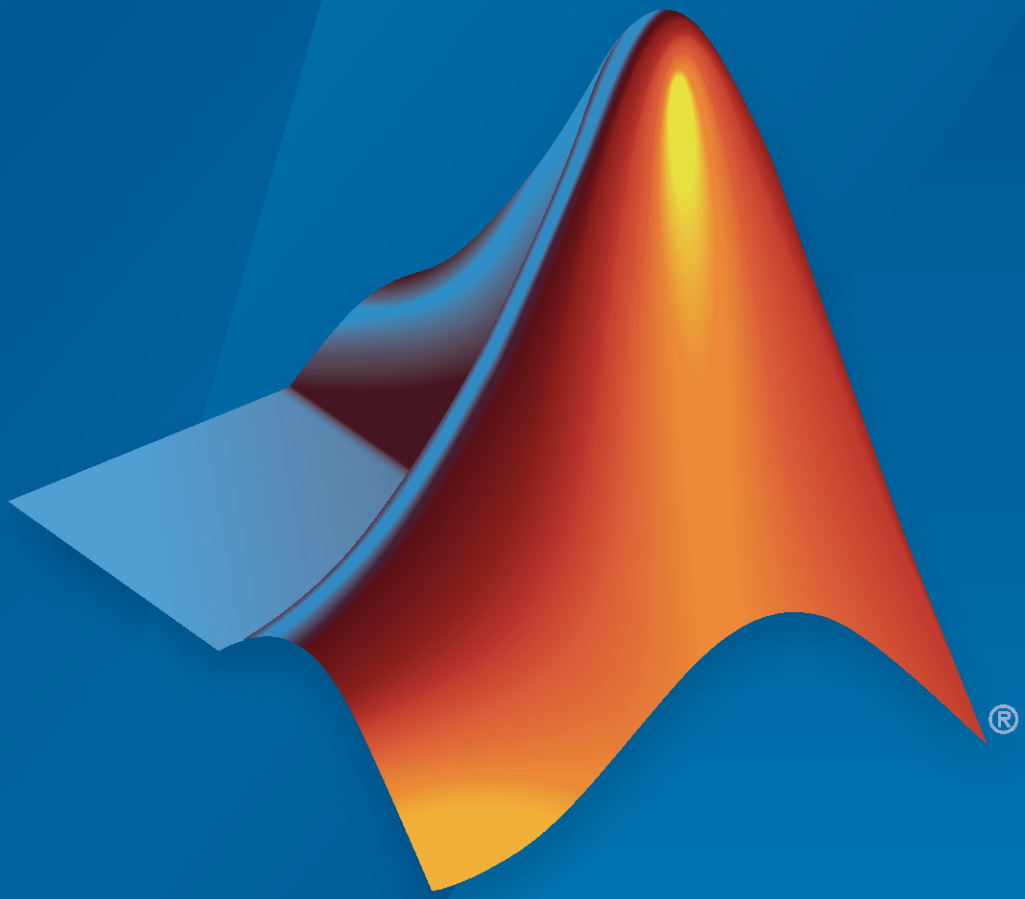
Simulink® Control Design™

Getting Started Guide



MATLAB® & SIMULINK®

R2022b

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.
1 Apple Hill Drive
Natick, MA 01760-2098

Simulink® Control Design™ Getting Started Guide

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)
March 2018	Online only	Revised for Version 5.1 (Release 2018a)
September 2018	Online only	Revised for Version 5.2 (Release 2018b)
March 2019	Online only	Revised for Version 5.3 (Release 2019a)
September 2019	Online only	Revised for Version 5.4 (Release 2019b)
March 2020	Online only	Revised for Version 5.5 (Release 2020a)
September 2020	Online only	Revised for Version 5.6 (Release 2020b)
March 2021	Online only	Revised for Version 5.7 (Release 2021a)
September 2021	Online only	Revised for Version 6.0 (Release 2021b)
March 2022	Online only	Revised for Version 6.1 (Release 2022a)
September 2022	Online only	Revised for Version 6.2 (Release 2022b)

Contents

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 also implement model-free extremum-seeking control and model reference adaptive control for applications where the controller must adapt to changing plant conditions.

