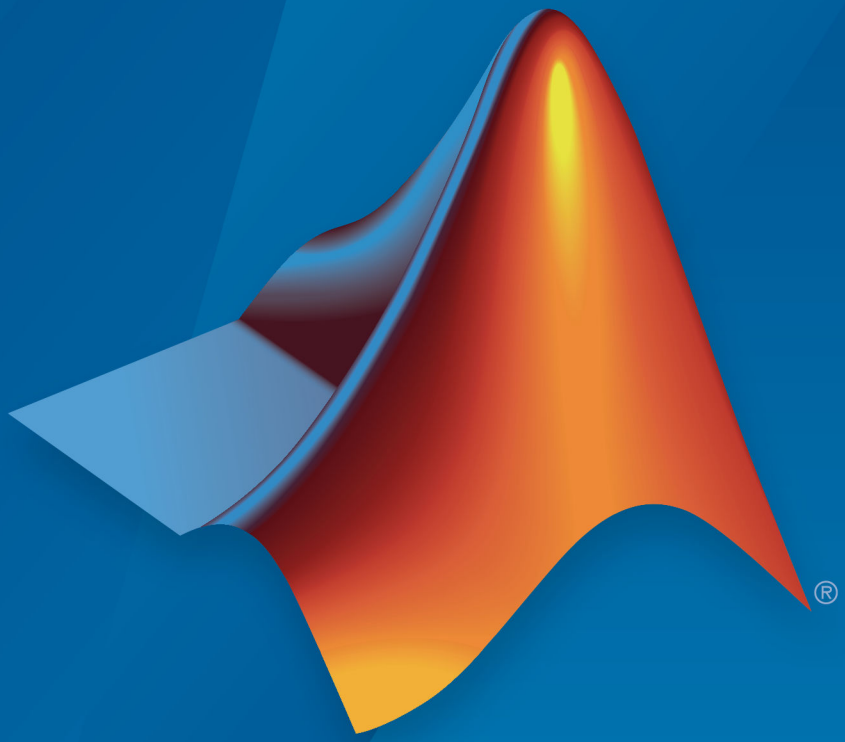
Simulink® Control Design™

Getting Started Guide



MATLAB® & SIMULINK®

R2017b

MathWorks®

How to Contact MathWorks

Latest news: www.mathworks.com

Sales and services: www.mathworks.com/sales_and_services

User community: www.mathworks.com/matlabcentral

Technical support: www.mathworks.com/support/contact_us

Phone: 508-647-7000

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

Simulink® Control Design™ Getting Started Guide

© COPYRIGHT 2004–2017 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

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Revision History

June 2004	Online only	New for Version 1.0 (Release 14)
October 2004	Online only	Revised for Version 1.1 (Release 14SP1)
March 2005	Online only	Revised for Version 1.2 (Release 14SP2)
September 2005	Online only	Revised for Version 1.3 (Release 14SP3)
March 2006	First printing	Revised for Version 2.0 (Release 2006a)
September 2006	Online only	Revised for Version 2.0.1 (Release 2006b)
March 2007	Online only	Revised for Version 2.1 (Release 2007a)
September 2007	Online only	Revised for Version 2.2 (Release 2007b)
March 2008	Second printing	Revised for Version 2.3 (Release 2008a)
October 2008	Online only	Revised for Version 2.4 (Release 2008b)
March 2009	Online only	Revised for Version 2.5 (Release 2009a)
September 2009	Third printing	Revised for Version 3.0 (Release 2009b)
March 2010	Online only	Revised for Version 3.1 (Release 2010a)
September 2010	Online only	Revised for Version 3.2 (Release 2010b)
April 2011	Online only	Revised for Version 3.3 (Release 2011a)
September 2011	Online only	Revised for Version 3.4 (Release 2011b)
March 2012	Online only	Revised for Version 3.5 (Release 2012a)
September 2012	Online only	Revised for Version 3.6 (Release 2012b)
March 2013	Online only	Revised for Version 3.7 (Release 2013a)
September 2013	Online only	Revised for Version 3.8 (Release 2013b)
March 2014	Online only	Revised for Version 4.0 (Release 2014a)
October 2014	Online only	Revised for Version 4.1 (Release 2014b)
March 2015	Online only	Revised for Version 4.2 (Release 2015a)
September 2015	Online only	Revised for Version 4.2.1 (Release 2015b)
March 2016	Online only	Revised for Version 4.3 (Release 2016a)
September 2016	Online only	Revised for Version 4.4 (Release 2016b)
March 2017	Online only	Revised for Version 4.5 (Release 2017a)
September 2017	Online only	Revised for Version 5.0 (Release 2017b)

Contents

1 Product Overview

Simulink Control Design Product Description 1-2
Key Features 1-2

2 Steady-State Operating Points

What Is a Steady-State Operating Point? 2-2

Steady-State Operating Points (Trimming) From Specifications 2-3

magball Simulink Model 2-12

3 Linearization

Applications of Linearization 3-2

Open-Loop Response of Control System for Stability Margin Analysis 3-3

Bode Response of Simulink Model 3-7

watertank Simulink Model 3-12

4 PID Control Design

PID Controller Tuning in Simulink **4-2**

1

Product Overview

Simulink Control Design Product Description

Linearize models and design control systems

Simulink Control Design lets you design and analyze control systems modeled in Simulink. You can automatically tune arbitrary SISO and MIMO control architectures, including PID controllers. PID autotuning can be deployed to embedded software for automatically computing PID gains in real time.

You can find operating points and compute exact linearizations of Simulink models at various operating conditions. Simulink Control Design provides tools that let you compute simulation-based frequency responses without modifying your model.

Key Features

- Automatic tuning of PID, gain-scheduled, and arbitrary SISO and MIMO control systems in Simulink
- PID autotuning algorithm deployable to embedded software
- Operating-point calculation (trimming) and model linearization
- Frequency response estimation from simulation data
- Batch linearization for varying parameters and operating points
- Numerical optimization of compensators to meet time-domain and frequency-domain requirements (with Simulink Design Optimization™)

Steady-State Operating Points

- "What Is a Steady-State Operating Point?" on page 2-2
- "Steady-State Operating Points (Trimming) From Specifications" on page 2-3
- "magball Simulink Model" on page 2-12

What Is a Steady-State Operating Point?

A *steady-state operating point* of a model, also called an equilibrium or *trim* condition, includes state variables that do not change with time.

A model can have several steady-state operating points. For example, a hanging damped pendulum has two steady-state operating points at which the pendulum position does not change with time. A *stable steady-state operating point* occurs when a pendulum hangs straight down. When the pendulum position deviates slightly, the pendulum always returns to equilibrium. In other words, small changes in the operating point do not cause the system to leave the region of good approximation around the equilibrium value.

An *unstable steady-state operating point* occurs when a pendulum points upward. As long as the pendulum points *exactly* upward, it remains in equilibrium. However, when the pendulum deviates slightly from this position, it swings downward and the operating point leaves the region around the equilibrium value.

When using optimization search to compute operating points for nonlinear systems, your initial guesses for the states and input levels must be near the desired operating point to ensure convergence.

When linearizing a model with multiple steady-state operating points, it is important to have the right operating point. For example, linearizing a pendulum model around the stable steady-state operating point produces a stable linear model, whereas linearizing around the unstable steady-state operating point produces an unstable linear model.

See Also

`findop`

More About

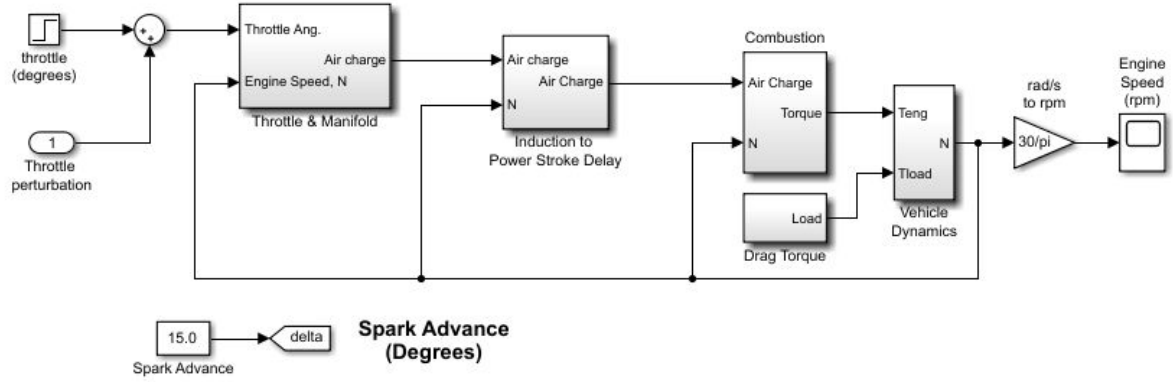
- “Compute Steady-State Operating Points”
- “Simulink Model States Included in Operating Point Object”
- “Steady-State Operating Points (Trimming) From Specifications” on page 2-3
- “Compute Operating Points at Simulation Snapshots”

Steady-State Operating Points (Trimming) From Specifications

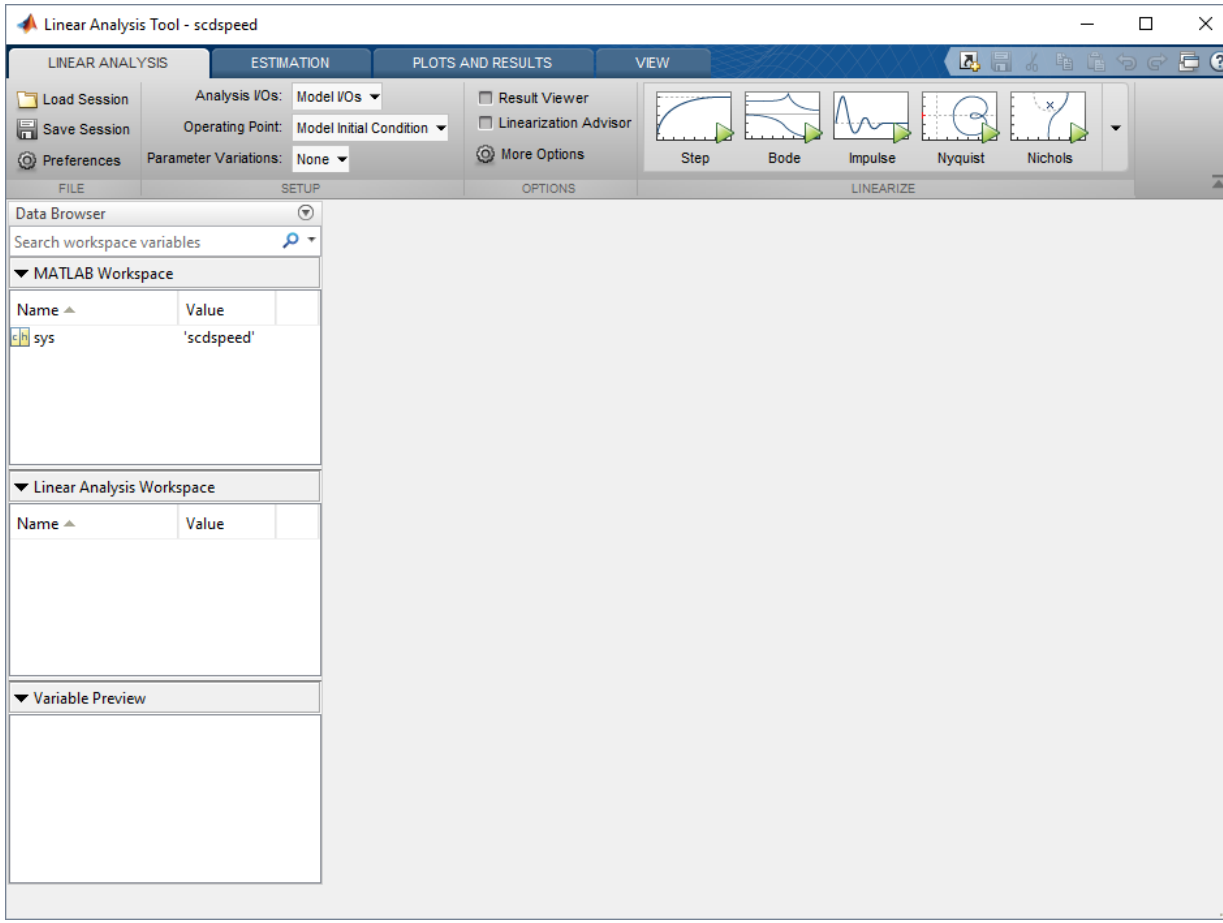
This example shows how to compute a steady-state operating point by specifying known state values and constraints using the Linear Analysis Tool.

