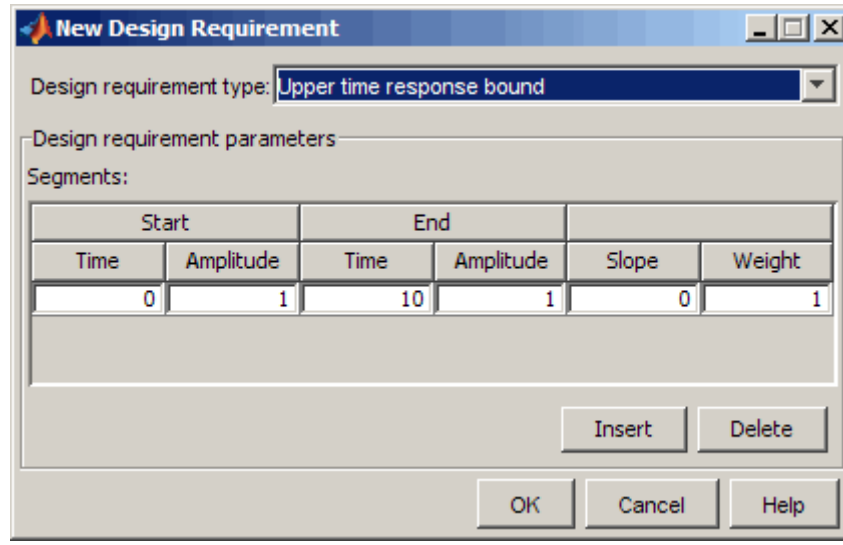
1. Add the upper-bound on the controller's output:
   a. In the LTI Viewer, right-click the white area on the bottom plot, and select **Design requirement > New**.



This action opens the New Design Requirement dialog box.



b In the New Design Requirement dialog box, select `Upper time response bound` from the **Design requirement type** drop-down list.

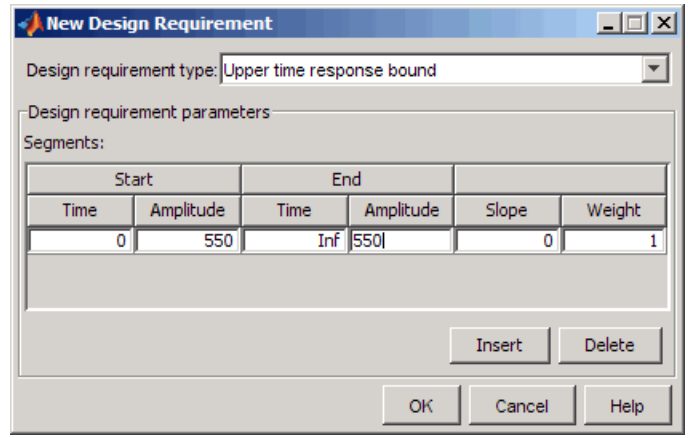


c In the **Time** field of the **End** column, enter `Inf`.

d In the **Amplitude** field of the **Start** column, enter `550`.

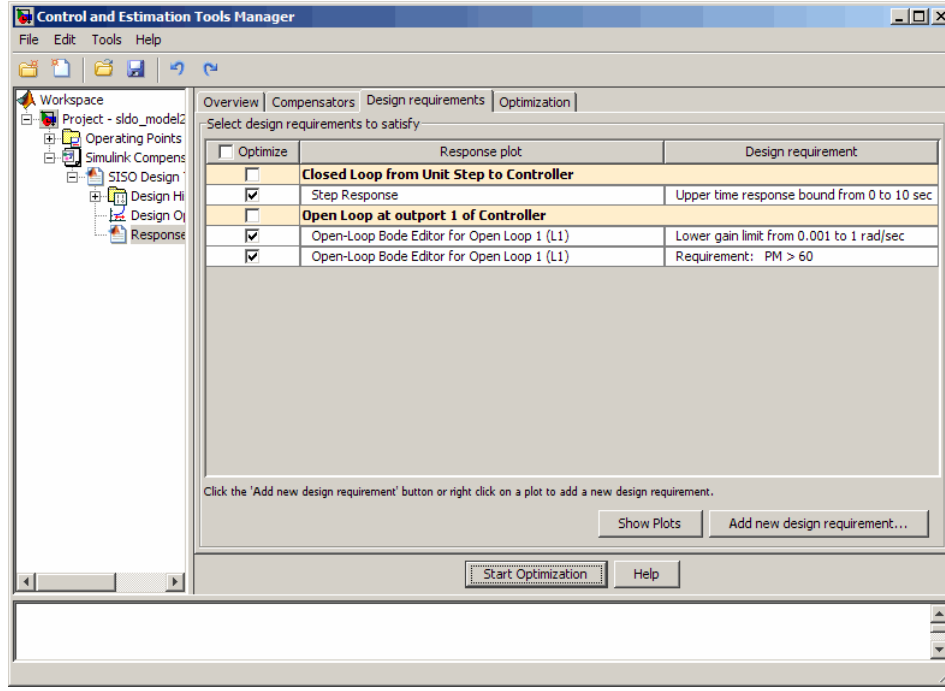
e In the **Amplitude** field of the **End** column, enter `550`.

The New Design Requirements dialog box resembles the following figure.

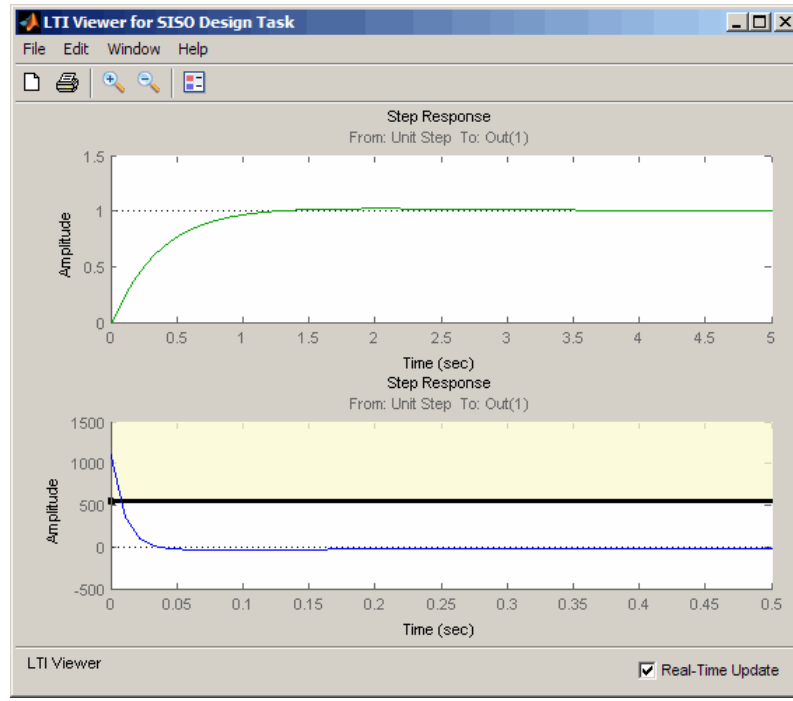


f Click **OK** to close the dialog box.

The **Design requirements** tab in the Control and Estimation Tools Manager GUI updates. It now displays the upper-bound requirement, as shown in the next figure.



The LTI Viewer also updates to show the design requirement, as shown in the next figure.