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.

Steady-State Operating Points

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
- “Find Operating Points at Simulation Snapshots”

Steady-State Operating Points (Trimming) From Specifications

You can compute a steady-state operating point of a Simulink model by specifying constraints on the model states, outputs, and inputs, and finding a model operating condition that satisfies these constraints. You can trim your model to meet any combination of state, input, or output specifications. Computing an operating point in this way is called *trimming*. For more information on steady-state operating points, see “About Operating Points”.

You can trim your Simulink model:

- In the **Steady State Manager**. For an example, see “Compute Operating Points from Specifications Using Steady State Manager”.
- At the command line. For more information, see “Compute Operating Points from Specifications at the Command Line”.
- In the **Model Linearizer**. For more information, see “Compute Operating Points from Specifications Using Model Linearizer”.

For more information on selecting a trimming tool, see “Compute Steady-State Operating Points”.

For state specifications, you can constrain the values of model states to known values or ranges. You can also define bounds for 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 an example that trims a model for state specifications, see “Compute Operating Points from Specifications Using Model Linearizer”.

You can constrain the values of any root-level input or output ports to known values or ranges. You can also add output specifications to signals in your Simulink model. For an example that adds an output specification in this way, see “Compute Operating Points from Specifications Using Steady State Manager”.

If your trimming is unsuccessful; that is, if the optimization search was unable to meet all of your specifications, determine the specifications that could not be met by validating your trimmed operating point against the original specifications. For more information, see “Validate Operating Point Against Specifications”.

After trimming your model, you can:

- Linearize your model at the resulting operating point. For more information, see “Linearize at Trimmed Operating Point”.
- Simulate your model at the resulting operating point. For more information, see “Simulate Simulink Model at Specific Operating Point”.

See Also

Functions
`findop` | `findopOptions`

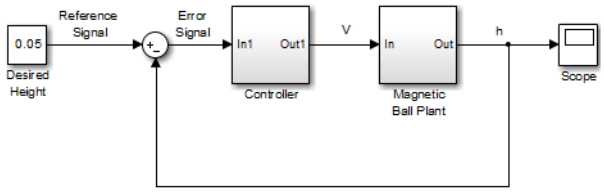
Apps
Steady State Manager | **Model Linearizer**

More About

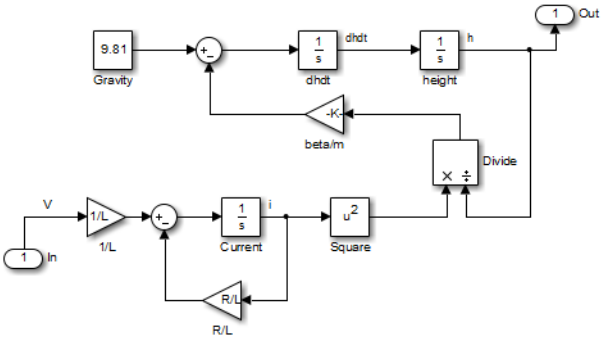
- “Compute Operating Points from Specifications at the Command Line”
- “Compute Operating Points from Specifications Using Steady State Manager”
- “Compute Operating Points from Specifications Using Model Linearizer”

magball Simulink Model

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

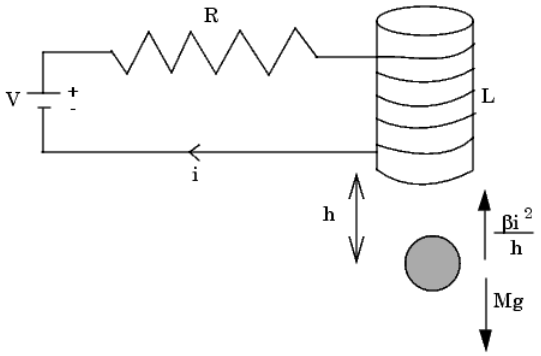


The Magnetic Ball Plant subsystem is shown in the following figure.