Open the Simulink model.

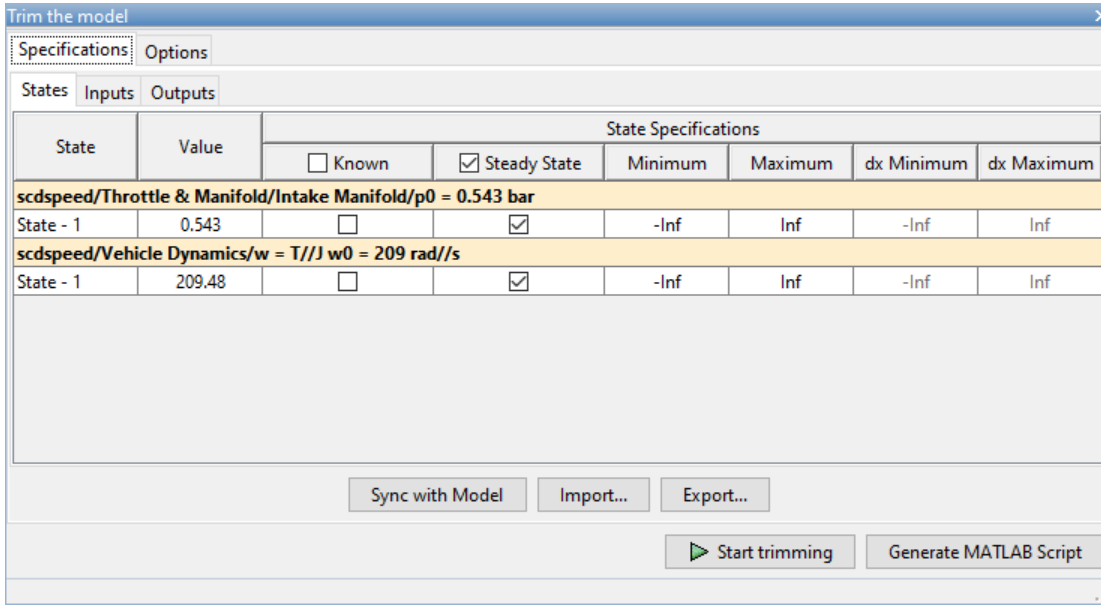
```
sys = 'scdspeed';
open_system(sys)
```



To open the Linear Analysis Tool, in the Simulink model window, select **Analysis > Control Design > Linear Analysis**.

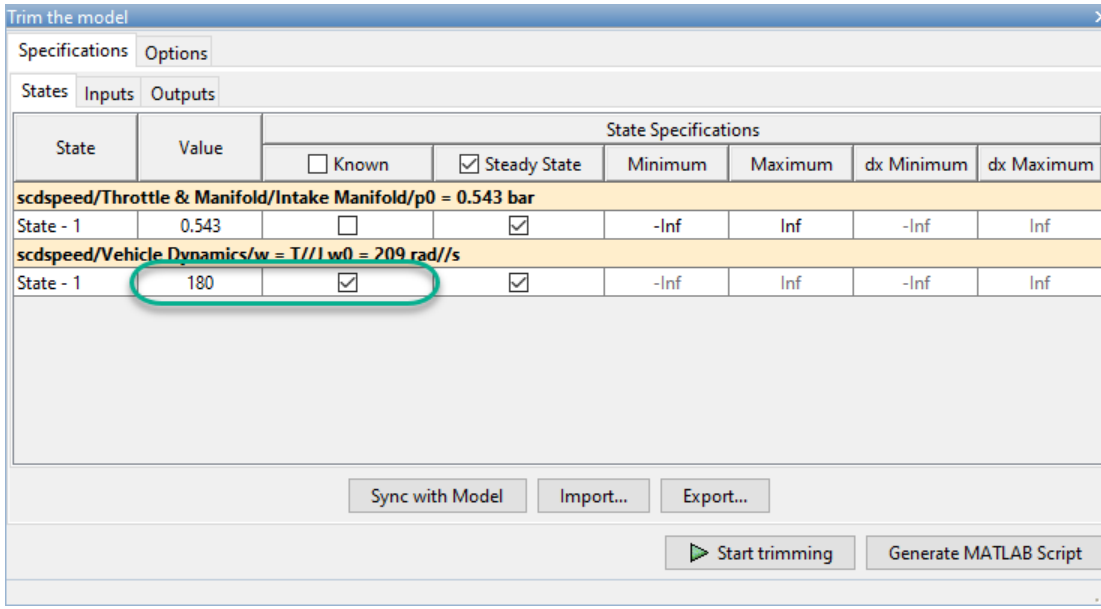


In the Linear Analysis Tool, on the **Linear Analysis** tab, in the **Operating Point** drop-down list, select `Trim Model`.

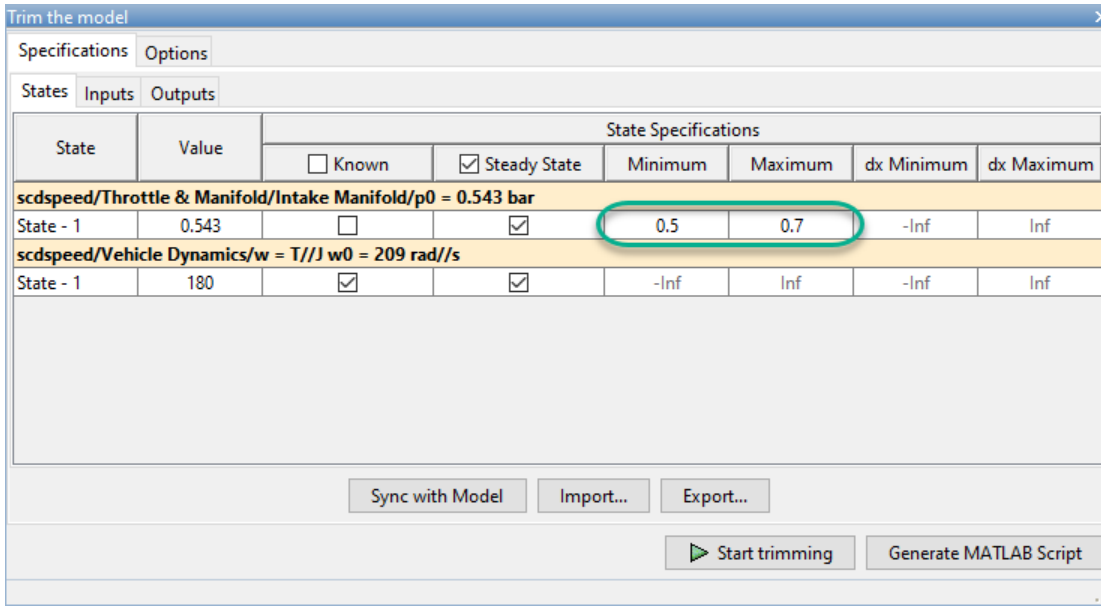


By default, on the **States** tab, the software specifies both model states to be at equilibrium, as shown by the check marks in the **Steady State** column. both states are also specified as unknown values; that is, their steady-state values are calculated during trimming, with an initial guess specified in the **Value** column.

Change the second state, the engine angular velocity, to be a known value. In the **Known** column, select the corresponding row and, in the **Value** column, set the value to `180`.

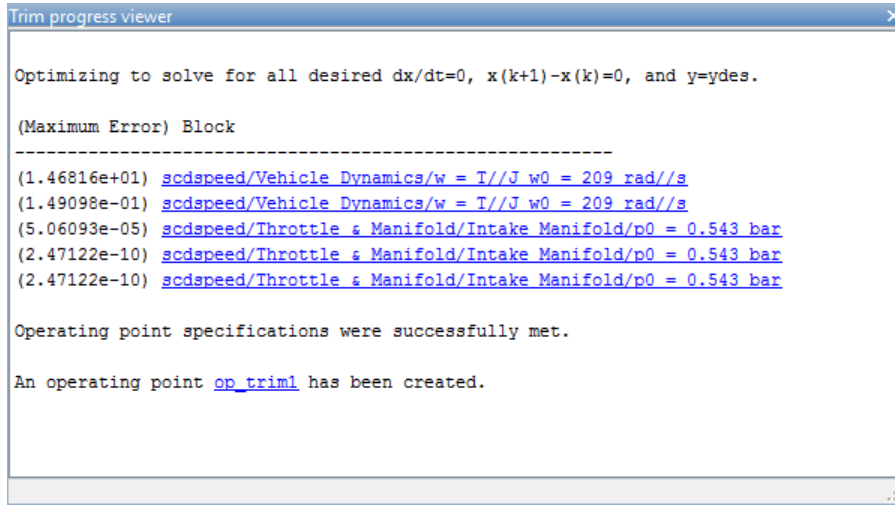


You can also specify bounds for model states during trimming. For this example, constrain the first state to be between `0.5` and `0.7`. To do so, enter these values in the **Minimum** and **Maximum** columns, respectively.



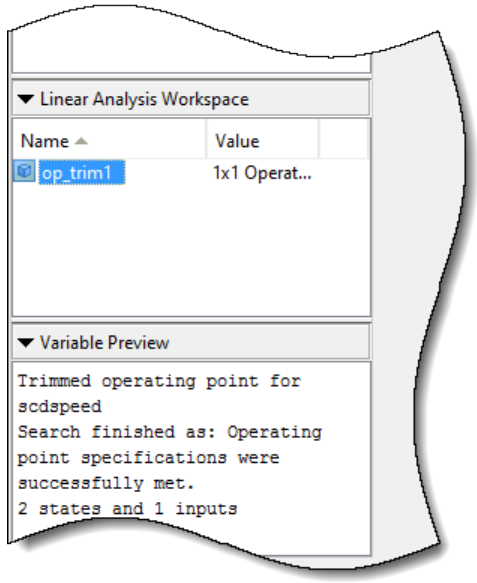
To compute the operating point that meets these specifications, click **Start trimming**.