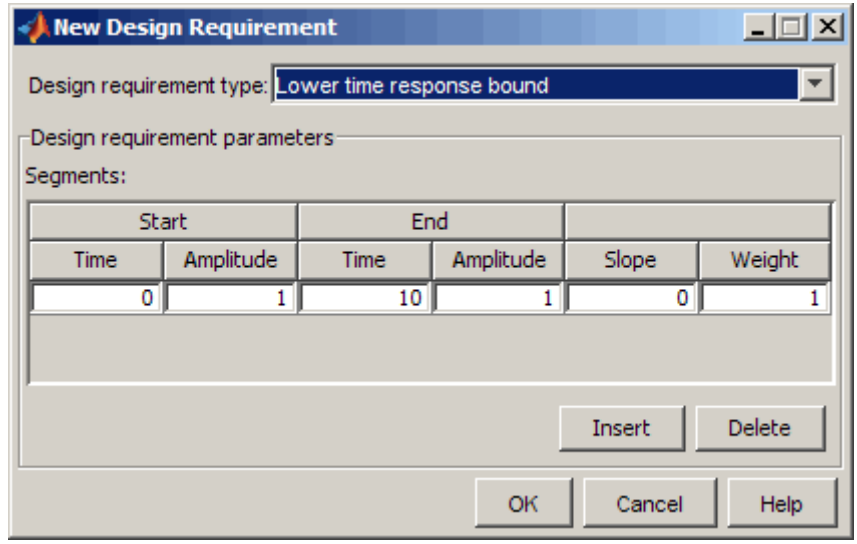
2 Add the lower-bound on the controller's output:

a Right-click the white area in the bottom plot in the LTI Viewer, and select **Design requirement > New**.

This action opens the New Design Requirement dialog box.

b Select `Lower time response bound` from the **Design requirement type** drop-down list.

The New Design Requirement dialog box updates to show the lower bound, as shown in the next figure.

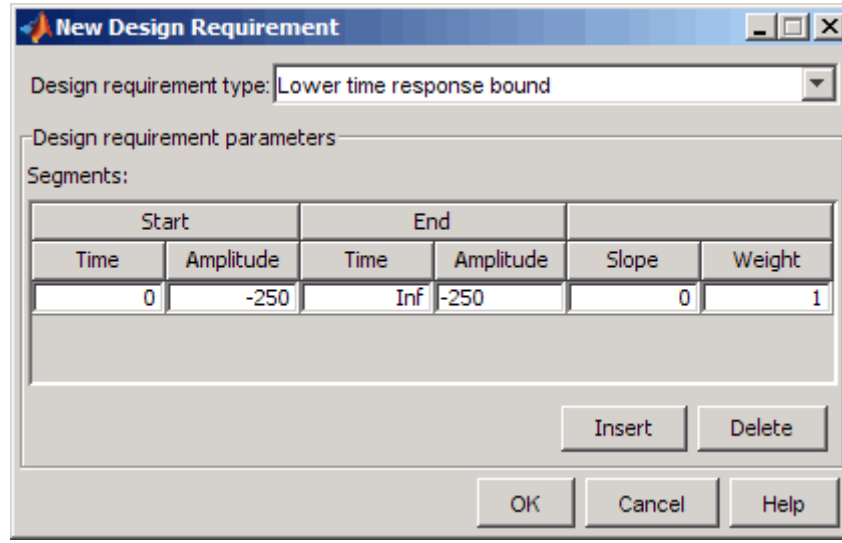


c In the **Time** field of the **End** column, enter `Inf`.

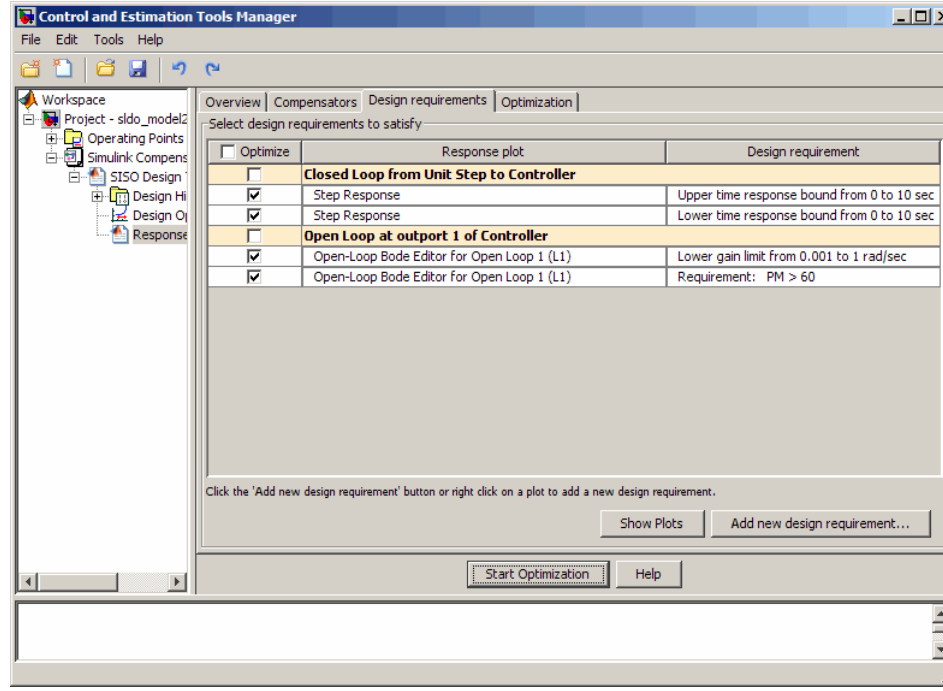
d In the **Amplitude** field of the **Start** column, enter `-250`.

e In the **Amplitude** field of the **End** column, enter `-250`.

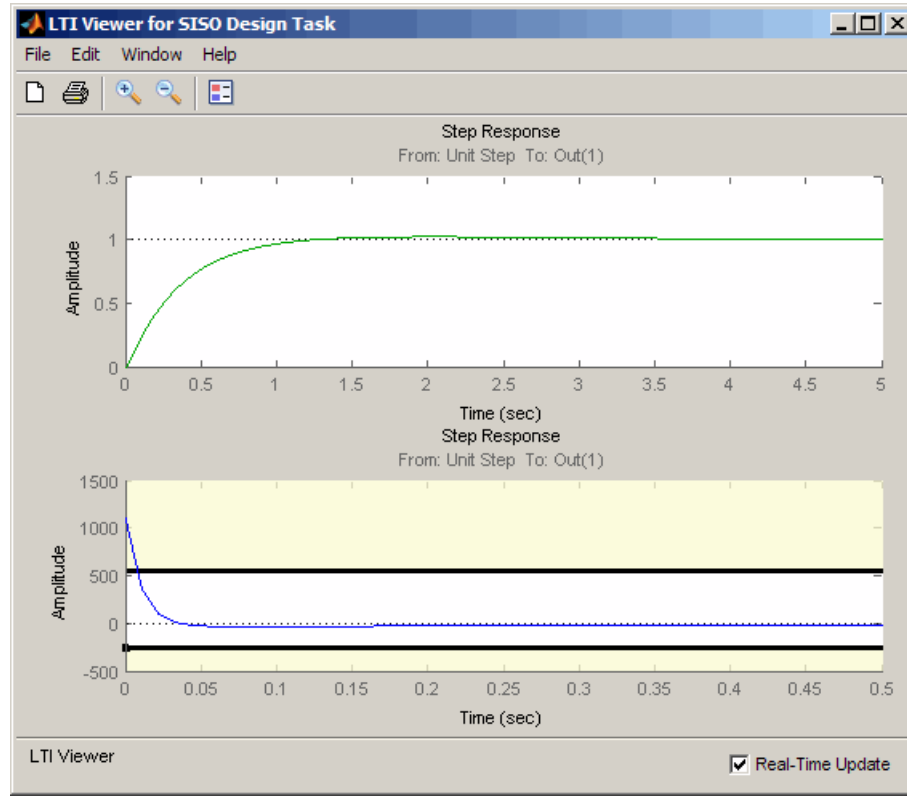
The New Design Requirement dialog box resembles the next figure. Click **OK** to close the dialog box.



The **Design requirements** tab in the Control and Estimation Tools Manager GUI updates. It now displays the lower-limit requirement, as shown in the next figure.