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

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”

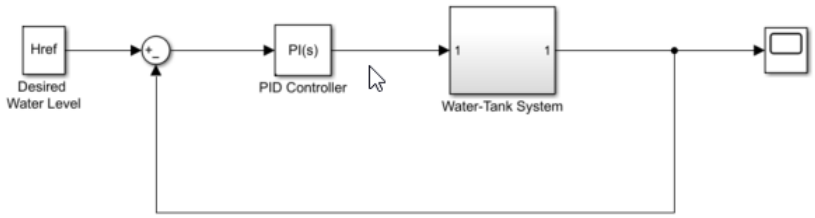
Open-Loop Response of Control System for Stability Margin Analysis

This example shows how to analyze the open-loop response of a control system using the **Model Linearizer**.

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)
```

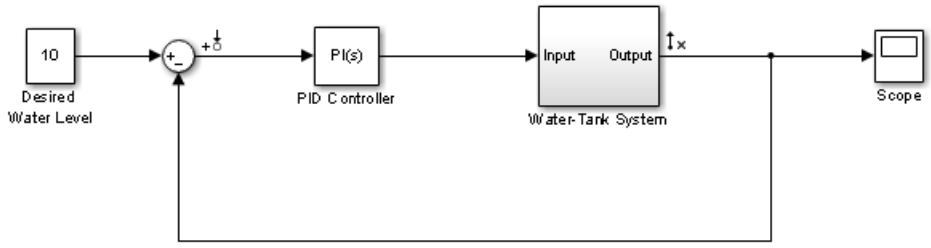


The Water-Tank System block represents the plant in this control system and contains all of the system nonlinearities.

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

1. Open the **Linearization** tab. To do so, in the **Apps** gallery, click **Linearization Manager**.
2. To specify an analysis point for a signal, click the signal in the model. Then, on the **Linearization** tab, in the **Insert Analysis Points** gallery, select the type of analysis point.