The software uses optimization to find the operating point that meets your specifications.



The Trim progress viewer shows the optimization progress and that the optimization algorithm terminated successfully. The **(Maximum Error)** column shows the maximum constraint violation at each iteration. The **Block** column shows the block to which the constraint violation applies.

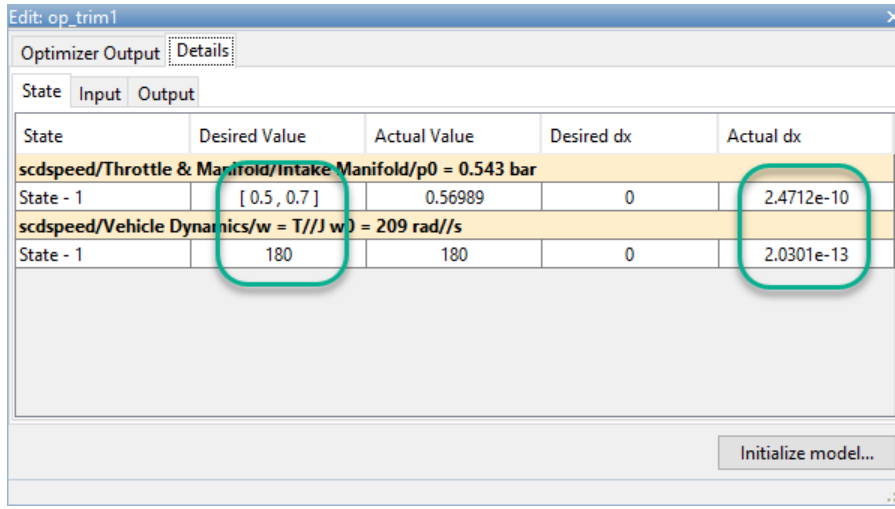
The trimmed operating point, `op_trim1`, appears in the **Linear Analysis Workspace**.



To evaluate whether the resulting operating point values meet the specifications, in the **Linear Analysis Workspace**, double-click `op_trim1`.

In the Edit dialog box, on the **State** tab, the **Actual Value** for the first state falls within the **Desired Value** bounds, and the actual angular velocity is `180`, as specified.

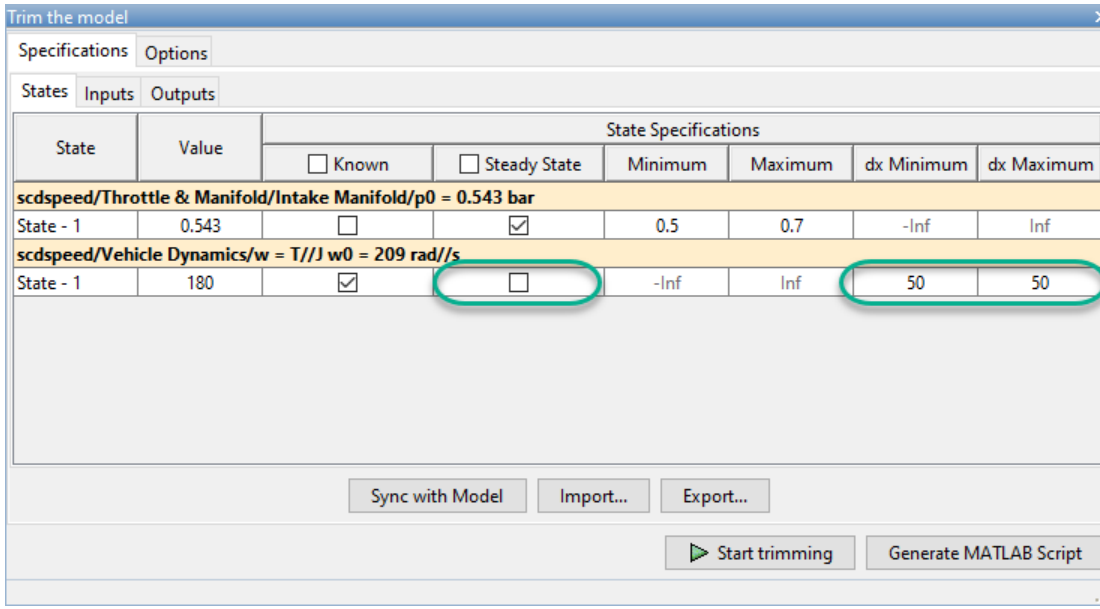
The **Actual dx** column shows the rates of change of the state values at the operating point. Since these values are at or near zero the states are not changing, showing that the operating point is in a steady state.



You can also constrain the derivatives of states that are not at steady state. Using such constraints, you can trim derivatives to known nonzero values or specify derivative tolerances for states that cannot reach steady state.

For example, suppose that you want to find the operating condition at which the engine angular velocity is 180 rad/s and the angular acceleration is `50` rad/s^2. To do so, first open the Trim the model dialog box. In the Linear Analysis Tool, in the **Operating Point** drop-down list, select `Trim Model`.

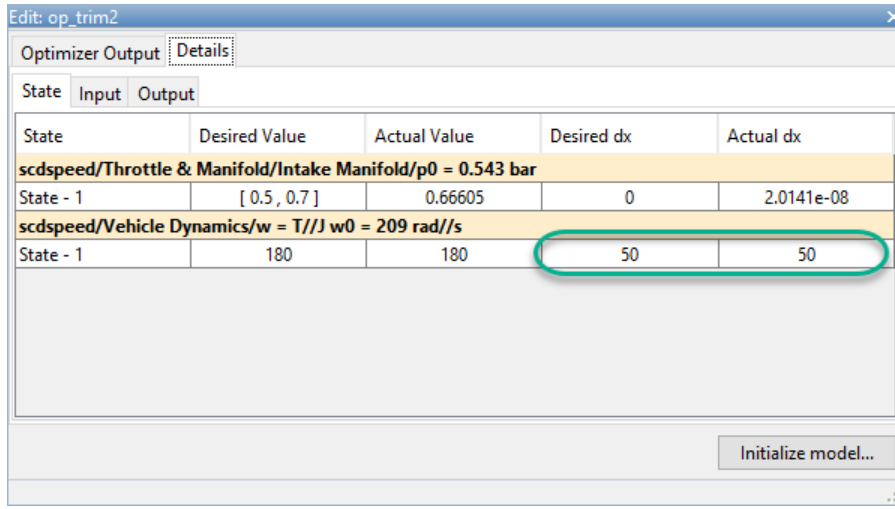
In the **Steady State** column, clear the selection in the corresponding row. Then, in the **dx Minimum** and **dx Maximum** columns, set both state derivative bounds to `50`.



To compute the operating point, click **Start trimming**.

In the Linear Analysis Tool, in the **Linear Analysis Workspace**, double-click `op_trim2`.

In the Edit dialog box, in the second row, the **Actual dx** column matches the **Desired dx** column. Therefore the operating point meets the specified state derivative constraints.



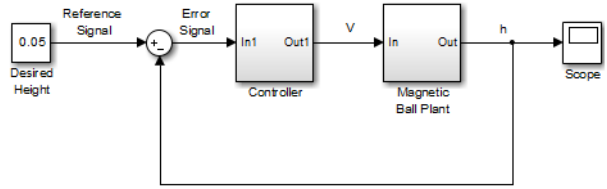
See Also

More About

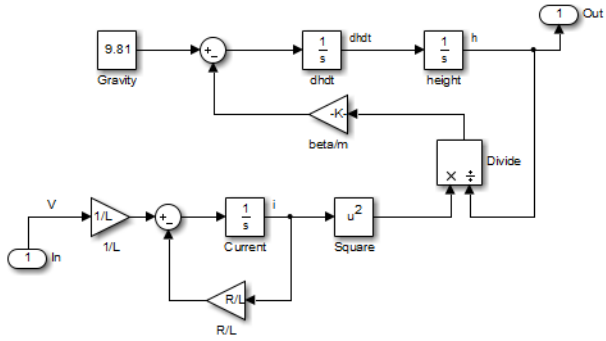
- “magball Simulink Model” on page 2-12
- “Compute Steady-State Operating Points”
- “Compute Operating Points at Simulation Snapshots”
- “Steady-State Operating Points (Trimming) From Specifications” on page 2-3

magball Simulink Model

The Simulink model `magball` includes the nonlinear Magnetic Ball Plant in a single-loop feedback system.

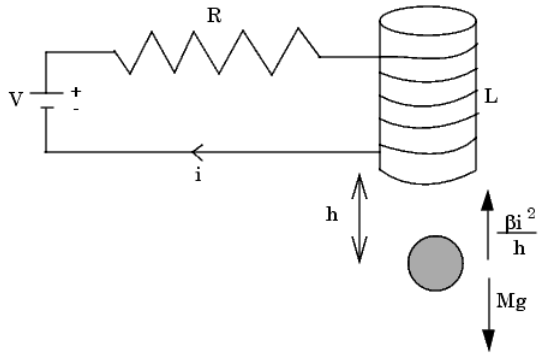


The Magnetic Ball Plant subsystem is shown in the following figure.



The Magnetic Ball Plant model represents an iron ball of mass *M*. This ball moves under the influence of the gravitational force, *Mg*, and an induced magnetic force, $\frac{\beta i^2}{h}$. The presence of the squared term in the induced magnetic force results in a nonlinear plant.

The inductor in the electric circuit, shown in the following figure, causes the induced magnetic force. This circuit also includes a voltage source and a resistor.



The following table describes the variables, parameters, differential equations, states, inputs, and outputs of the Magnetic Ball Plant subsystem.

Variables	h is the height of the ball. i is the current. V is the voltage in the circuit.
Parameters	M is the mass of the ball. g is the gravitational acceleration. β is a constant related to the magnetic force. L is the inductance of the coil. R is the resistance of the circuit.
Differential equations	The height of the ball, h, is described in the following equation: $M\frac{d^2h}{dt^2} = Mg - \frac{\beta i^2}{h}$ The current in the circuit, i, is described in the following equation: $L\frac{di}{dt} = V - iR$

States	h dh/dt i
Inputs	V
Outputs	h

See Also

More About

- “Steady-State Operating Points (Trimming) From Specifications” on page 2-3

Linearization

- "Applications of Linearization" on page 3-2
- "Open-Loop Response of Control System for Stability Margin Analysis" on page 3-3
- "Bode Response of Simulink Model" on page 3-7
- "watertank Simulink Model" on page 3-12

Applications of Linearization

Linearization is useful in model analysis and control design applications. After you linearize a Simulink model at a specific operating point, you can use your linear model to:

- Compute the Bode response of the Simulink model.
- Evaluate loop stability margins by computing open-loop response.
- Obtain linear state-space, transfer-function, or zero-pole-gain representation of the combined Simulink model that contains only linear blocks.
- Analyze and compare plant response near different operating points.
- Design linear controller

  Classical control system analysis and design methodologies require linear, time-invariant models. Simulink Control Design automatically linearizes the plant when you tune your compensator. See “Design Compensator Using Automated PID Tuning and Graphical Bode Design”.
- Analyze closed-loop stability.
- Measure the size of resonances in frequency response by computing closed-loop linear model for control system.
- Generate controllers with reduced sensitivity to parameter variations and modeling errors (requires Robust Control Toolbox™).

