

# Simulink<sup>®</sup> Control Design

For Use with Simulink<sup>®</sup>

■ Modeling

■ Simulation

■ Implementation

Advanced Topics

*Version 1*



## How to Contact The MathWorks:



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



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



508-647-7000      Phone



508-647-7001      Fax



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

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

### *Simulink Control Design Advanced Topics*

© COPYRIGHT 2004 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.

MATLAB, Simulink, Stateflow, Handle Graphics, and Real-Time Workshop are registered trademarks, and TargetBox is a trademark of The MathWorks, Inc.

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

Printing History: June 2004      Online only      New for Version 1.0 (Release 14)

## Understanding and Controlling Results

1

<b>Comparing the Linearized and Original Models</b> .....	<b>1-2</b>
<b>Linearization Algorithms</b> .....	<b>1-9</b>
<b>Block-by-Block Analytic Linearization</b> .....	<b>1-11</b>
Individual Block Linearization Methods .....	<b>1-11</b>
<b>Numerical-Perturbation Linearization</b> .....	<b>1-16</b>
Invoking Numerical-Perturbation Linearization .....	<b>1-16</b>
Perturbation Algorithm .....	<b>1-17</b>
Controlling the Results of Numerical-Perturbation Linearization .....	<b>1-19</b>



# Understanding and Controlling Results

---

To create accurate linearized models, it is important to be able to interpret the results and to understand the linearization algorithms. One method of interpreting the results is by simulating the linearized model and comparing the output with the original model. The linearization algorithms can be adjusted in various ways to control these results, as outlined in this chapter.

Comparing the Linearized and Original Models (p. 1-2)

Methods for simulating the linearized model and comparing the results to the original model.

Linearization Algorithms (p. 1-9)

Brief introduction to the two main linearization methods with advantages and disadvantages of each.

Block-by-Block Analytic Linearization (p. 1-11)

Description of the default linearization method with suggestions for controlling the results.

Numerical-Perturbation Linearization (p. 1-16)

Description of an alternative linearization method with suggestions for controlling the results.

## Comparing the Linearized and Original Models

Comparing simulations of the original model with simulations of the linearized model helps to determine if the linearized system behaves in a similar way to the original model. When comparing models, remember that the states, inputs, and outputs of the linearized model are defined about an operating point of the original model, using the following variables:

$$\begin{aligned}\delta x(t) &= x(t) - x_0 \\ \delta u(t) &= u(t) - u_0 \\ \delta y(t) &= y(t) - y_0\end{aligned}$$

This means that when the original model is at the operating point  $x(t)=x_0$ ,  $u(t)=u_0$ ,  $y(t)=y_0$ , the linearized model will be at the operating point  $\delta x(t)=0$ ,  $\delta u(t)=0$ ,  $\delta y(t)=0$ . To compare the models accurately, subtract  $u_0$  from input values and  $x_0$  from the initial state values in the linearized model, then add  $y_0$  to the output signal.

When you linearize only a portion of the original model, you should simulate the linearized model by substituting it back into the model in place of the original portion. This ensures that the operating point and inputs to the linearized portion are correct. To do this, export the linearized model to the workspace, delete the original portion from the model, and replace it with an LTI System block based on the linearized model.

### Example

This example compares the `magball` model with the linearized model computed in “Linearizing the Model” on page 3-29:

- 1 If you have not done so already, linearize the `magball` model at the targeted operating point computed in “Computing Operating Points from Specifications” on page 3-21.
- 2 To create a new model containing the linearized plant system, first export the linearized model and operating point from the Control and Estimation Tools Manager to the MATLAB® workspace. To do this, right click the linearized model name in the project tree of the Control and Estimation

Tools Manager. Select **Export** from the menu. Accept the default name for the model, `Model_sys`, and for the operating point, `Model_op`.

Then, create a new Simulink® model, `magball_lin`, which is a copy of the original model, `magball`. Replace the Magnetic Ball Plant subsystem in `magball_lin` with an LTI System block (located in the Control System Toolbox category of the Simulink Library Browser). Import the linearized model into this block by entering `Model_sys` in the **LTI system variable** field in the Block Parameters window.