   - Configure the input signal of the PID Controller block as an **Input Perturbation**.
   - Configure the output signal of the Water-Tank System block as an **Open-loop Output**.

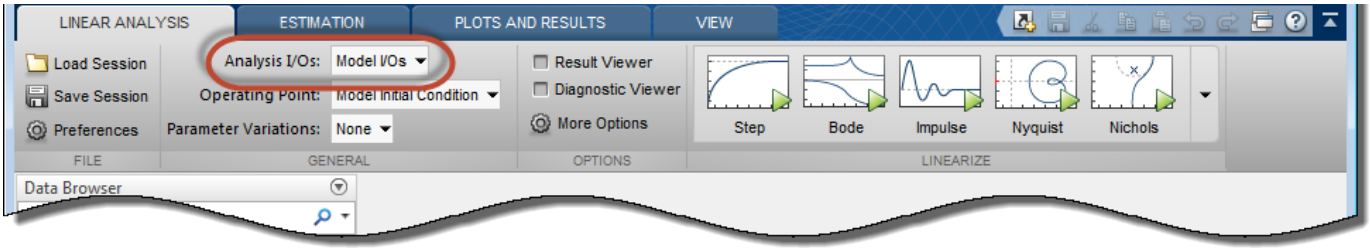
Annotations appear in the model indicating which signals are designated as analysis points.

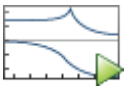


Tip If you do not want to introduce changes to the Simulink model, you can specify the analysis points in the **Model Linearizer**. For more information, see “Specify Portion of Model to Linearize in Model Linearizer”.

Open the **Model Linearizer** for the model. In the Simulink model window, in the **Apps** gallery, click **Model Linearizer**.

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

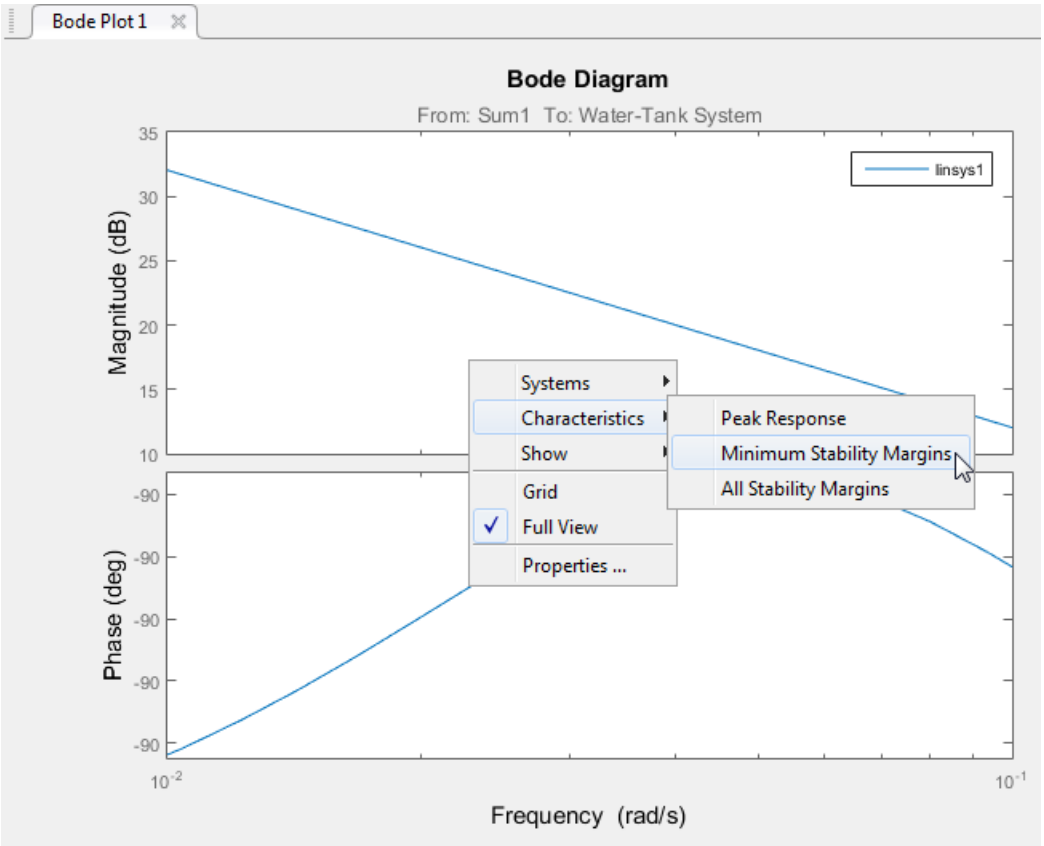