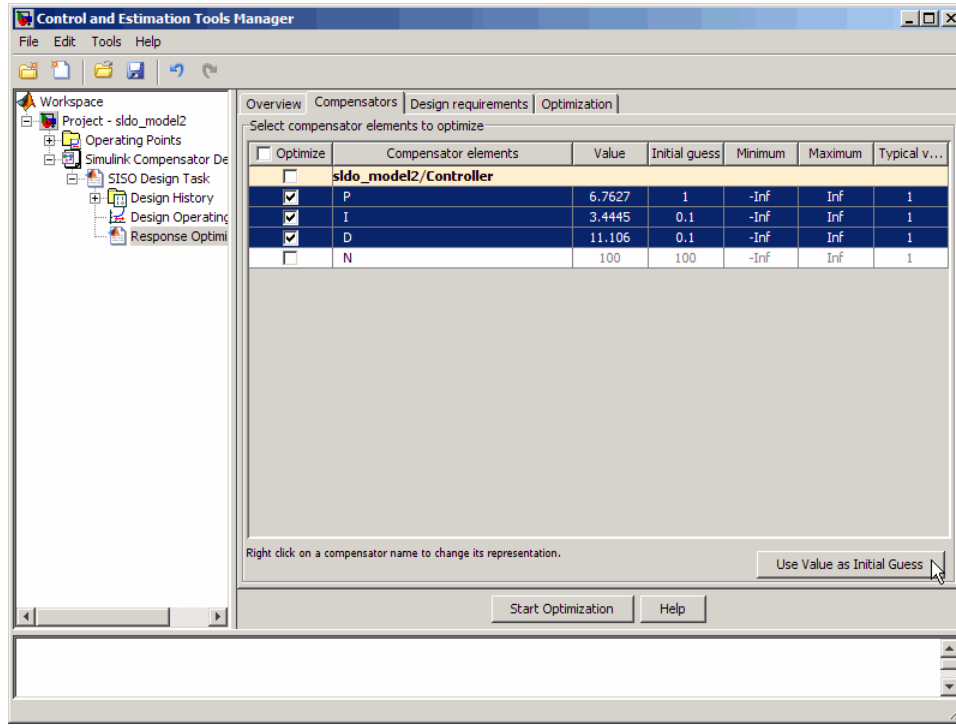


The LTI Viewer also updates to show the lower-bound on the controller's output, as shown in the next figure.



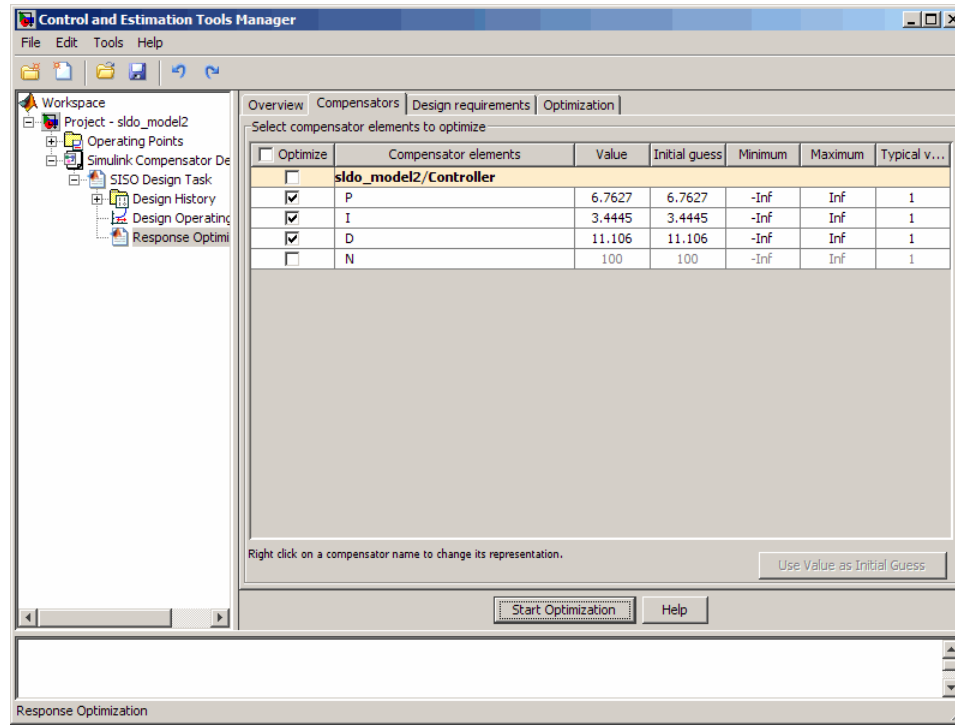
3 Optimize the parameters to meet the design requirements on the controller output:

a In the **Compensators** tab, select the rows containing `P`, `I`, and `D`, and click **Use Value as Initial Guess**.



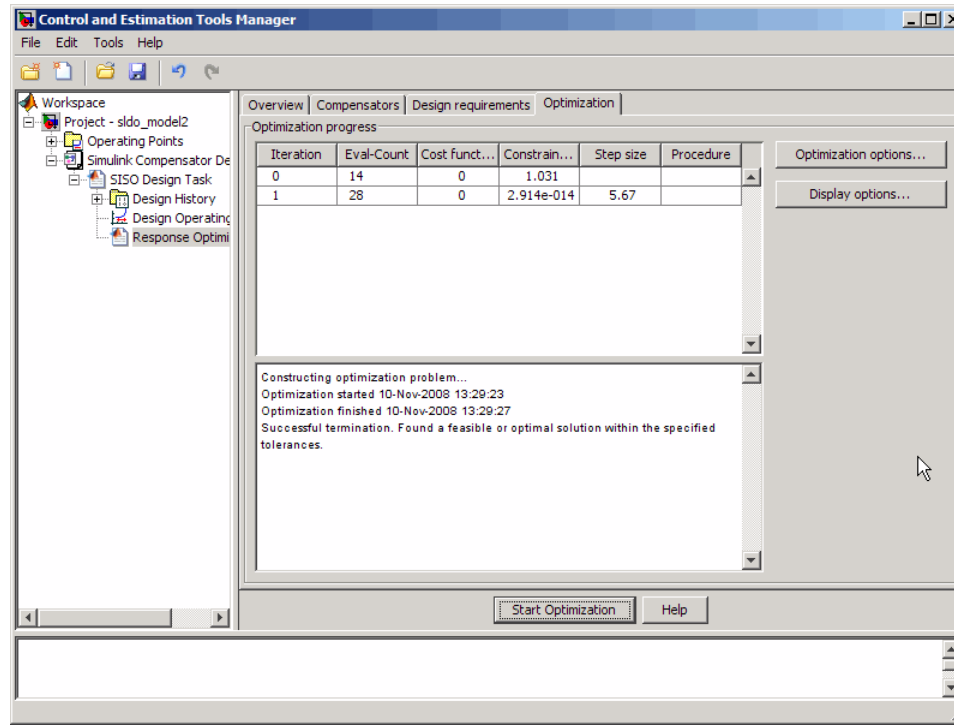
When you run the optimization again, the optimization method uses the updated parameter values as the starting point for refining the values.

Clicking **Use Value as Initial Guess** updates the values in the **Initial Guess** column, as shown in the next figure.



b In the **Optimization** tab, click **Start Optimization**.

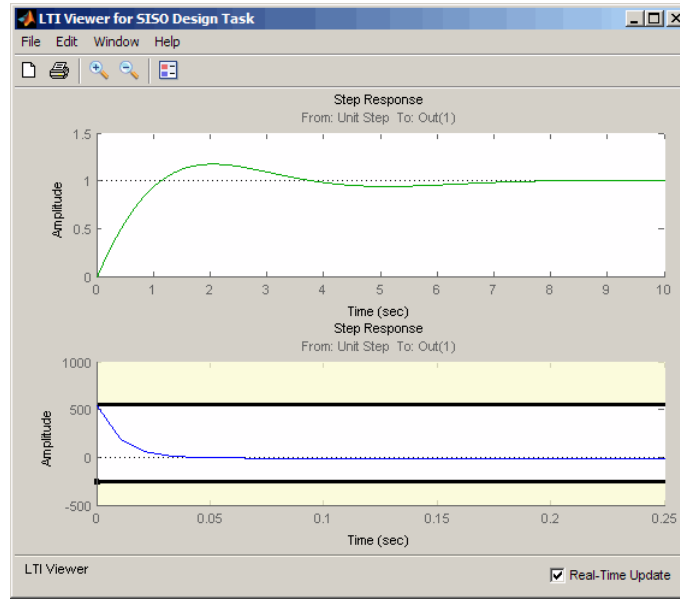
- At every optimization iteration, the optimization method reduces the distance between the current response and the upper and lower bounds on the signal.
- After the optimization completes, the **Optimization** tab resembles the next figure.