Examples and How To

- “Open-Loop Response of Control System for Stability Margin Analysis” on page 3-3
- “Bode Response of Simulink Model” on page 3-7
- “Steady-State Operating Points (Trimming) From Specifications” on page 2-3

More About

“Linearize Nonlinear Models”

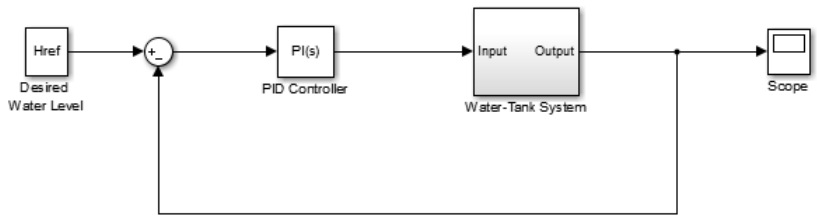
Open-Loop Response of Control System for Stability Margin Analysis

This example shows how to use the Linear Analysis Tool to analyze the open-loop response of a control system.

This example shows how to compute a linear model of the combined controller-plant system without the effects of the feedback signal. You can analyze the resulting linear model using, for example, a Bode plot.

Open Simulink model.

```
sys = 'watertank';
open_system(sys)
```

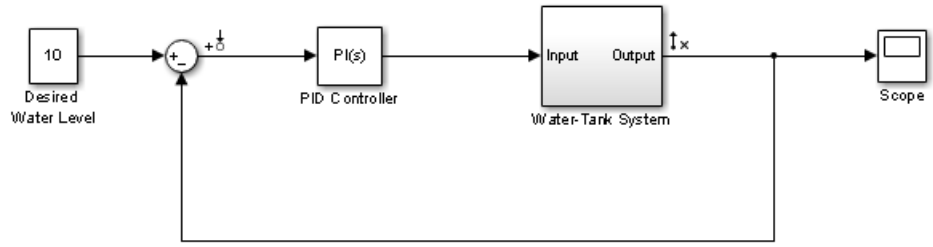


The Water-Tank System block represents the plant in this control system and contains all of the system nonlinearities.

In the Simulink Editor, specify the portion of the model to linearize. For this example, specify the loop opening using open-loop output analysis point.

1. Right-click the PID Controller block input signal (the output of the Sum block), and select **Linear Analysis Points > Input Perturbation**.
2. Right-click the Water-Tank System output signal, and select **Linear Analysis Points > Open-loop Output**.

Annotations appear in the model indicating which signals are designated as analysis points.

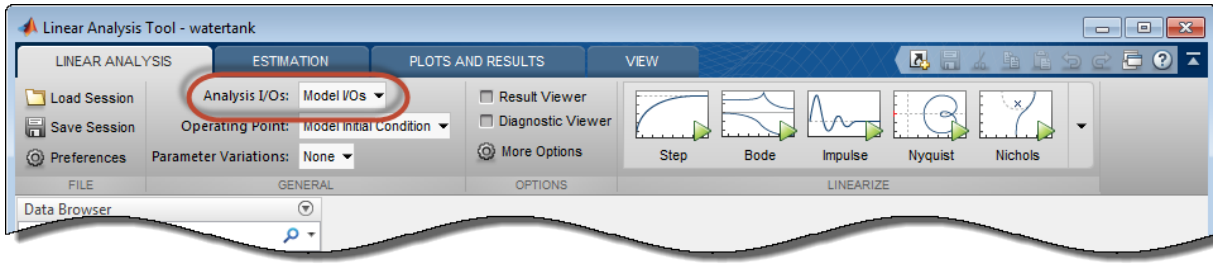


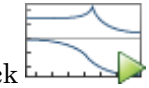
Tip If you do not want to introduce changes to the Simulink model, you can specify the analysis points in the Linear Analysis Tool. For more information, see "Specify Portion of Model to Linearize in Linear Analysis Tool".

Open the Linear Analysis Tool for the model.

In the Simulink Editor, select **Analysis > Control Design > Linear Analysis**.

By default, the analysis points you specified in the model are selected for linearization, as displayed in the **Analysis I/Os** drop-down list.

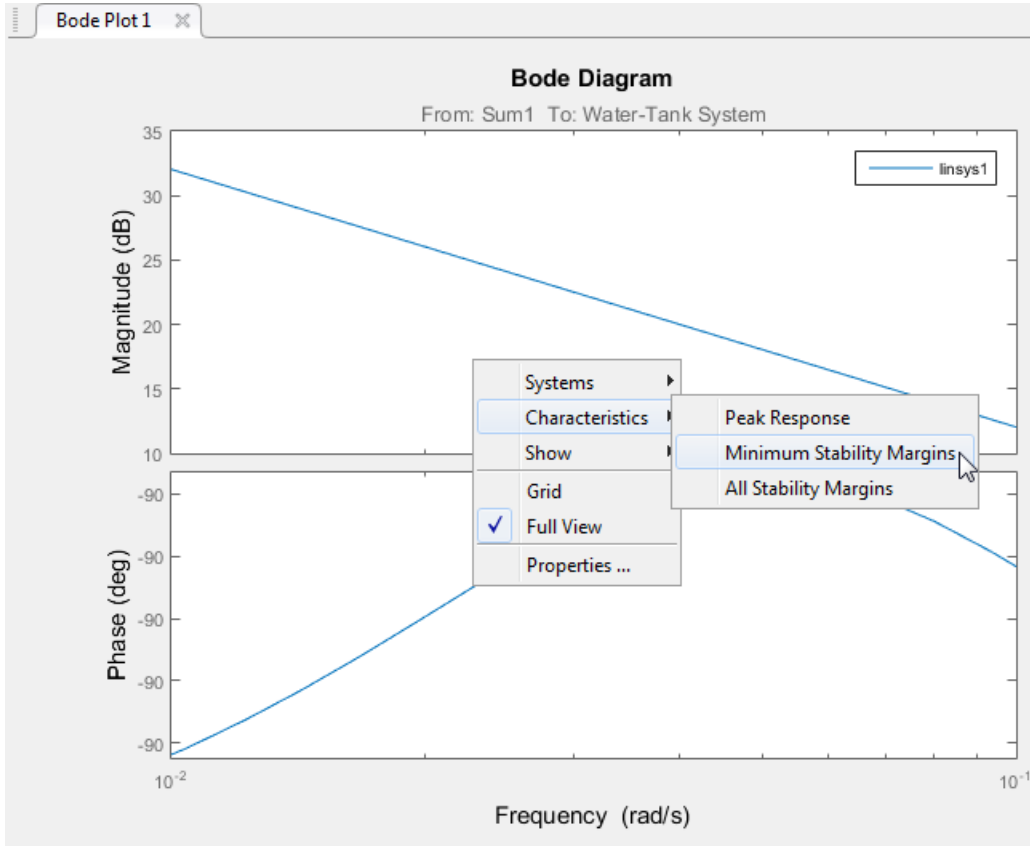


To linearize the model using the specified analysis points and generate a Bode plot of the linearized model, click  **Bode**.

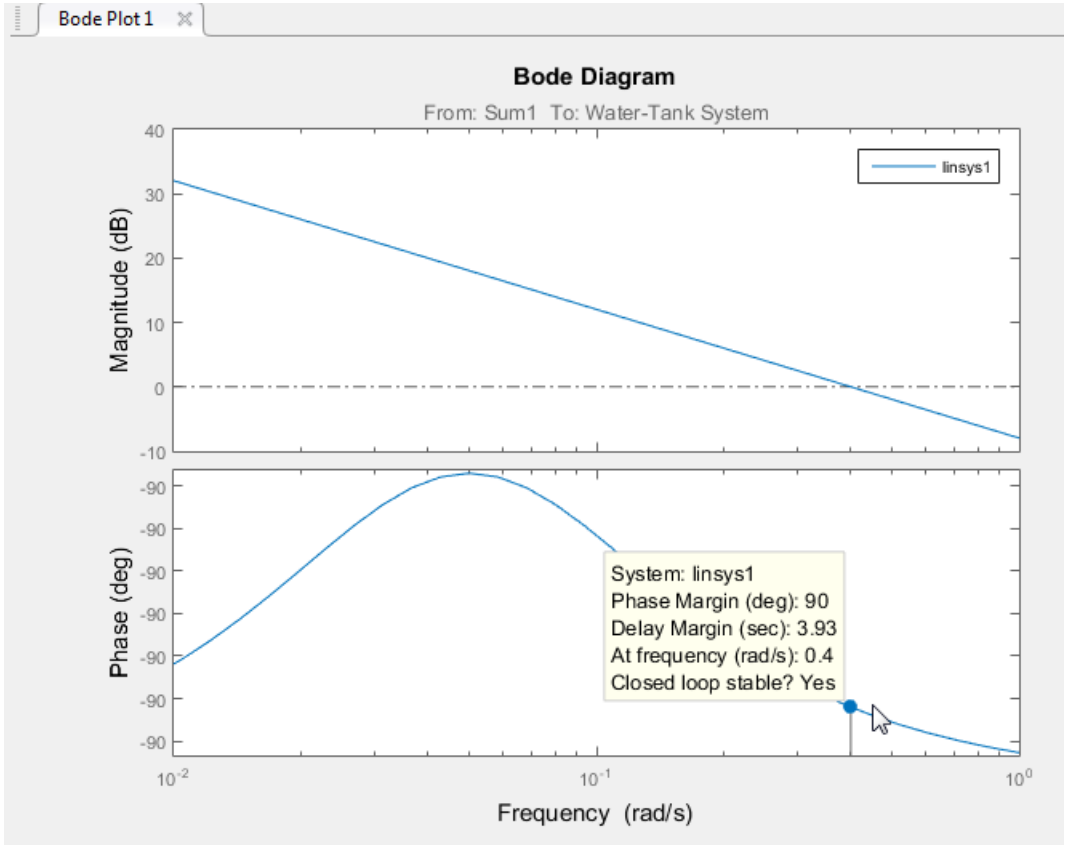
By default, the Linear Analysis Tool linearizes the model at the model initial conditions, as shown in the **Operating Point** drop-down list. For examples of linearizing a model at a different operating point, see "Linearize at Trimmed Operating Point" and "Linearize at Simulation Snapshot".

Tip To generate response types other than a Bode plot, click the corresponding button in the plot gallery.

To view the minimum stability margins for the model, right-click the Bode plot, and select **Characteristics > Minimum Stability Margins**.



The Bode plot displays the phase margin marker. To show a data tip that contains the phase margin value, click the marker.



For this system, the phase margin is 90 degrees at a crossover frequency of 0.4 rad/s.