- 3 Set the operating points of the models by specifying the initial values of the states in the models:

- a To set the initial values for `magball`, first enter the following at the command line

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

This returns vectors of state values and input values from the object `Model_op`.

```
x =  
    0.0500  
     0  
   -0.0000  
    7.0036  
     0  
u =  
 []
```

The ordering of states in these vectors is the same as that used in the Simulink model. To use the values in the state vector, `x`, as initial values for the model, select **Simulation -> Configuration Parameters** in the `magball` model window, then select the **Data Import/Export** panel. Select the check box next to **Initial State** and enter `x` on the right. Click **OK**.

- b In `magball_lin`, the operating point values for the linearized system will all be zero since this subsystem was linearized about the operating point values. The operating point values in the Controller will be the same as in the original model since the Controller was not linearized. To create a

vector of initial state values with the correct state ordering, first create a new operating point object for the system by typing

```
op=operpoint('magball_lin')
```

Change the operating point for the Controller in op to be the same as those in Model\_op.

```
op.States(1).x=Model_op.States(1).x
```

This returns the following operating point.

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

```
States:
```

```
-----
```

```
(1.) magball_lin/Controller/Controller
```

```
  x: 0
```

```
  x: -2.56e-006
```

```
(2.) magball_lin/LTI System/Internal
```

```
  x: 0
```

```
  x: 0
```

```
  x: 0
```

```
Inputs: None
```

Keep the operating point for the LTI system as zero. Extract vectors of states and inputs from this edited operating point.

```
[x1,u1]=getxu(op)
```

which returns

```
x1 =  
  1.0e-005 *  
      0  
      0  
      0  
      0  
 -0.2556
```

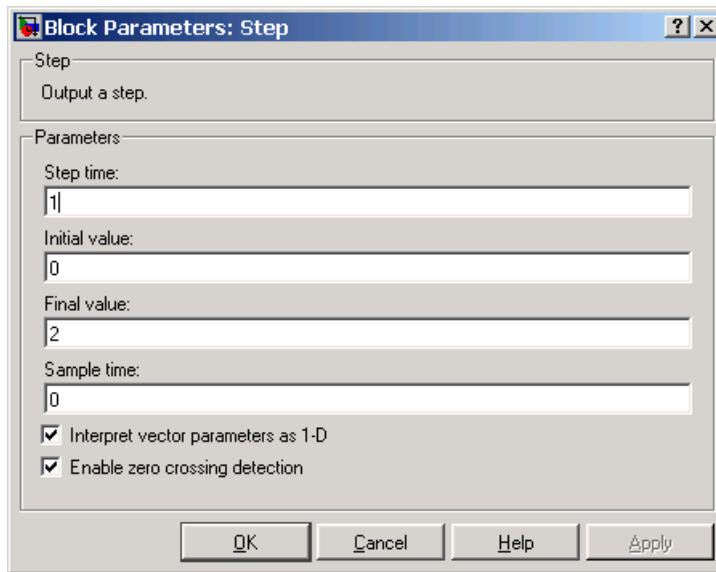
```
u1 =  
 []
```

- c To use the values in the state vector, x1, as initial values for magball\_lin, select **Simulation -> Configuration Parameters** in the



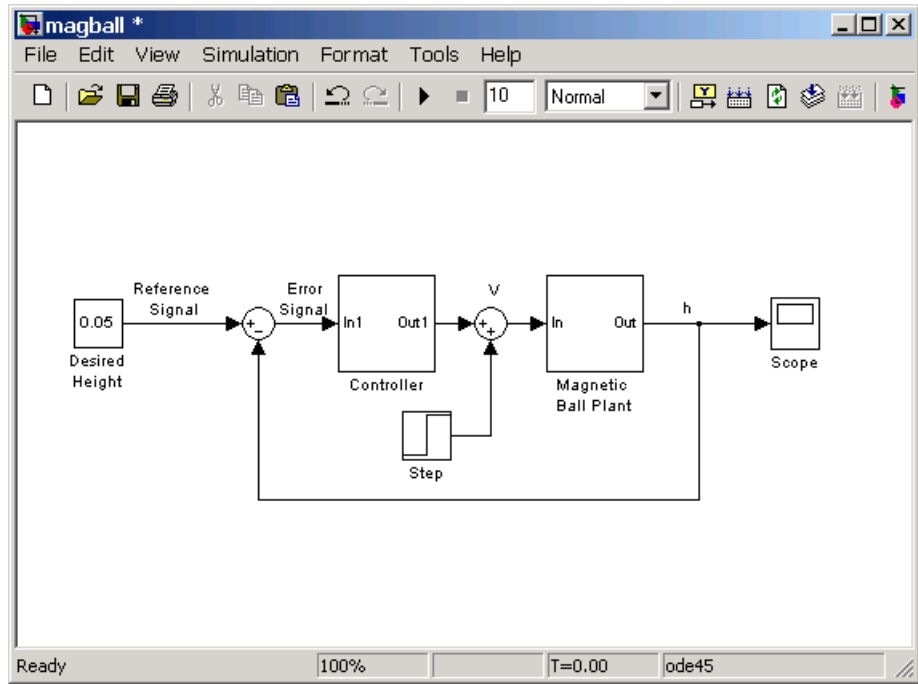
magball\_lin model window, then select the **Data I/O** panel. Select the check box next to **Initial State** and enter x1 on the right. Click **OK**