4 Examine the parameter values refined controller and the system's response:

a Examine the following response plots:

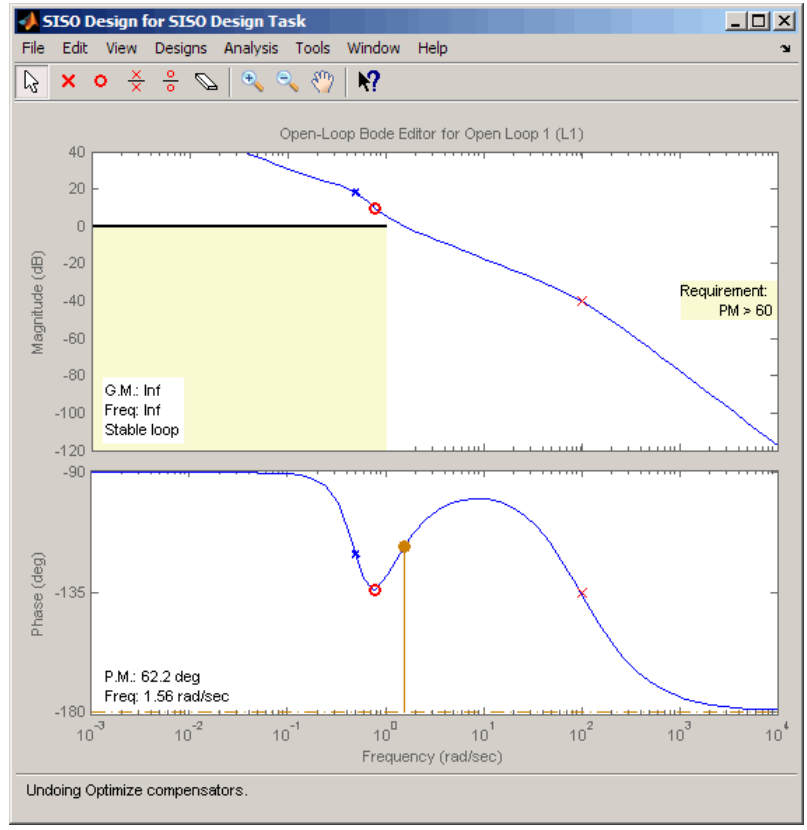
- The LTI viewer



The bottom plot shows that the controller output lies between 550 and -250, and thus meets the design requirement on the controller's output.

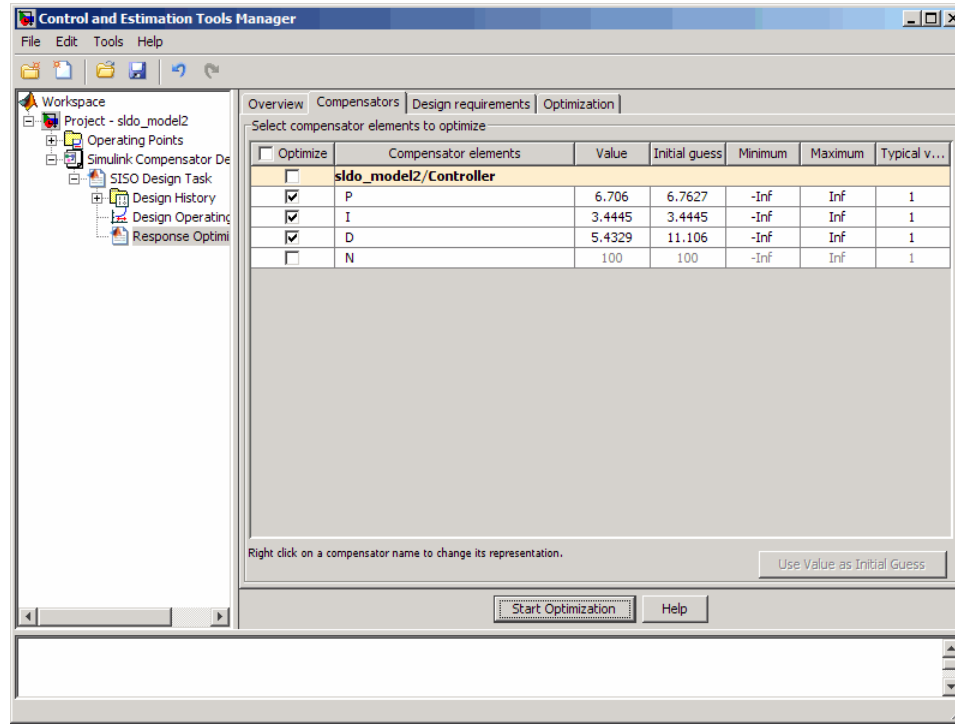
Note You must also check that the closed-loop response, shown in the top plot, remains stable after refining the controller design.

- The SISO Design window



The plots show that after refining the design, the system continues to meet the magnitude and phase margin requirements specified in “Design Requirements” on page 9-4.

b View the optimized controller parameter values in the **Value** field in the **Compensators** tab.



5 Select the **SISO Design Task** node, and click **Update Simulink Block Parameters**.

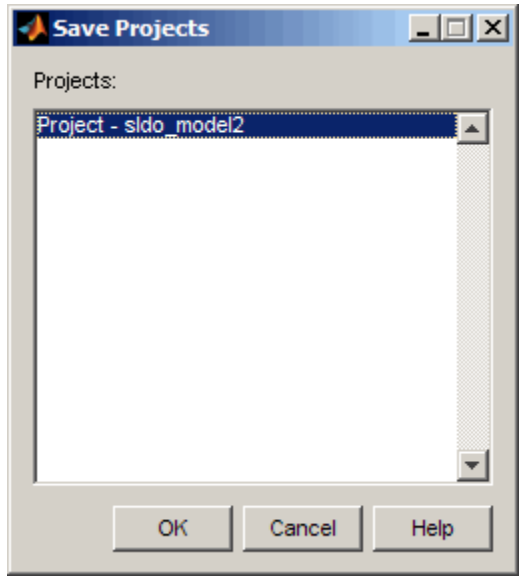
This action writes the optimized controller parameter values to the `Controller` block in the Simulink model.

Saving the Project

To save a project with the optimized controller parameters:

1. In Control and Estimation Tools Manager GUI, select **File > Save**.

   This action opens the Save Projects dialog box.



2. In the Save Projects dialog box, select **Project - sldo_model2**, and click **OK** to open the Save Projects window .

3. In the Save Projects window, enter `sldo_model2_optimized.mat` in the **File name** field, and click **Save**.

   The action saves the project as a MAT-file.

Tip You can reload this project by typing `sisotool('sldo_model2_optimized')` at the MATLAB prompt.

10

Tutorial — Modeling a System Using Adaptive Lookup Table

- "About This Tutorial" on page 10-2
- "Building a Model Using Adaptive Lookup Table Blocks" on page 10-4
- "Adapting the Lookup Table Values Using Time-Varying I/O Data" on page 10-16

About This Tutorial

In this section...

"Objectives" on page 10-2

"About the Data" on page 10-2

"Overview of Modeling a System Using Adaptive Lookup Table" on page 10-3

Objectives

In this tutorial, you learn how to capture the time-varying behavior of an engine using an n-D adaptive lookup table. You accomplish the following tasks using the Simulink software:

- Configure an adaptive lookup table block to model your system.
- Simulate the model to update the lookup table values dynamically.
- Export the adapted lookup table values to the MATLAB workspace.
- Disable the adaptation process and use the adaptive lookup table as a static lookup table.