Related Examples

- "Bode Response of Simulink Model" on page 3-7
- "Steady-State Operating Points (Trimming) From Specifications" on page 2-3

More About

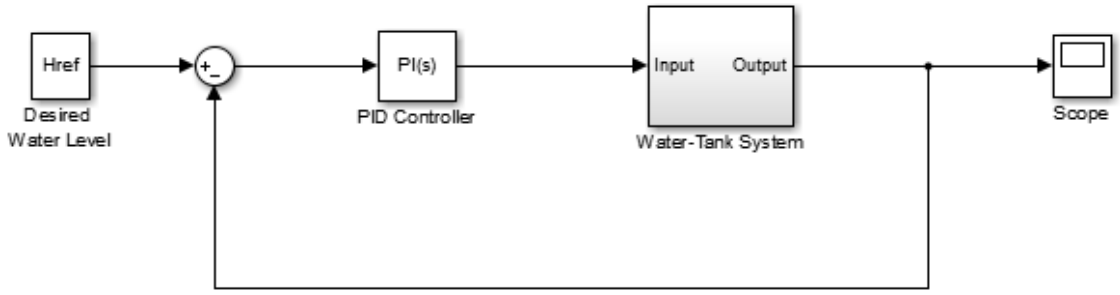
- "Linearize Nonlinear Models"
- "watertank Simulink Model" on page 3-12

Bode Response of Simulink Model

This example shows how to linearize a Simulink model at the operating point specified in the model using the **Linear Analysis Tool**.

Open Simulink model.

```
mdl = 'watertank';
open_system(mdl)
```

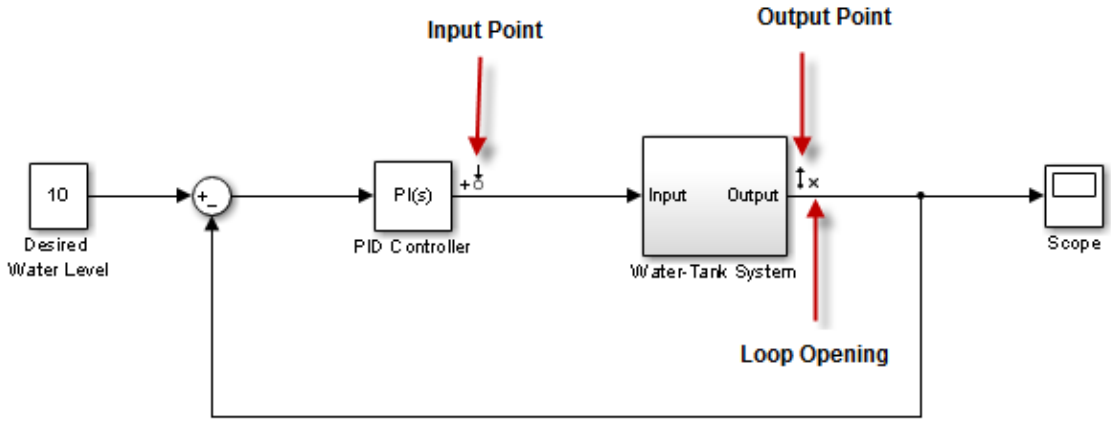


The Water-Tank System block represents the plant in this control system and includes all of the system nonlinearities.

In the Simulink window, specify the portion of the model to linearize:

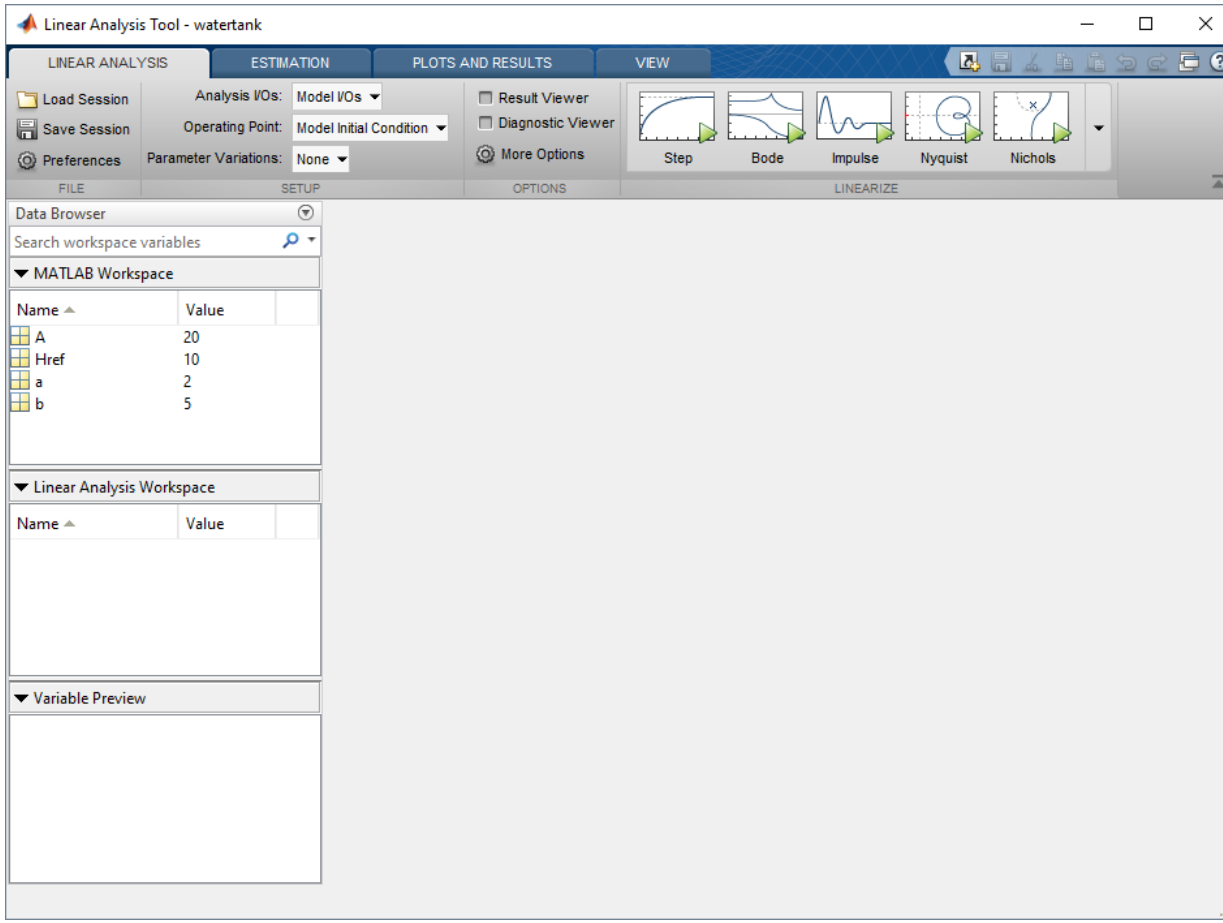
1. To specify the linearization input, right-click the output signal of the PID Controller block, and select **Linear Analysis Points** > **Input Perturbation**.
2. To specify the linearization output, right-click the output signal of the Water-Tank System, and select **Linear Analysis Points** > **Open-loop Output**. An open-loop output point is an output measurement followed by a loop opening, which removes the effects of the feedback signal on the linearization without changing the model operating point.

When you add linear analysis points, the software adds markers at their respective locations in the model. For more information on the different types of analysis points, see “Specify Portion of Model to Linearize”.



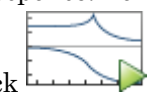
For more information on defining analysis points in a Simulink model, see “Specify Portion of Model to Linearize in Simulink Model”. Alternatively, if you do not want to introduce changes to the Simulink model, you can define analysis points using the Linear Analysis Tool. For more information, see “Specify Portion of Model to Linearize in Linear Analysis Tool”.

To open the Linear Analysis Tool for the model, in the Simulink model window, select **Analysis > Control Design > Linear Analysis**.

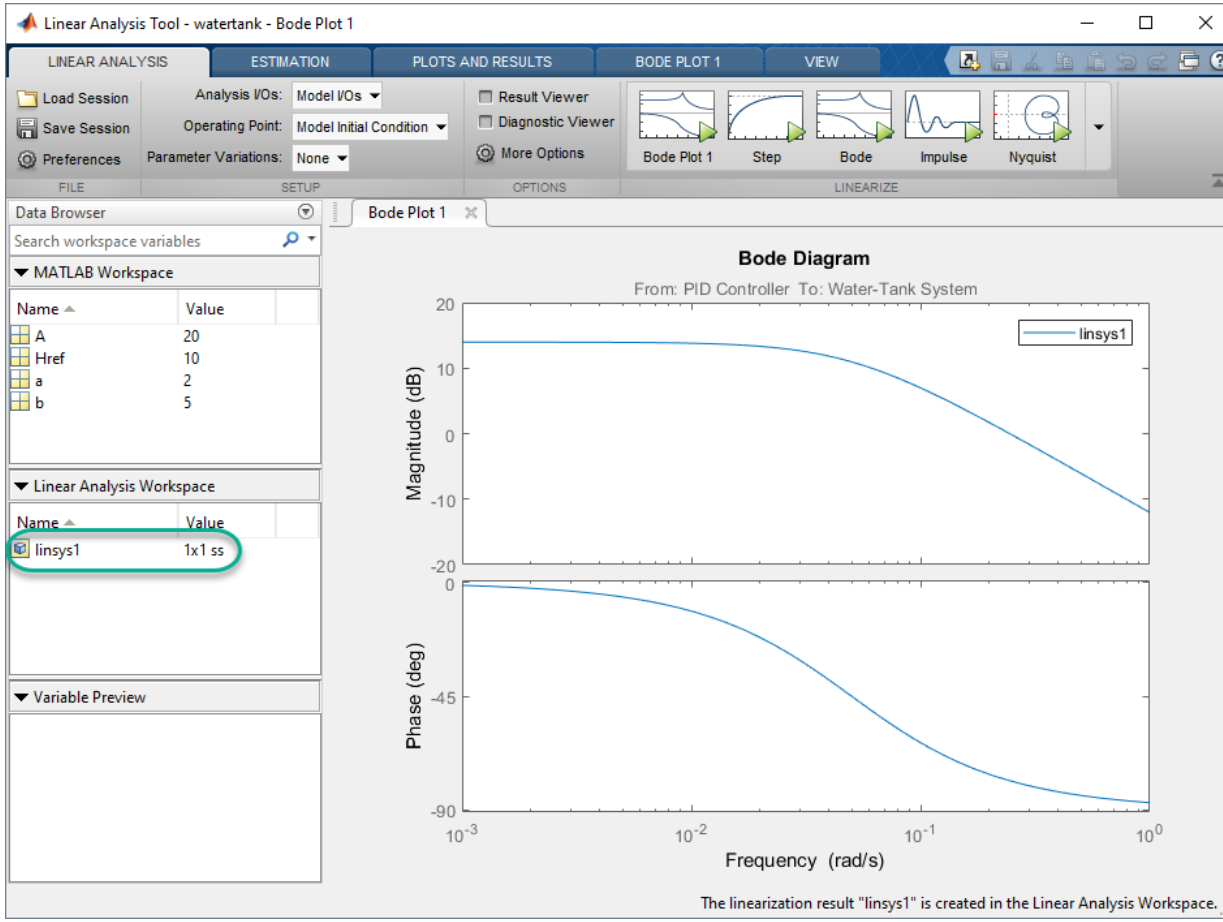


To use the analysis points you defined in the Simulink model as linearization I/Os, on the **Linear Analysis** tab, in the **Analysis I/Os** drop-down list, leave `Model I/Os` selected.