- 4 The output of magball\_lin will be zero at the operating point. To create an output signal that is comparable with that in magball, add a Constant block, with a value of 0.05 to the output of magball\_lin. Similarly, the input to magball\_lin should be zero at the operating point. This is achieved by subtracting a value of 14 from the input signal of the linearized system. The operating point values, 0.05 and 14, were found using a Scope block to measure steady-state signal levels in the original model.
- 5 To observe the response of the models to a perturbation, add a Step block with the following parameter values to the input to the plant in both models.

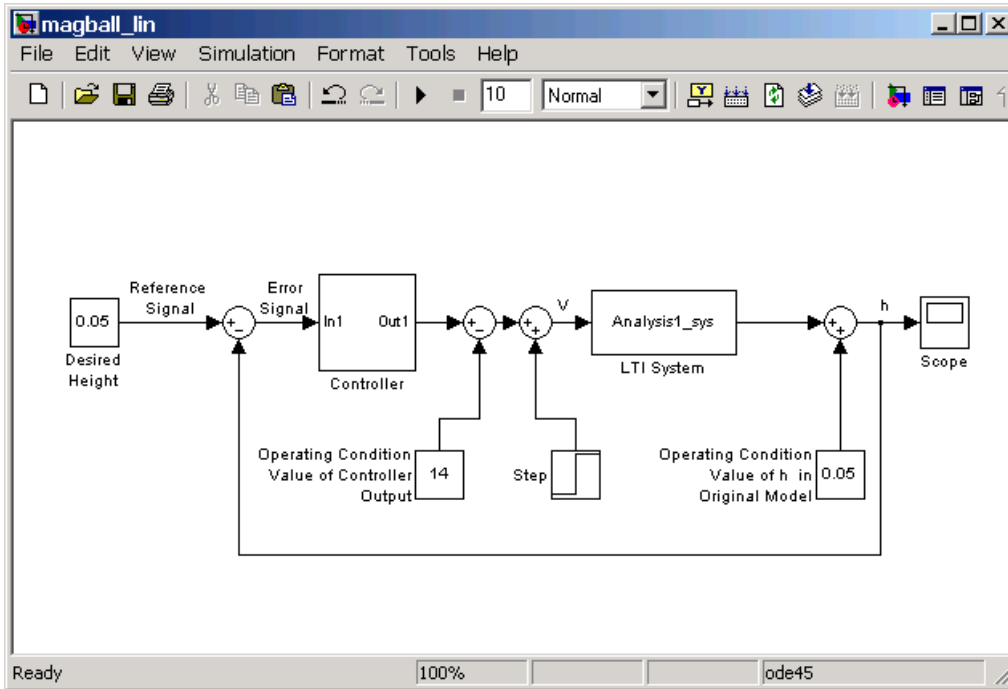


**Figure 1-1: Parameter Values for Step Block**

The model diagrams should now look like those in the following figures.