About the Data

In this tutorial, you use the data in `vedata.mat` which contains the following variables measured from the engine:

- `X` — 10 input breakpoints for intake manifold pressure in the range [10,100]
- `Y` — 36 input breakpoints for engine speed in the range [0,7000]
- `Z` — 10x36 matrix of table data for engine volumetric efficiency

  To learn more about breakpoints and table data, see "Anatomy of a Lookup Table" in the Simulink documentation.

The output volumetric efficiency of the engine is time varying, and a function of two inputs—intake manifold pressure and engine speed. The data in the MAT-file is used to generate the time-varying input and output (I/O) data for the engine.

Overview of Modeling a System Using Adaptive Lookup Table

The process for modeling a system using adaptive lookup table consists of the following tasks:

- "Building a Model Using Adaptive Lookup Table Blocks" on page 10-4.
- "Adapting the Lookup Table Values Using Time-Varying I/O Data" on page 10-16.

Simulink Design Optimization software provides blocks for modeling systems as adaptive lookup tables. You build a model using the adaptive lookup table blocks, and then simulate the model to adapt the lookup table values to the time-varying I/O data. During simulation, the software uses the input data to locate the table values, and then uses the output data to recalculate the table values. The updated table values are stored in the adaptive lookup table block. To learn more about adaptive lookup tables, see "Adaptive Lookup Tables".

Building a Model Using Adaptive Lookup Table Blocks

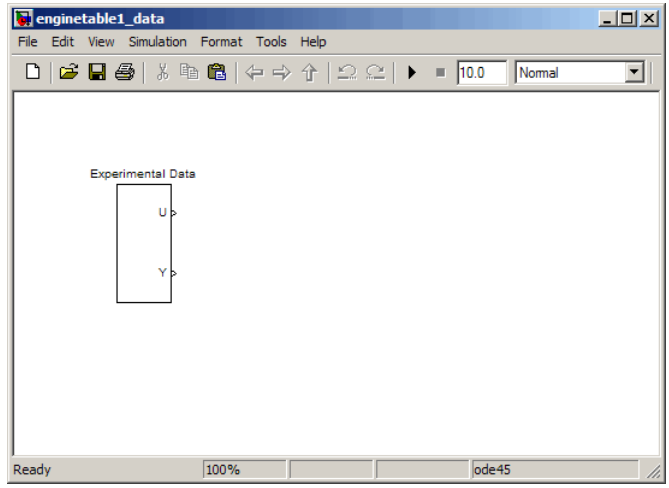
In this portion of the tutorial, you learn how to build a model of an engine using an Adaptive Lookup Table block.

1 Open a preconfigured Simulink model by typing the model name at the MATLAB prompt:

```
enginetable1_data
```

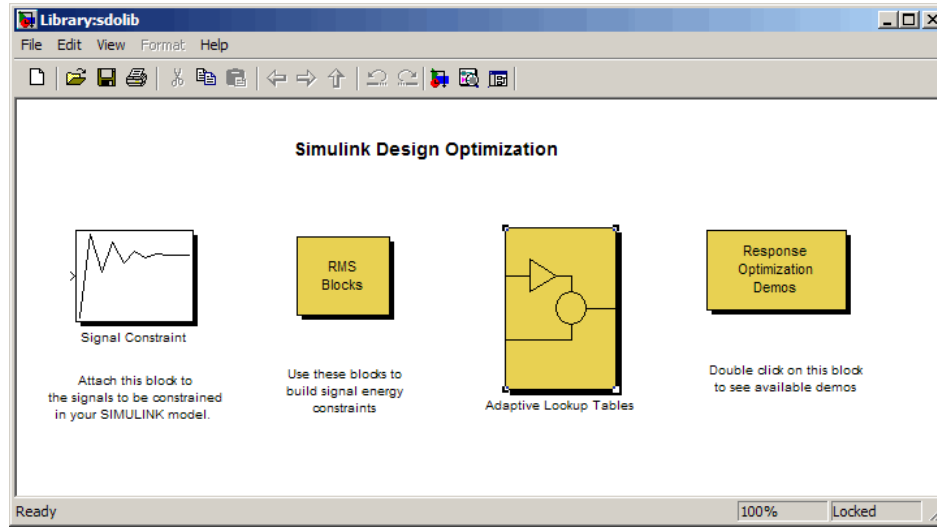
This command also loads the variables `X`, `Y` and `Z` into the MATLAB workspace. To learn more about this data, see “About the Data” on page 10-2.

The Experimental Data subsystem in the Simulink model generates the time-varying I/O data during simulation.

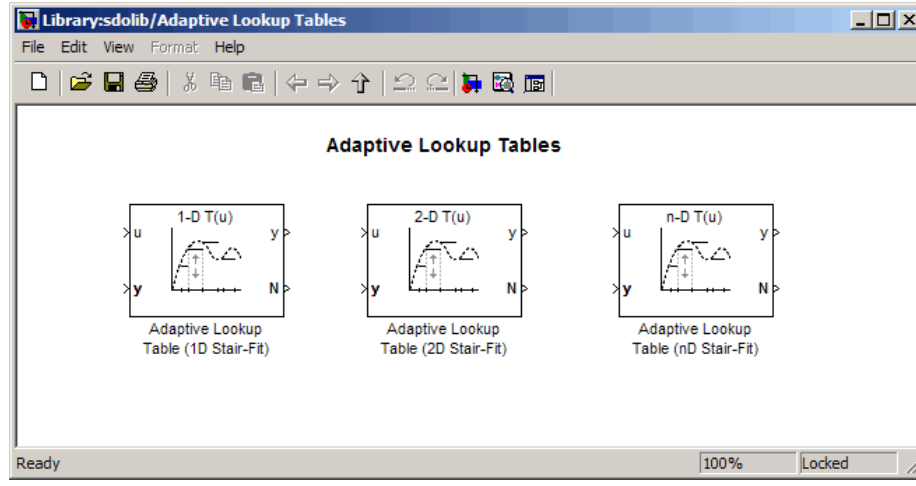


2 Open the Simulink Design Optimization library by typing the following command at the MATLAB prompt:

```
sdolib
```

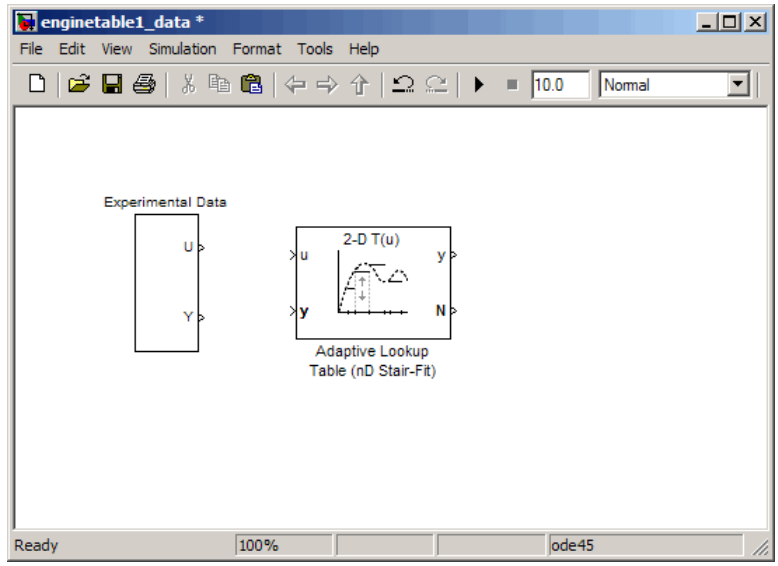


3 Double-click the Adaptive Lookup Tables block to open the Adaptive Lookup Tables library.

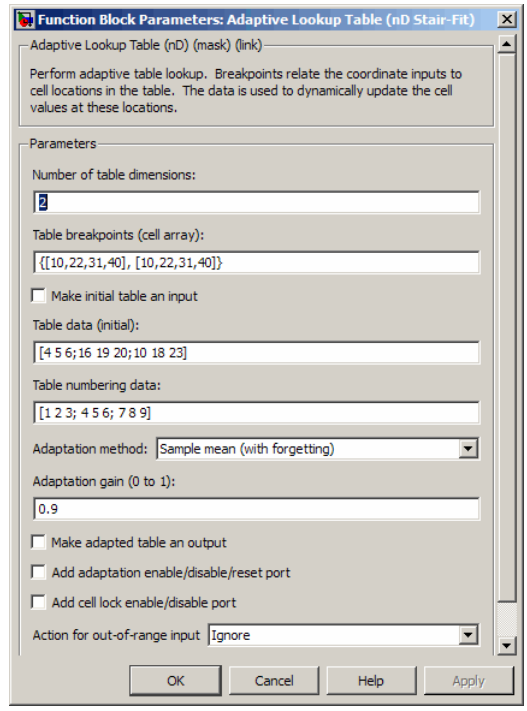


Simulink Design Optimization software provides three Adaptive Lookup Table blocks. In this tutorial, you use the Adaptive Lookup Table (nD Stair-Fit) block to model your system. To learn more about the blocks, see “Block Reference”.