For this example, use the model operating point for linearization. In the **Operating Point** drop-down list, leave `Model Initial Condition` selected.

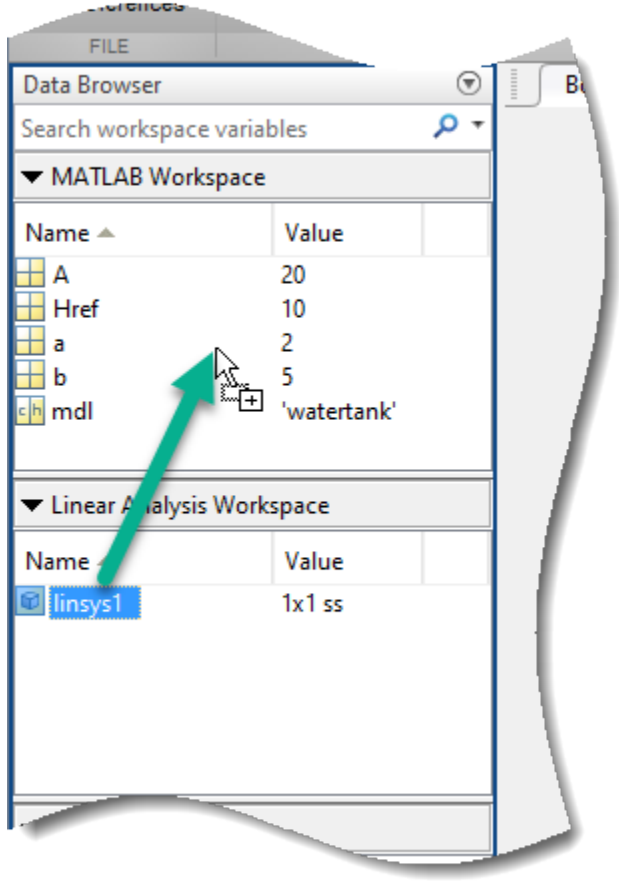
To linearize the system and generate a response plot for analysis, in the **Linearize** section, click a response. For this example, to generate a Bode plot for the resulting linear model, click  **Bode**.

The software adds the linearized model, `linsys1`, to the **Linear Analysis Workspace** and generates a Bode plot for the model. `linsys1` is the linear model from the specified input to the specified output, computed at the default model operating point.



For more information on analyzing linear models, see “Analyze Results Using Linear Analysis Tool Response Plots”.

You can also export the linearized model to the MATLAB® workspace. To do so, in the **Data Browser**, drag `linsys1` from the **Linear Analysis Workspace** to the **MATLAB Workspace**.



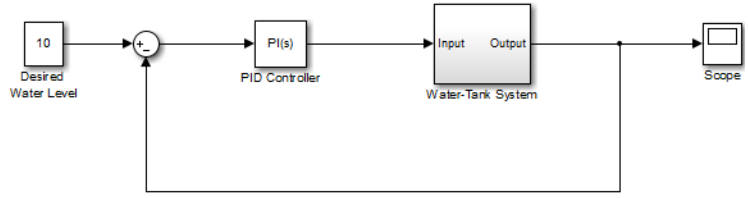
See Also

More About

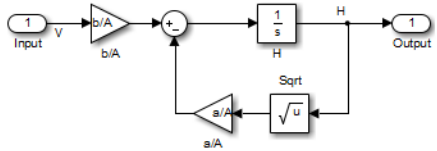
- “Linearize Nonlinear Models”
- “watertank Simulink Model” on page 3-12
- “Open-Loop Response of Control System for Stability Margin Analysis” on page 3-3
- “Steady-State Operating Points (Trimming) From Specifications” on page 2-3

watertank Simulink Model

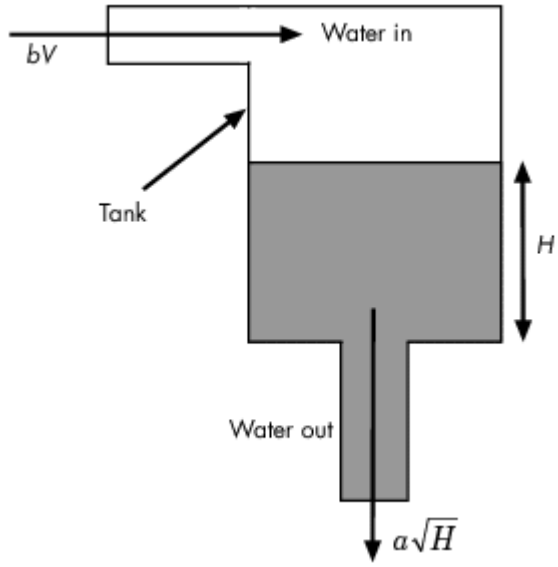
The Simulink model `watertank` includes the nonlinear Water-Tank System plant and a PI controller in a single-loop feedback system.



The Water-Tank System is shown in the following figure.



Water enters the tank from the top at a rate proportional to the voltage, *V*, applied to the pump. The water leaves through an opening in the tank base at a rate that is proportional to the square root of the water height, *H*, in the tank. The presence of the square root in the water flow rate results in a nonlinear plant.



The following table describes the variables, parameters, differential equations, states, inputs, and outputs of the Water-Tank System.

Variables	*H* is the height of water in the tank. *Vol* is the volume of water in the tank. *V* is the voltage applied to the pump.
Parameters	*A* is the cross-sectional area of the tank. *b* is a constant related to the flow rate into the tank. *a* is a constant related to the flow rate out of the tank.
Differential equation	$\frac{d}{dt}Vol = A\frac{dH}{dt} = bV - a\sqrt{H}$
States	*H*
Inputs	*V*

Outputs	H

4

PID Control Design

PID Controller Tuning in Simulink

This example shows how to automatically tune a PID Controller block using PID Tuner.

Introduction of the PID Tuner

PID Tuner provides a fast and widely applicable single-loop PID tuning method for the Simulink® PID Controller blocks. With this method, you can tune PID controller parameters to achieve a robust design with the desired response time.

A typical design workflow with the PID Tuner involves the following tasks:

(1) Launch the PID Tuner. When launching, the software automatically computes a linear plant model from the Simulink model and designs an initial controller.

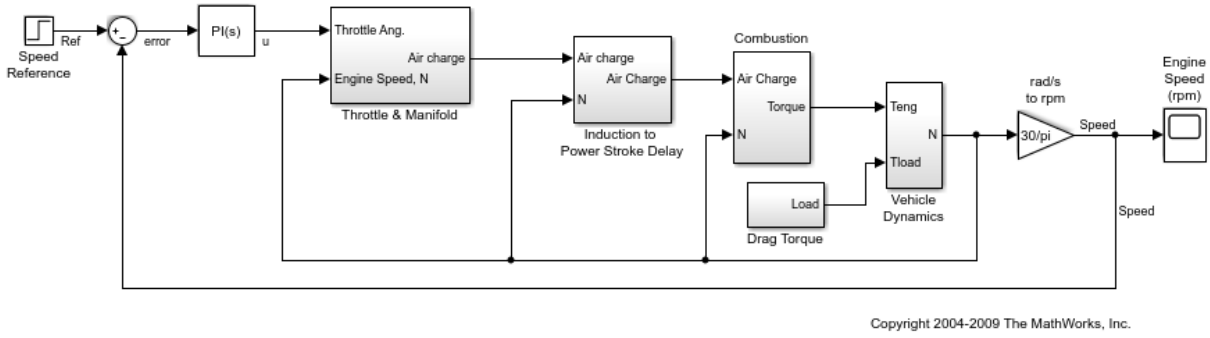
(2) Tune the controller in the PID Tuner by manually adjusting design criteria in two design modes. The tuner computes PID parameters that robustly stabilize the system.

(3) Export the parameters of the designed controller back to the PID Controller block and verify controller performance in Simulink.

Opening the Model

Open the engine speed control model with PID Controller block and take a few moments to explore it.

```
open_system('scdspeedctrlpidblock');
```



Design Overview

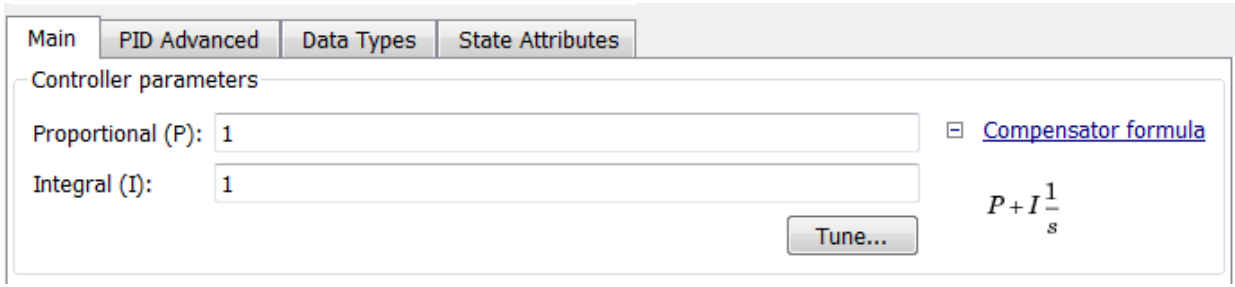
In this example, you design a PI controller in an engine speed control loop. The goal of the design is to track the reference signal from a Simulink step block `scdspeedctrlpidblock/Speed Reference`. The design requirement are:

- Settling time under 5 seconds
- Zero steady-state error to the step reference input.

In this example, you stabilize the feedback loop and achieve good reference tracking performance by designing the PI controller `scdspeedctrl/PID Controller` in the PID Tuner.

Opening the PID Tuner

To launch the PID Tuner, double-click the PID Controller block to open its block dialog. In the **Main** tab, click **Tune**.