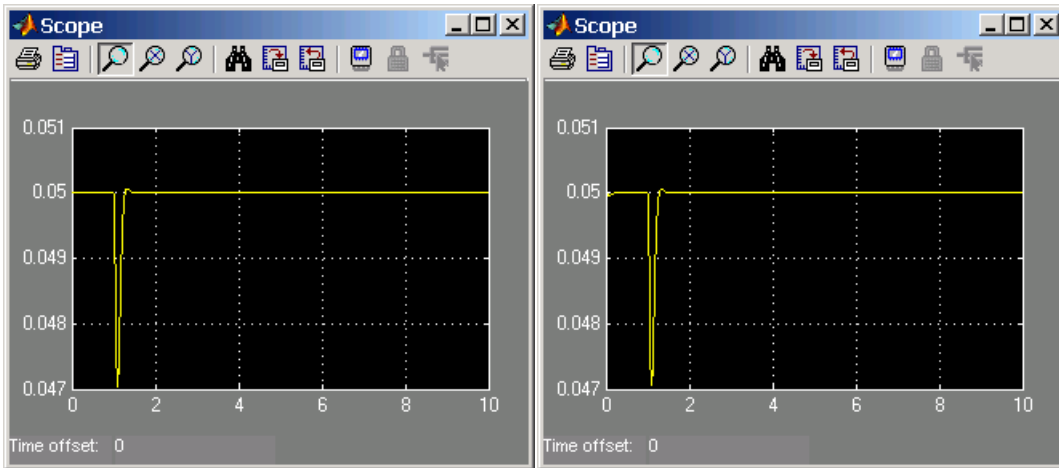


**Figure 1-2: Magball Model with a Step Block Added to the Input**



**Figure 1-3: Magball Model with Linearized Magnetic Ball Plant**

- 6 Run simulations in both models. The output signals, in the Scope blocks, are shown in the following figure.



**Figure 1-4: Scope Blocks from Original (left) and Linearized (right) Models**

As shown in the figure, both the original and linearized models react to the step input in a similar way.

## Linearization Algorithms

Simulink Control Design can use two different linearization methods. The default method, which is used unless an option is selected, is called block-by-block analytic linearization. To use the alternative method, numerical-perturbation linearization, you must select an option in the Linearization Options dialog box of the GUI, or if using functions, with the `linoptions` function. The remainder of this chapter describes the two linearization methods in more detail and provides suggestions for controlling the results to create more accurate linearized models.

The default linearization method, block-by-block analytic linearization, linearizes the blocks individually and then combines the results to produce the linearization of the whole system. This method has several advantages:

- It divides the linearization problem into several smaller, easier problems.
- It defines the system being linearized by input and output markers on the signal lines rather than root-level inport and outport blocks.
- It supports open loop analysis.
- You can control the linearization of each block by using an analytic linearization that is programmed into the block or by selecting a perturbation level for the block.

The main disadvantage of this method is that for large or complicated systems it might be slower than numerical-perturbation linearization.

Alternatively, numerical-perturbation linearization linearizes the *whole system* by numerically perturbing the system's inputs and states about the operating point. This method has the advantage that it is quick and simple, especially for large or complicated systems. However, there are also several disadvantages with this method:

- It relies on root-level inport and outport blocks to define the system being linearized.
- There is no support for open loop analysis.
- You have limited control over the perturbation levels for each block.
- It does not use any of the analytic, pre-programmed block linearizations.
- It is sensitive to scaling issues (models with large and small signal values).

“Block-by-Block Analytic Linearization” on page 1-11 and  
“Numerical-Perturbation Linearization” on page 1-16 discuss these methods  
further.

## Block-by-Block Analytic Linearization

Block-by-block analytic linearization is the default linearization method in Simulink Control Design. In this method, each of the blocks within the linearization path is first linearized individually. The linearization of the whole system is then computed by combining these results using the algorithm discussed in “Numerical-Perturbation Linearization” on page 1-16. This approach breaks the problem into several smaller problems. The following section gives details of the methods used to linearize each block, with suggestions for controlling the linearizations to create more accurate linearized models.

### Individual Block Linearization Methods

There are two methods that Simulink Control Design uses to linearize the individual blocks in a model. Each method has options that you can control to create accurate linearized models.

#### Analytic Linearization

Several nonlinear Simulink blocks contain analytic linearized models. When linearizing a system using block-by-block analytic linearization, you can use these analytic linearizations instead of numerically perturbing the block. This is especially useful for blocks that contain discontinuities and do not give good results using numerical perturbation.

The following blocks contain analytic linearizations. For more information see the reference page for each block.

- Abs
- Dead Zone
- Gain
- Memory
- Quantizer
- Rate limiter
- Saturation
- Transport Delay
- Variable Transport Delay

Options to control the linearizations of these blocks include adjusting any optional linearization parameters in the Block Parameters window (such as the order of the Padé approximation in the Transport Delay block) or choosing not to use the preprogrammed linearization. For example, do not select the **Treat as gain when linearizing** option in the Saturation block.

---

**Note** The preprogrammed, analytic block linearizations are only used in block-by-block analytic linearization. When using numerical-perturbation linearization, these blocks will be numerically perturbed along with the rest of the system.