4 Drag and drop the Adaptive Lookup Table (nD Stair-Fit) block from the Simulink Design Optimization library to the Simulink model window.



5 Double-click the Adaptive Lookup Table (nD Stair-Fit) block to open the Function Block Parameters: Adaptive Lookup Table (nD Stair-Fit) dialog box.



6 In the Function Block Parameters dialog box:

a Specify the following block parameters:

- **Table breakpoints (cell array)** — Enter `{[X; 110], [Y; 7200]}` to specify the range of input breakpoints.
- **Table data (initial)** — Enter `rand(10,36)` to specify random numbers as the initial table values for the volumetric efficiency.
- **Table numbering data** — Enter `reshape(1:360,10,36)` to specify a numbering scheme for the table cells.

b Verify that `Sample mean (with forgetting)` is selected in the **Adaptation method** drop-down list.

c Enter `0.98` in the **Adaptation gain (0 to 1)** field to specify the *forgetting factor* for the `Sample mean (with forgetting)` adaptation algorithm.

An adaptation gain close to 1 indicates high robustness of the lookup table values to input noise. To learn more about the adaptation gain, see "Sample Mean with Forgetting" in "Selecting an Adaptation Method".

d Select the **Make adapted table an output** check box.

This action adds a new port named `Tout` to the Adaptive Lookup Table block. You use this port to plot the table values as they are being adapted.

e Select the **Add adaptation enable/disable/reset port** check box.

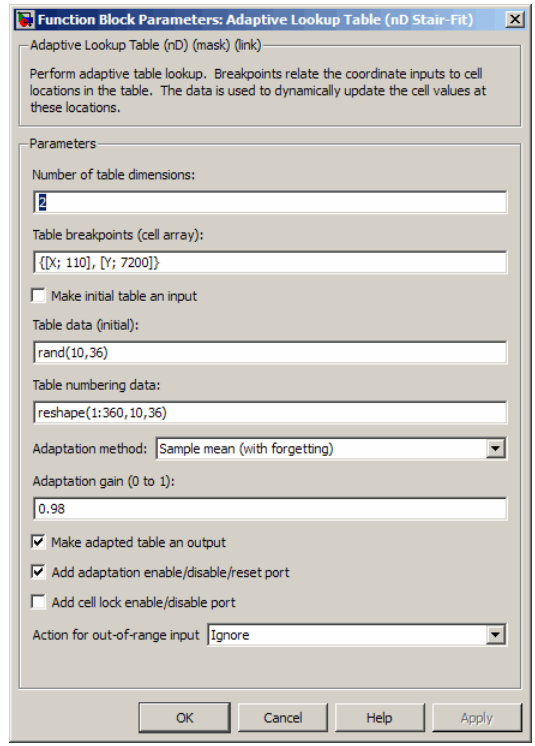
This action adds a new port named `Enable` to the Adaptive Lookup Table block. You use this port to enable or disable the adaptation process.

f Verify that `Ignore` is selected in the **Action for out-of-range** drop-down list.

This selection specifies that the software ignores any time-varying inputs outside the range of input breakpoints during adaptation.

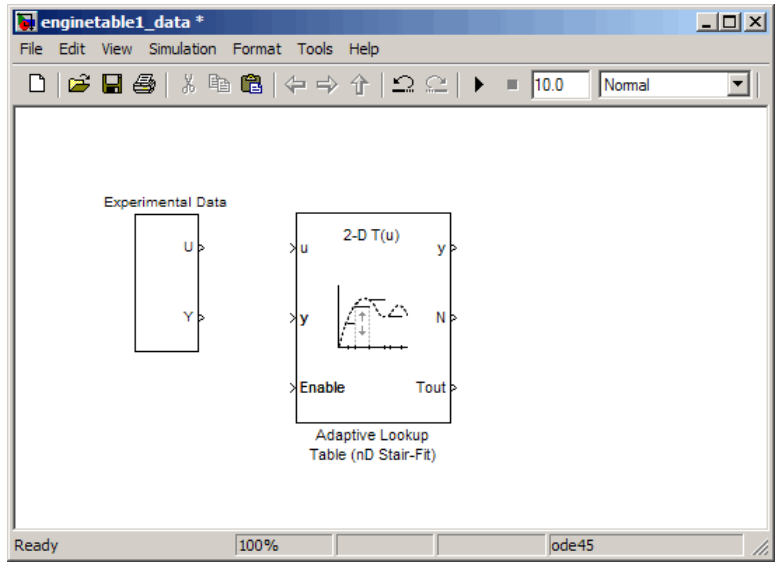
Tip To learn more about the Adaptive Lookup Table (nD Stair-Fit) block parameters, see the Adaptive Lookup Table (nD Stair-Fit) block reference page.

After you configure the parameters, the block parameters dialog box looks like the following figure.

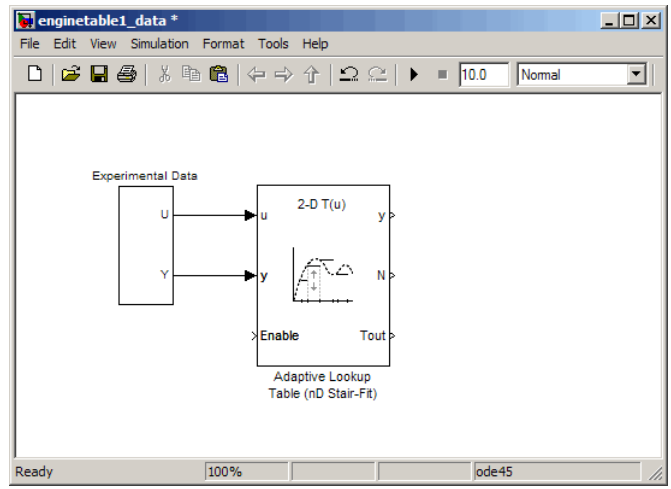


7 Click **OK** to close the Function Block Parameters dialog box.

The Simulink model now looks similar to the following figure.



8 Assign the input and output data to the engine model by connecting the `U` and `Y` ports of the Experimental Data block to the `u` and `y` ports of the Adaptive Lookup Table block, respectively.



Tip To learn how to connect blocks in the Simulink model window, see "Connecting Blocks in the Model Window" in the Simulink documentation.