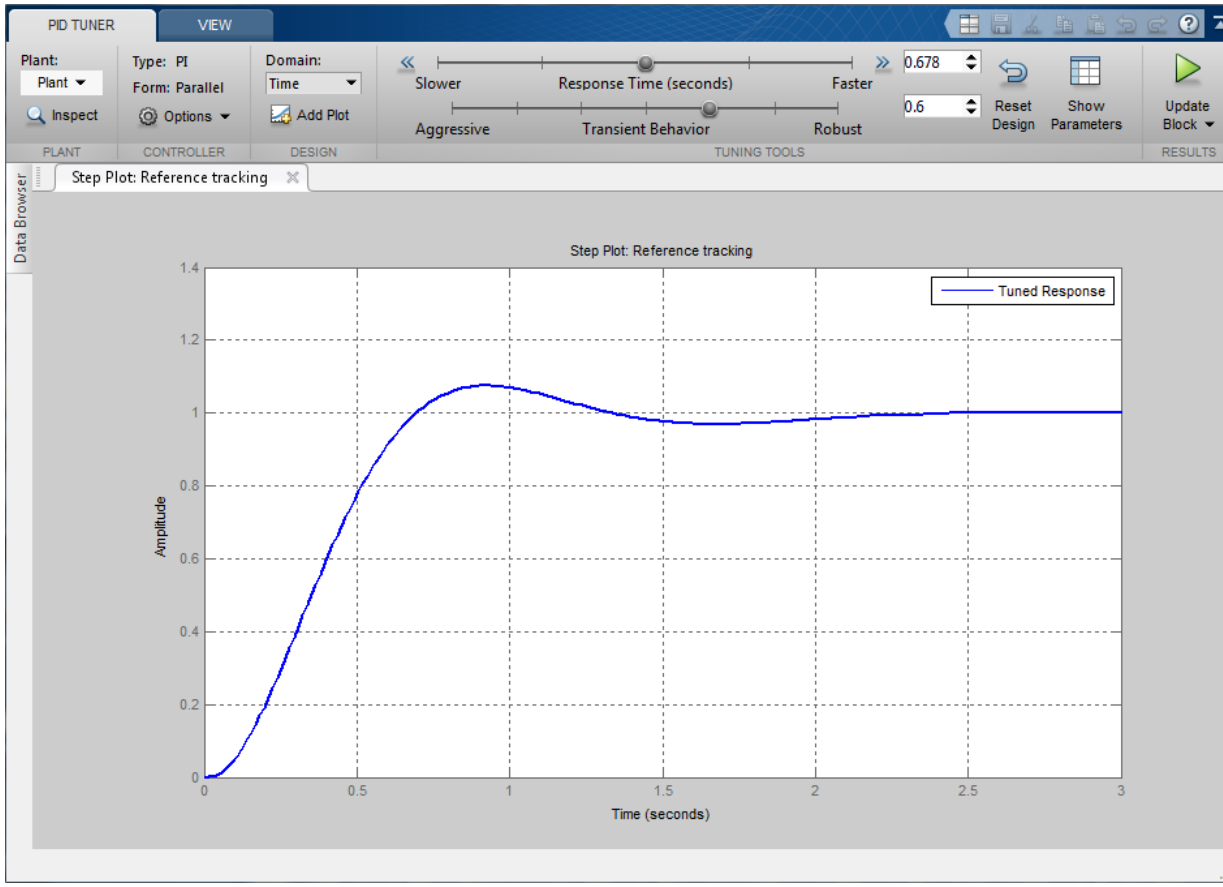


Initial PID Design

When the PID Tuner launches, the software computes a linearized plant model seen by the controller. The software automatically identifies the plant input and output, and uses the current operating point for the linearization. The plant can have any order and can have time delays.

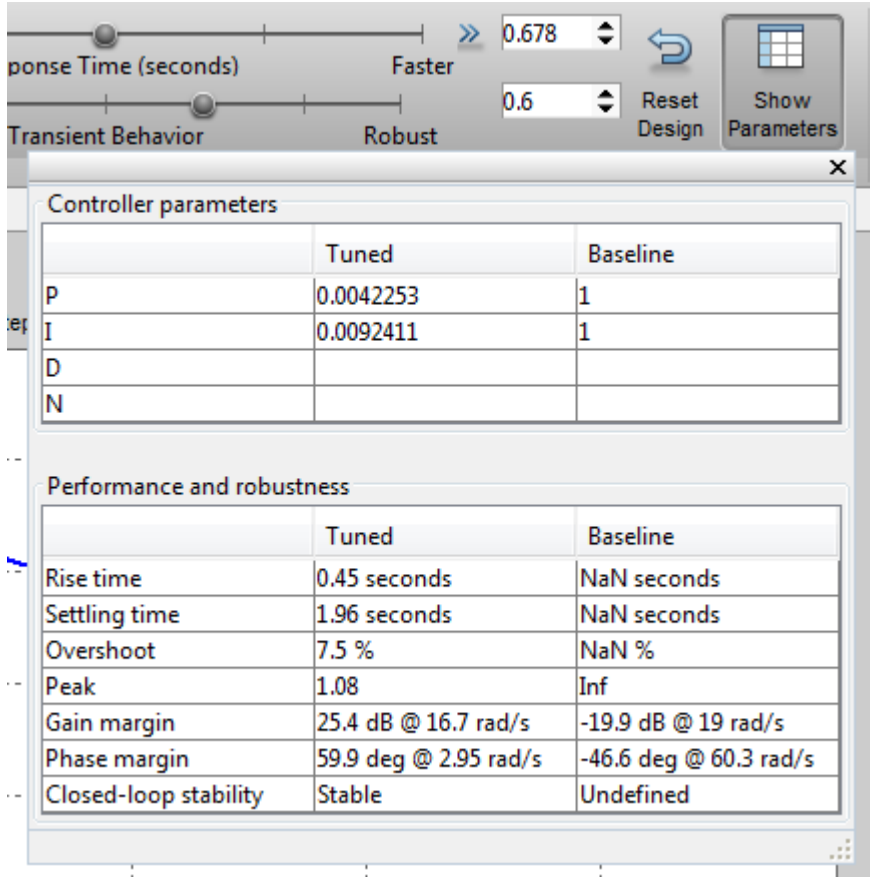
The PID Tuner computes an initial PI controller to achieve a reasonable tradeoff between performance and robustness. By default, step reference tracking performance displays in the plot.

The following figure shows the PID Tuner dialog with the initial design:



Displaying PID Parameters

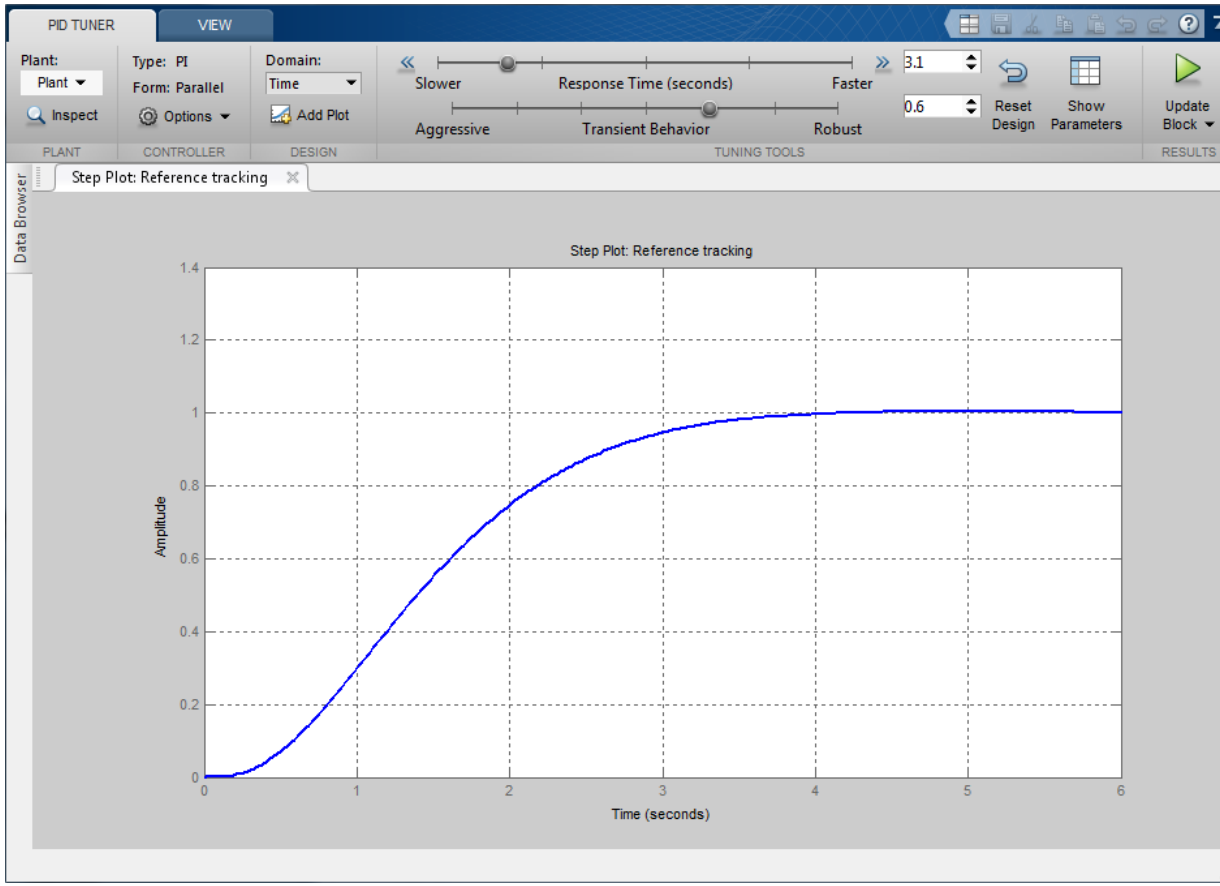
Click **Show parameters** to view controller parameters P and I, and a set of performance and robustness measurements. In this example, the initial PI controller design gives a settling time of 2 seconds, which meets the requirement.

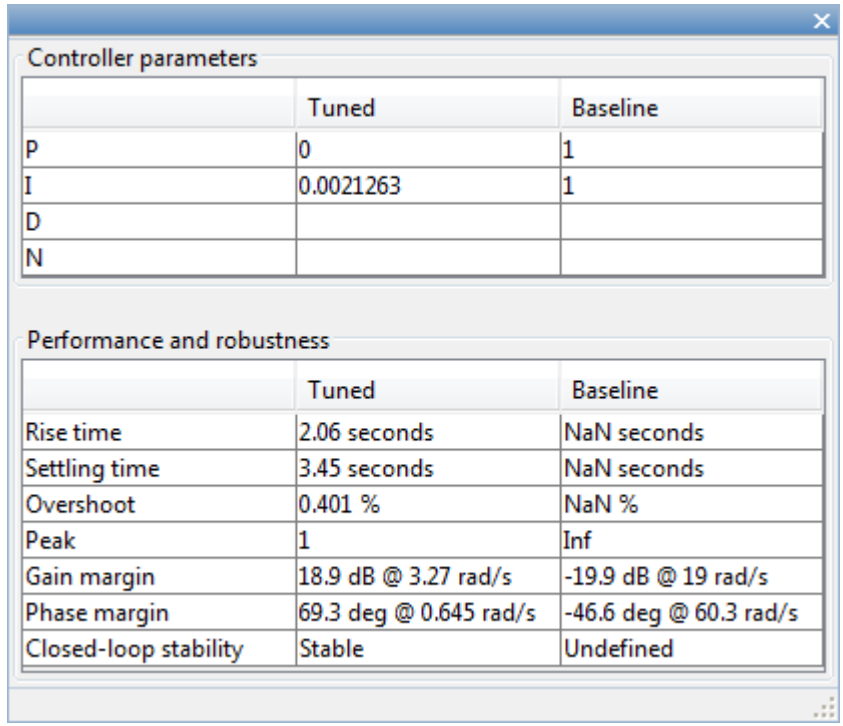


Adjusting PID Design in the PID Tuner

The overshoot of the reference tracking response is about 7.5 percent. Since we still have some room before reaching the settling time limit, you could reduce the overshoot by increasing the response time. Move the response time slider to the left to increase the closed loop response time. Notice that when you adjust response time, the response plot and the controller parameters and performance measurements update.

The following figure shows an adjusted PID design with an overshoot of zero and a settling time of 4 seconds. The designed controller effectively becomes an integral-only controller.