---

### Block Perturbation

When a preprogrammed block linearization cannot be used, Simulink Control Design will compute the block linearization by numerically perturbing the states and inputs of the block about the operating point of the block. As opposed to the numerical-perturbation linearization method, this perturbation is local and its propagation through the rest of the model is restricted.

The block perturbation algorithm involves introducing a small perturbation to the nonlinear block and measuring the response to this perturbation. Both the perturbation and the response are used to create the matrices in the linear state-space model of this block. Changing the size of the perturbations will change the resulting linearized model.

As described in “Linearization of Nonlinear Models” on page 2-5, a nonlinear Simulink block can be written as a state-space system:

$$\begin{aligned}\dot{x}(t) &= f(x(t), u(t), t) \\ y(t) &= g(x(t), u(t), t)\end{aligned}$$

In these equations,  $x(t)$  represents the states of the block,  $u(t)$  represents the inputs of the block, and  $y(t)$  represents the outputs of the block.

A linearized model of this system is valid in a small region around the operating point  $t=t_0$ ,  $x(t_0)=x_0$ ,  $u(t_0)=u_0$ , and  $y(t_0)=g(x_0, u_0, t_0)=y_0$ . Subtracting the operating point values from the states, inputs, and outputs defines a set of variables centered about the operating point:



$$\begin{aligned}\delta x(t) &= x(t) - x_o \\ \delta u(t) &= u(t) - u_o \\ \delta y(t) &= y(t) - y_o\end{aligned}$$

The linearized model can be written in terms of these new variables and is usually valid when these variables are small, i.e. when the departure from the operating point is small:

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

The state-space matrices  $A$ ,  $B$ ,  $C$ , and  $D$  of this linearized model represent the Jacobians of the block, as defined in “Linearization of Nonlinear Models” on page 2-5. To compute the matrices, the states and inputs are perturbed, one at a time, and the response of the system to this perturbation is measured by computing  $\delta \dot{x}$  and  $\delta y$ . The perturbation and response are then used to compute the matrices in the following way

$$\begin{aligned}A(:,i) &= \frac{\dot{x}|_{x_{p,i}} - \dot{x}_o}{x_{p,i} - x_o}, & B(:,i) &= \frac{\dot{x}|_{u_{p,i}} - \dot{x}_o}{u_{p,i} - u_o} \\ C(:,i) &= \frac{y|_{x_{p,i}} - y_o}{x_{p,i} - x_o}, & D(:,i) &= \frac{y|_{u_{p,i}} - y_o}{u_{p,i} - u_o}\end{aligned}$$

where

- $x_{p,i}$  is the state vector whose  $i$ th component is perturbed from the operating point value.
- $x_o$  is the state vector at the operating point.
- $u_{p,i}$  is the input vector whose  $i$ th component is perturbed from the operating point value.
- $u_o$  is the input vector at the operating point.
- $\dot{x}|_{x_{p,i}}$  is the value of  $\dot{x}$  at  $x_{p,i}, u_o$ .
- $\dot{x}|_{u_{p,i}}$  is the value of  $\dot{x}$  at  $u_{p,i}, x_o$ .
- $\dot{x}_o$  is the value of  $\dot{x}$  at the operating point.

- $y|_{x_{p,i}}$  is the value of  $y$  at  $x_{p,i}, u_o$ .
- $y|_{u_{p,i}}$  is the value of  $y$  at  $u_{p,i}, x_o$ .
- $y_o$  is the value of  $y$  at the operating point.

Linearized models of discrete-time or multi-rate blocks are computed in a similar way. See “Linearization of Discrete-Time Models” on page 2-7 and “Linearization of Multi-Rate Models” on page 2-8 for the equations of linearized discrete-time and multi-rate systems.

---

**Note** A perturbed value is one that has been changed by a very small amount from the operating point value. The default difference between the perturbed value and the operating point value is  $10^{-5}(1 + |x|)$  for block-by-block analytic linearization, where  $x$  is the operating point value.