9 Design a logic that enables or disables the adaptation process:

a Open the Simulink Library Browser by typing `simulink` at the MATLAB prompt.

b In the **Libraries** area of the Simulink Library Browser window, select **Commonly Used Blocks**.

c Drag the following blocks to the Simulink model window:

- Constant block
- Logical Operator block

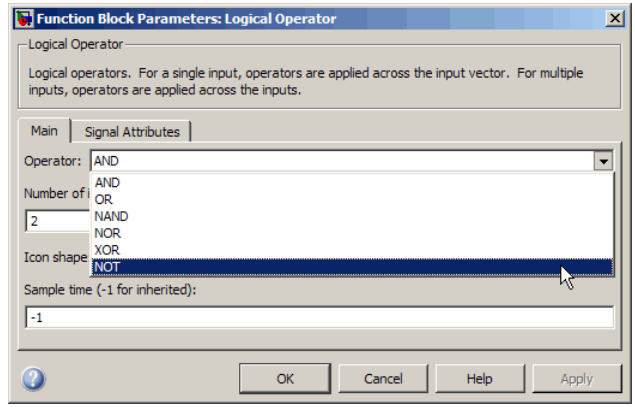
To learn more about the blocks, see the Constant, and Logical Operator block reference pages in the Simulink documentation.

d In the **Libraries** area of the Simulink Library Browser window, select **Signal Routing**, and drag the Manual Switch block to the Simulink model window.

To learn more about the block, see the Manual Switch block reference page in the Simulink documentation.

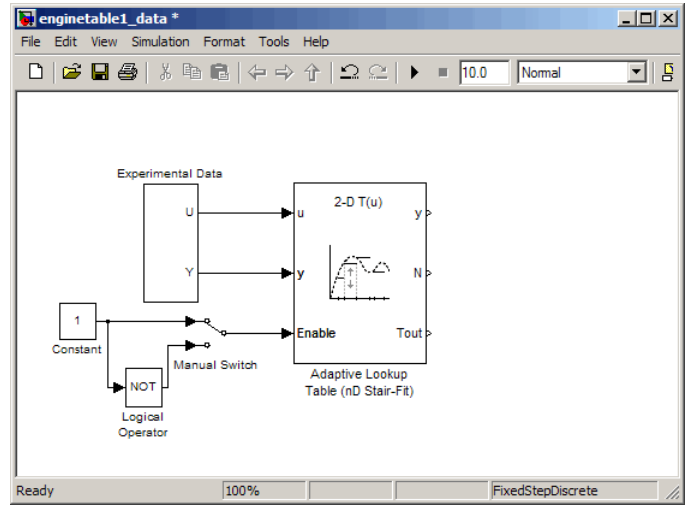
e Double-click the Logical Operator block to open the Function Block Parameters: Logical Operator dialog box.

f Select `NOT` from the **Operator** drop-down list.



g Click **OK**.

h Connect the blocks to create a logic, as shown in the next figure.



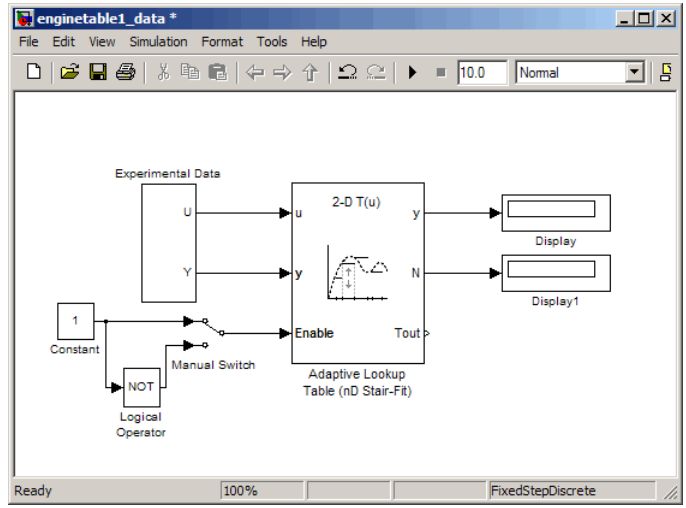
This logic outputs an initial value of 1 which enables the adaptation process.

10 Add Display blocks to view the output of the adaptation process:

a In the **Libraries** area of the Simulink Library Browser window, select **Sinks**, and drag the Display block to the Simulink model window.

b Drag another Display block from the Simulink Library Browser window to the model window.

c Connect the Display blocks to the `y` and `N` ports of the Adaptive Lookup Table block, as shown in the next figure.



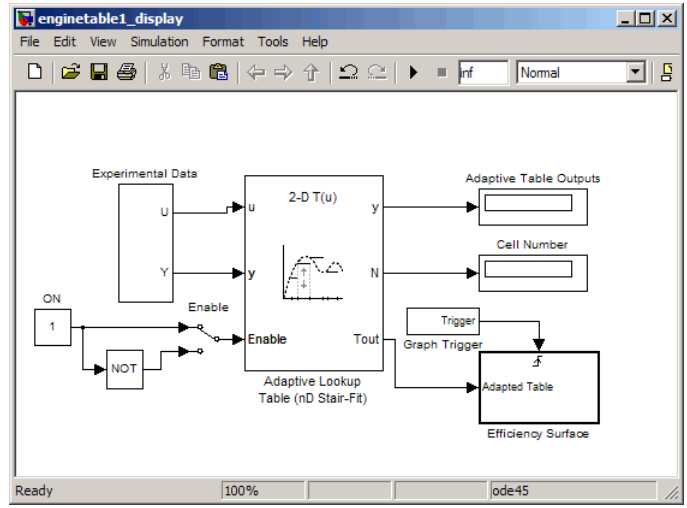
During simulation, the Display blocks show the following:

- `Display` block — Shows the value of the current cell being adapted.
- `Display1` block — Shows the number of the current cell being adapted.

11 Open a preconfigured Simulink model that already contains a plot to display the lookup table values as they adapt during simulation. To do so, type the following model name at the MATLAB prompt:

```
enginetable1
```

The model opens, as shown in the next figure.



This model already contains a subsystem named `Efficiency Surface` that generates a plot of the lookup table values as they adapt during simulation.

Tip You can also write your own MATLAB function to plot the adapted table values during simulation.

You have now configured the adaptive lookup table parameters and built the model for updating and viewing the adaptive lookup table values. You must now simulate the model to start the adaptation, as described in "Adapting the Lookup Table Values Using Time-Varying I/O Data" on page 10-16.

Adapting the Lookup Table Values Using Time-Varying I/O Data

In this portion of the tutorial, you learn how to update the lookup table values to adapt to the time-varying input and output values.

You must have already built the Simulink model, as described "Building a Model Using Adaptive Lookup Table Blocks" on page 10-4. To open a saved model that already contains the block parameters and display blocks, type the following model name at the MATLAB prompt:

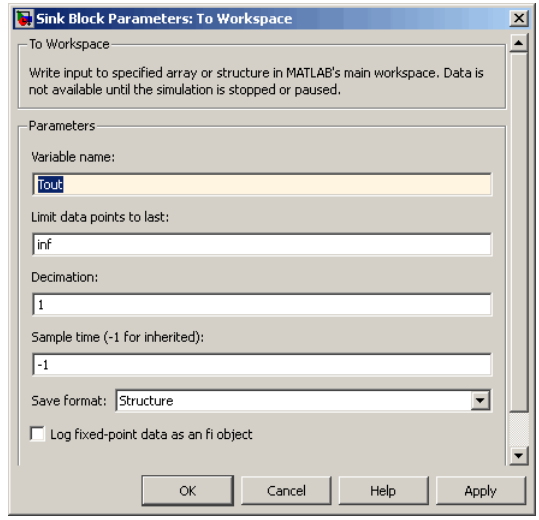
```
enginetable1
```

To perform the adaptation:

1. Connect a To Workspace Simulink block to export the adapted table values:
   a. Open the Simulink Library Browser, if it is not already open, by typing `simulink` at the MATLAB prompt.
   b. In the **Libraries** area of the Simulink Library Browser window, select **Sinks**, and drag the To Workspace block to the `enginetable1` Simulink model window.