PID TUNER
VIEW
Plant:
Plant
Inspect
PLANT
Type: PI
Form: Parallel
Options
CONTROLLER
Domain:
Time
Add Plot
DESIGN
Slower
Response Time (seconds)
Faster
3.1
Aggressive
Transient Behavior
Robust
0.6
Reset Design
Show Parameters
TUNING TOOLS
Update Block
RESULTS
Data Browser
Step Plot: Reference tracking
Step Plot: Reference tracking
Amplitude
Time (seconds)
0
0.2
0.4
0.6
0.8
1
1.2
1.4
0
1
2
3
4
5
6



Controller parameters

	Tuned	Baseline
P	0	1
I	0.0021263	1
D		
N		

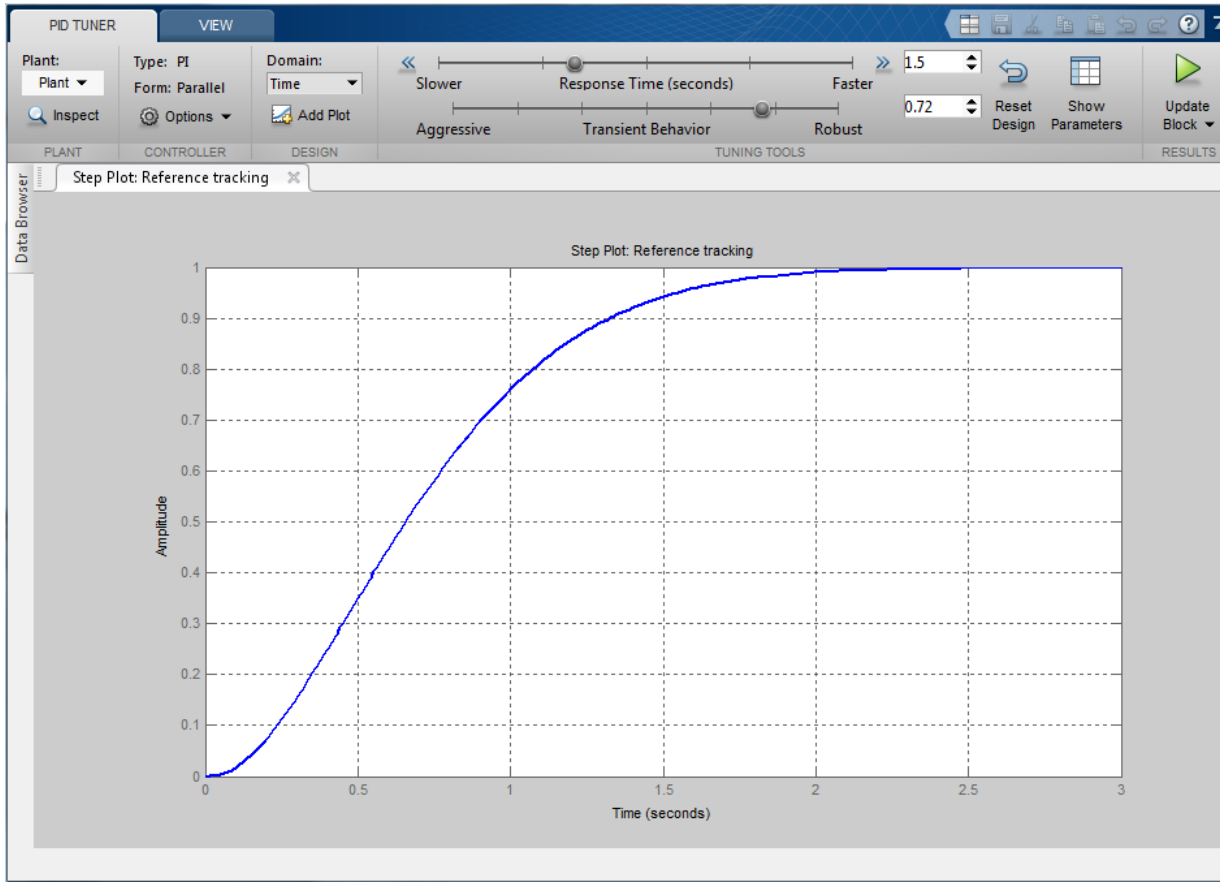
Performance and robustness

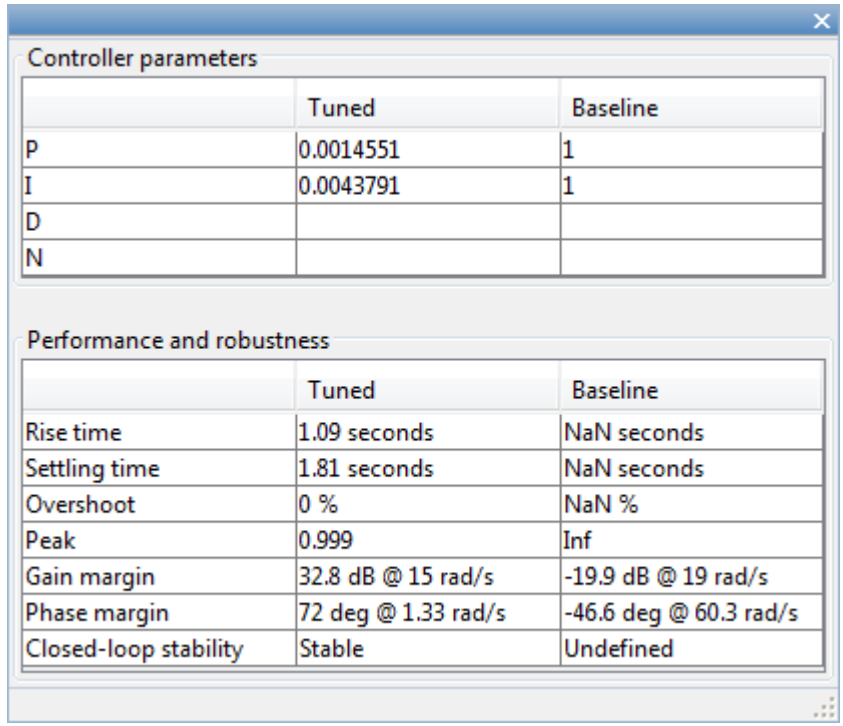
	Tuned	Baseline
Rise time	2.06 seconds	NaN seconds
Settling time	3.45 seconds	NaN seconds
Overshoot	0.401 %	NaN %
Peak	1	Inf
Gain margin	18.9 dB @ 3.27 rad/s	-19.9 dB @ 19 rad/s
Phase margin	69.3 deg @ 0.645 rad/s	-46.6 deg @ 60.3 rad/s
Closed-loop stability	Stable	Undefined

Completing PID Design with Performance Trade-Off

In order to achieve zero overshoot while reducing the settling time below 2 seconds, you need to take advantage of both sliders. You need to make control response faster to reduce the settling time and increase the robustness to reduce the overshoot. For example, you can reduce the response time from 3.4 to 1.5 seconds and increase robustness from 0.6 to 0.72.

The following figure shows the closed-loop response with these settings:


PID TUNER
VIEW
Plant:
Plant
Inspect
Type: PI
Form: Parallel
Options
Domain:
Time
Add Plot
Slower
Response Time (seconds)
Faster
1.5
Aggressive
Transient Behavior
Robust
0.72
Reset Design
Show Parameters
Update Block
PLANT
CONTROLLER
DESIGN
TUNING TOOLS
RESULTS
Data Browser
Step Plot: Reference tracking
Step Plot: Reference tracking
Amplitude
Time (seconds)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.5
1
1.5
2
2.5
3



Controller parameters

	Tuned	Baseline
P	0.0014551	1
I	0.0043791	1
D		
N		

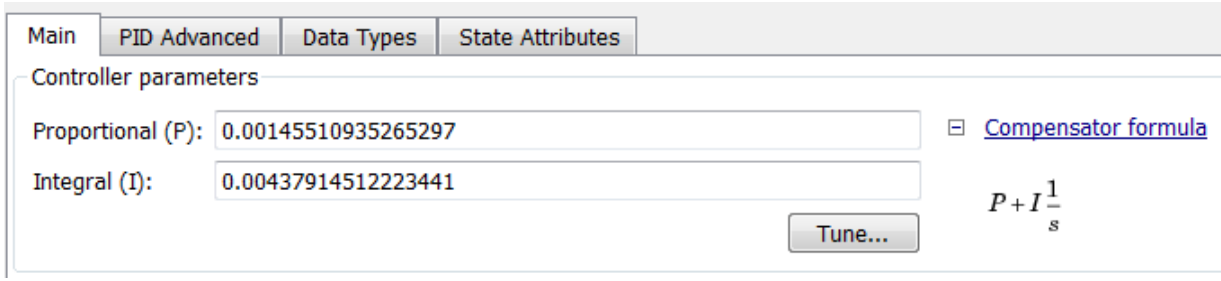
Performance and robustness

	Tuned	Baseline
Rise time	1.09 seconds	NaN seconds
Settling time	1.81 seconds	NaN seconds
Overshoot	0 %	NaN %
Peak	0.999	Inf
Gain margin	32.8 dB @ 15 rad/s	-19.9 dB @ 19 rad/s
Phase margin	72 deg @ 1.33 rad/s	-46.6 deg @ 60.3 rad/s
Closed-loop stability	Stable	Undefined

Writing the Tuned Parameters to PID Controller Block

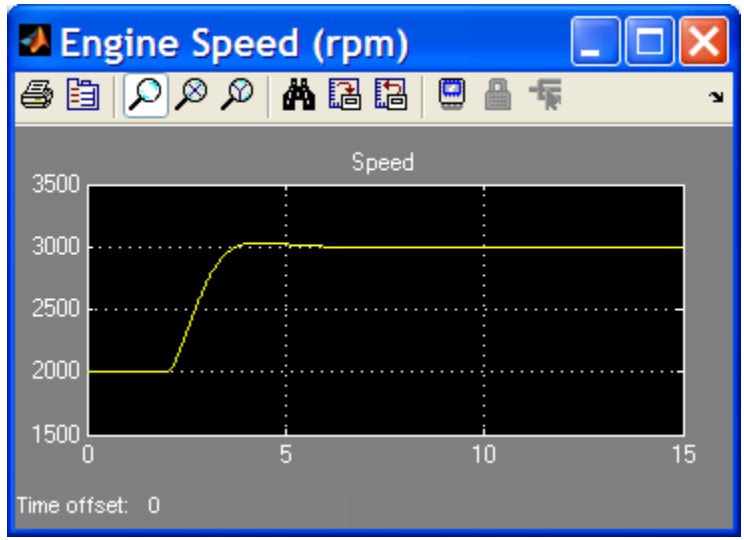
After you are happy with the controller performance on the linear plant model, you can test the design on the nonlinear model. To do this, click **Update Block** in the PID Tuner. This action writes the parameters back to the PID Controller block in the Simulink model.

The following figure shows the updated PID Controller block dialog:



Completed Design

The following figure shows the response of the closed-loop system:



The response shows that the new controller meets all the design requirements.

You can also use the Control System Designer to design the PID Controller block, when the PID Controller block belongs to a multi-loop design task. See the example "Single Loop Feedback/Prefilter Compensator Design".

```
bdclose('scdspeedctrlpidblock')
```

See Also

PID Tuner

More About

- “Tune PID Controller to Favor Reference Tracking or Disturbance Rejection”
- “Analyze Design in PID Tuner”