---

Changing the size of the perturbations will change the linearization results. The default perturbation size is  $10^{-5}(1 + |x|)$  where  $x$  is the operating point value of the state or input being perturbed. To change the perturbation size of the states in the Magnetic Ball Plant block in the magball model to  $10^{-7}(1 + |x|)$ , type

```
blockname='magball/Magnetic Ball Plant'  
set_param(blockname, 'StatePerturbationForJacobian', '1e-7')
```

To change the perturbation size of the input of the Magnetic Ball Plant block to  $10^{-7}(1 + |u|)$ , where  $u$  is the input signal level, follow these steps:

- 1** Get the block’s port handles

```
ph=get_param('magball/Magnetic Ball Plant', 'PortHandles')
```

- 2** Get the inport

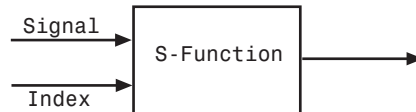
```
pin=ph.Inport(1)
```

- 3** Set the perturbation level for this inport

```
set_param(pin, 'PerturbationForJacobian', '1e-7')
```

If there is more than one inport, you can choose to assign a different perturbation level to each. The following figure shows an S-Function block with

two input signals, the actual signal and an index variable. Since you probably do not want to perturb the index signal, you can assign a perturbation level of zero to this input.



**Figure 1-5: Block Containing Two Inputs**

## Numerical-Perturbation Linearization

An alternative linearization method available for use in Simulink Control Design is numerical-perturbation linearization, which computes state-space matrices for the linearized model by numerical perturbation of the *whole system*. The method is relatively quick and simple, although as mentioned in “Linearization Algorithms” on page 1-9, it does have some disadvantages.

Numerical-perturbation linearization requires that root-level inport and outport blocks be present in the model. These blocks define the portion of the model that you want to linearize instead of inserting input and output points by right-clicking on the signal lines. Any input, output, or open loop points on signal lines in the model will be ignored when using numerical-perturbation linearization.

The perturbation is introduced to the system at the root level inport blocks and in the states of the system. The response to the perturbation is measured at the outport blocks. Suggestions for controlling the results of numerical-perturbation linearization to create accurate linearized models are given in “Controlling the Results of Numerical-Perturbation Linearization” on page 1-19

### Invoking Numerical-Perturbation Linearization

Prior to Simulink 3.0, numerical-perturbation linearization was the only linearization method available with Simulink. Although block-by-block analytic linearization is now the default linearization method, you might choose to use numerical-perturbation linearization if your model is very large or complicated.

To use numerical-perturbation linearization with the Simulink Control Design GUI, select **Tools -> Options** while in the **Linearizations** node of the Control and Estimation Tools Manager and select Numerical-Perturbation from the **Linearization Algorithms** menu.

To use numerical-perturbation linearization with the `linearize` function, set the `LinearizationAlgorithm` option to 'numericalpert' with the `linoptions` function.

```
linopt=linoptions('LinearizationAlgorithm','numericalpert')
```

To linearize the model, type

```
sys=linearize('modelName',op,linopt)
```

where `modelName` is the name of the model being linearized and `op` is the operating point object for the system.

## Perturbation Algorithm

The numerical perturbation algorithm involves introducing a small perturbation to the nonlinear model and measuring the response to this perturbation. Both the perturbation and the response are used to create the matrices in the linear state-space model. Changing the size of the perturbations will change the resulting linearized model.

As described in “Linearization of Nonlinear Models” on page 2-5, a nonlinear Simulink model can be written as a state-space system:

$$\begin{aligned}\dot{x}(t) &= f(x(t), u(t), t) \\ y(t) &= g(x(t), u(t), t)\end{aligned}$$

In these equations,  $x(t)$  represents the states of the model,  $u(t)$  represents the inputs of the model, and  $y(t)$  represents the outputs of the model.

A linearized model of this system is valid in a small region around the operating point  $t=t_0$ ,  $x(t_0)=x_0$ ,  $u(t_0)=u_0$ , and  $y(t_0)=g(x_0, u_0, t_0)=y_0$ . Subtracting the operating point values from the states, inputs, and outputs defines a set of variables centered about the operating point:

$$\begin{aligned}\delta x(t) &= x(t) - x_0 \\ \delta u(t) &= u(t) - u_0 \\ \delta y(t) &= y(t) - y_0\end{aligned}$$

The linearized model can be written in terms of these new variables and is usually valid when these variables are small, i.e. when the departure from the operating point is small:

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

The state-space matrices  $A$ ,  $B$ ,  $C$ , and  $D$  of this linearized model represent the Jacobians of the system, as defined in “Linearization of Nonlinear Models” on page 2-5. To compute the matrices, the states and inputs are perturbed, one at a time, and the response of the system to this perturbation is measured by

computing  $\delta\dot{x}$  and  $\delta y$ . The perturbation and response are then used to compute the matrices in the following way

$$A(:,i) = \frac{\dot{x}|_{x_{p,i}} - \dot{x}_o}{x_{p,i} - x_o}, \quad B(:,i) = \frac{\dot{x}|_{u_{p,i}} - \dot{x}_o}{u_{p,i} - u_o}$$

$$C(:,i) = \frac{y|_{x_{p,i}} - y_o}{x_{p,i} - x_o}, \quad D(:,i) = \frac{y|_{u_{p,i}} - y_o}{u_{p,i} - u_o}$$

where

- $x_{p,i}$  is the state vector whose  $i$ th component is perturbed from the operating point value.
- $x_o$  is the state vector at the operating point.
- $u_{p,i}$  is the input vector whose  $i$ th component is perturbed from the operating point value.
- $u_o$  is the input vector at the operating point.
- $\dot{x}|_{x_{p,i}}$  is the value of  $\dot{x}$  at  $x_{p,i}, u_o$ .
- $\dot{x}|_{u_{p,i}}$  is the value of  $\dot{x}$  at  $u_{p,i}, x_o$ .
- $\dot{x}_o$  is the value of  $\dot{x}$  at the operating point.
- $y|_{x_{p,i}}$  is the value of  $y$  at  $x_{p,i}, u_o$ .
- $y|_{u_{p,i}}$  is the value of  $y$  at  $u_{p,i}, x_o$ .
- $y_o$  is the value of  $y$  at the operating point.

Linearized models of discrete-time or multi-rate systems are computed in a similar way. See “Linearization of Discrete-Time Models” on page 2-7 and “Linearization of Multi-Rate Models” on page 2-8 for the equations of linearized discrete-time and multi-rate systems.

---

**Note** A perturbed value is one that has been changed by a very small amount from the operating point value. The default difference between the perturbed value and the operating point value is  $10^{-5} + 10^{-8}|x|$  for numerical-perturbation linearization.

---

## Controlling the Results of Numerical-Perturbation Linearization

Several factors influence the creation of accurate linearized models. Chapter 2, “What Is Linearization?” discusses some of these factors, such as careful selection of operating points. Factors that are particular to numerical-perturbation linearization are presented here, with suggestions for controlling them.

### Setting the Perturbation Level

In numerical-perturbation linearization, there are three options for setting the perturbation levels of states and inport blocks:

- 1 You can accept the default perturbation levels. The default perturbation levels for the states are  $10^{-5} + 10^{-8}|x|$ , where  $x$  is a vector of the operating point values for the states in the model. Similarly, default perturbation levels for the inport blocks are  $10^{-5} + 10^{-8}|u|$ , where  $u$  is a vector of the operating point values for the inputs in the model.
- 2 You can edit the linearization property `NumericalPertRel` using the `linoptions` function. The value of this property adjusts the perturbations in the following way
  - The perturbation of the states is  $\text{NumericalPertRel} + 1e - 3 \times \text{NumericalPertRel} \times |x|$ .
  - The perturbation of the inputs is  $\text{NumericalPertRel} + 1e - 3 \times \text{NumericalPertRel} \times |u|$ .
- 3 You can provide vectors of perturbation levels for the states and inport blocks. These values override the values computed using the `NumericalPertRel` value. Specify the perturbation levels by editing the linearization properties `NumericalXPert` and `NumericalUPert` using the

linoptions function. The properties NumericalXPert and NumericalUPert are vectors of absolute perturbation levels.

## Handling Special Blocks

Certain blocks, especially those containing discontinuities such as Saturation or Transport Delay, may not linearize well using numerical-perturbation. Although these blocks often have preprogrammed linearizations that are used with block-by-block analytic linearization instead of numerically perturbing them, they are *not* used in numerical-perturbation linearization. An alternative solution is to replace these blocks with an appropriate block before linearizing your model. For example, you might choose to replace a Saturation block with a Gain block.

## Handling Feedback Loops

“Understanding Open Loop Analysis” on page 2-11 discusses the effect of feedback loops on the results of a linearization. With block-by-block analytic linearization, you can perform open loop analysis without removing feedback loops. When using numerical-perturbation linearization, the only way to remove the effect of feedback loops is to manually remove them from the model *and* manually force the operating point to remain the same as the original model.