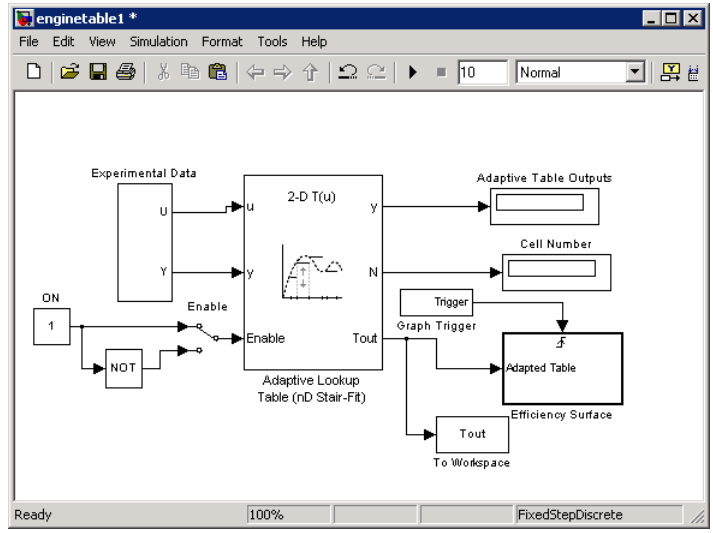
      To learn more about this block, see the To Workspace block reference page in the Simulink documentation.

c Double-click the To Workspace block to open the Sink Block Parameters dialog box, and type `Tout` in the **Variable name** field.

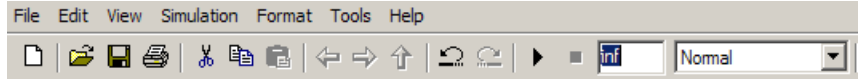


d Click **OK**.

e Connect the To Workspace block to the adaptive lookup table output signal `Tout`, as shown in the next figure.



2 In the Simulink model window, enter `inf` as the simulation time.

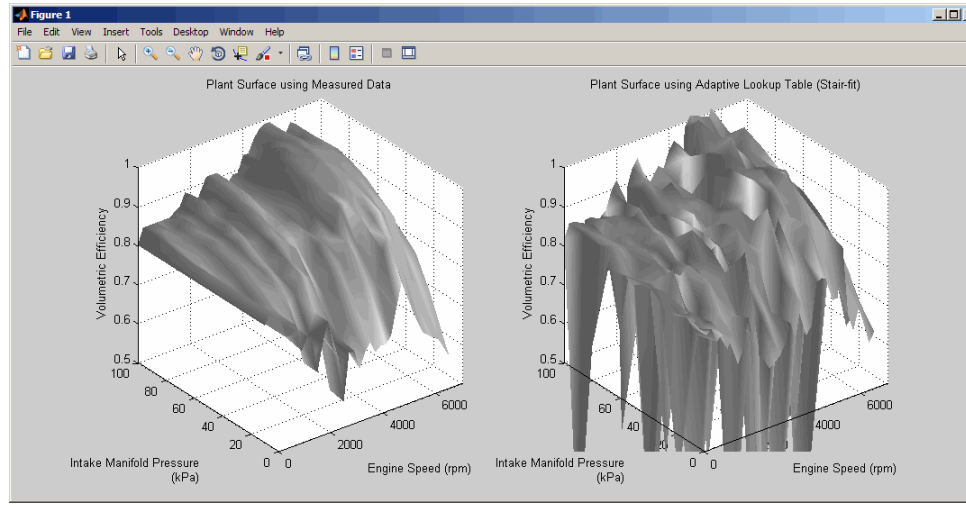


The simulation time of infinity specifies that the adaptation process continues as long as the input and output values of the engine change.

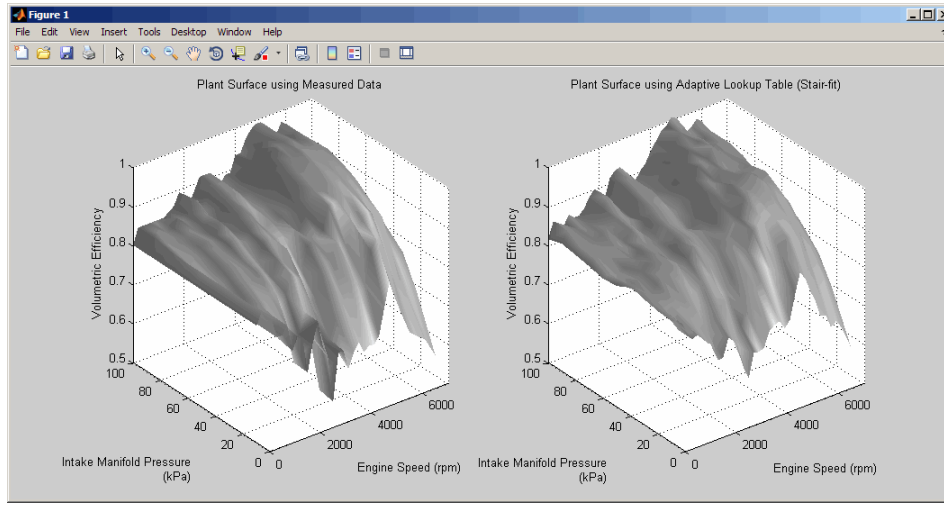
3 In the Simulink model window, select **Simulation > Start** to start the adaptation process.

A figure window opens that shows the volumetric efficiency of the engine as a function of the intake manifold pressure and engine speed:

- The left plot shows the measured volumetric efficiency as a function of intake manifold pressure and engine speed.
- The right plot shows the volumetric efficiency as it adapts with the time-varying intake manifold pressure and engine speed.



During simulation, the lookup table values displayed on the right plot adapt to the variations in the I/O data. The left and the right plots resemble each other after a few seconds, as shown in the next figure.



Tip During simulation, the `Cell Number` and `Adaptive Table Outputs` blocks in the Simulink model display the cell number, and the adapted lookup table value in the cell, respectively.

4 Examine that the left and the right plots match. This resemblance indicates that the table values have adapted to the time-varying I/O data.

5 Pause the simulation by selecting **Simulation > Pause**.

This action also exports the adapted table values `Tout` to the MATLAB workspace.

Note After you pause the simulation, the adapted table values are stored in the Adaptive Lookup Table block.

6 After the table values adapt to the time-varying I/O data, you can continue to use the Adaptive Lookup Table block as a static lookup table:

a In the Simulink model window, double-click the Manual Switch block.

This action toggles the switch to , and disables the adaptation.

b Select **Simulation > Start** to restart the simulation.

During simulation, the Adaptive Lookup Table block works like a static lookup table, and continues to estimate the output values as the input values change. You can see the current lookup table value in the `Adaptive Table Outputs` block in the Simulink model window.

Note After you disable the adaptation, the Adaptive Lookup Table block does not update the stored table values, and the figure that displays the table values does not update.

A

Examples

Use this list to find examples in the documentation.

Getting Started

Chapter 5, “Tutorial — Preparing Data for Parameter Estimation Using the GUI”
Chapter 6, “Tutorial — Estimating Parameters from Measured Data Using the GUI”
Chapter 7, “Tutorial — Optimizing Parameters to Meet Time-Domain Requirements Using the GUI”
Chapter 8, “Tutorial — Optimizing Parameters to Meet Time-Domain Requirements Using the Command Line”
Chapter 9, “Tutorial — Designing a PID Controller Using Optimization-Based Tuning”
Chapter 10, “Tutorial — Modeling a System Using Adaptive Lookup Table”

Index

A

add Signal Constraint block 7-5

B

block parameters window 7-10

C

configure for parameter optimization 7-5
configuring
    control design 9-5
Control and Estimation Tools Manager 9-5

D

design requirement line segments 7-8
design requirements
    time-domain 7-4
designing optimization-based linear
    controller 9-25

E

Edit Design Requirement dialog box
    rows 7-11

F

`findconstr` function
    example 8-8
`findpar` function
    example 8-11

L

linear control design 9-5

N

`newsro` function
    example 8-8

O

optimization-based control design 9-1
`optimize` function
    example 8-12

P

parameter settings 7-20
plots 9-15

R

response optimization
    tutorial using functions 8-1
    tutorial using GUI 7-1

S

signal constraints
    objects 8-8
Simulink Design Optimization product
    related products 1-5
    required products 1-5
SISO Design Task 9-5
specify reference signal using GUI 7-25
specifying controller parameters 9-11
specifying design requirements 9-15

T

time-domain design requirements 8-4
track reference signal 8-15