To linearize the model using the specified analysis points and generate a Bode plot of the linearized model, click  **Bode**.

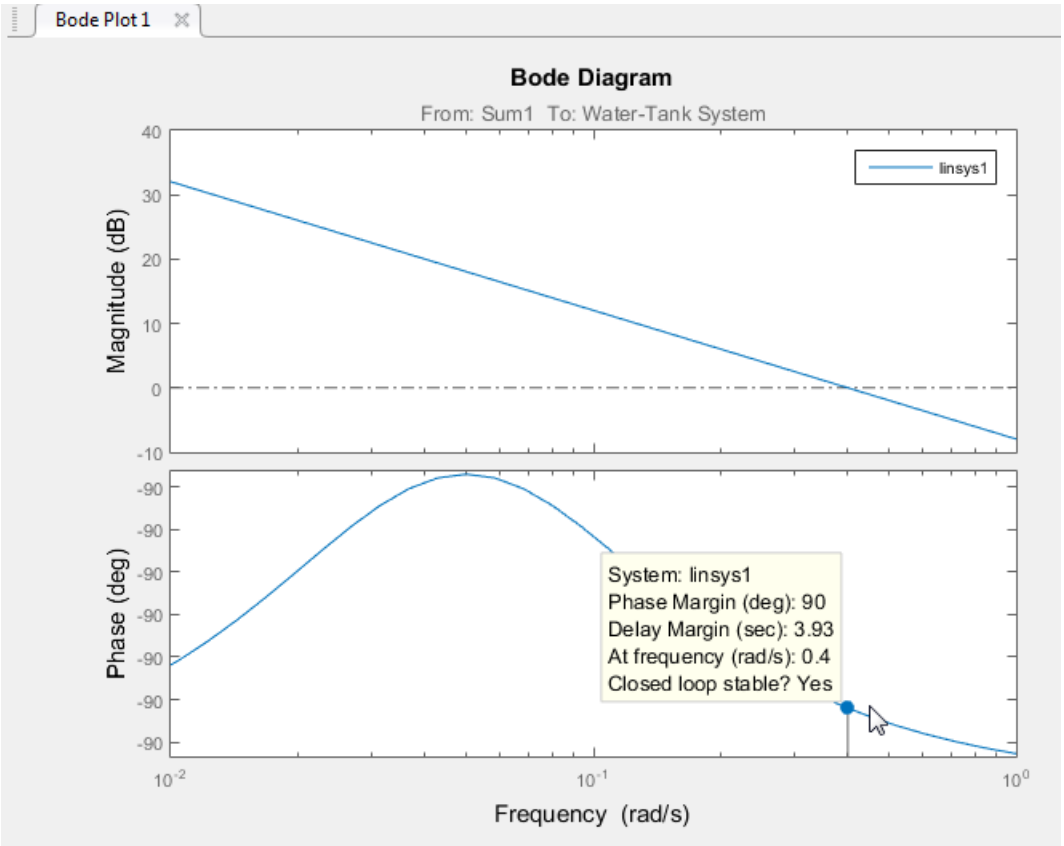
By default, the **Model Linearizer** linearizes the model at the model initial conditions, as shown in the **Operating Point** drop-down list. For examples of linearizing a model at a different operating point, see “Linearize at Trimmed Operating Point” and “Linearize at Simulation Snapshot”.

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

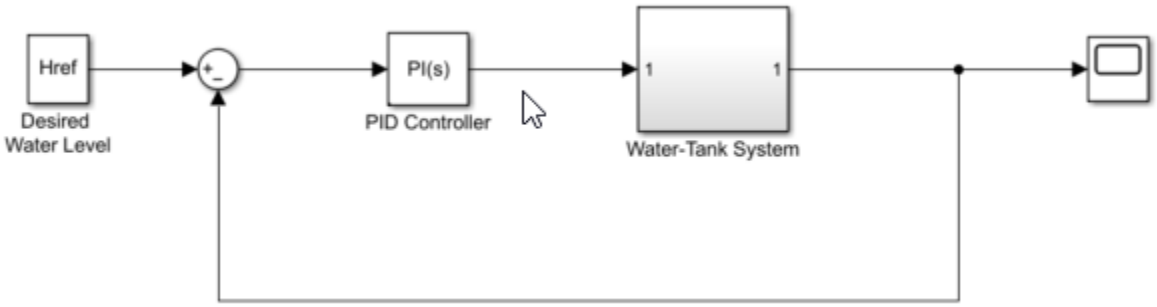
- “Linearize Nonlinear Models”
- “watertank Simulink Model” on page 3-12

Bode Response of Simulink Model

This example shows how to linearize a Simulink model at the operating point specified in the model using the **Model Linearizer**.

Open Simulink model.

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



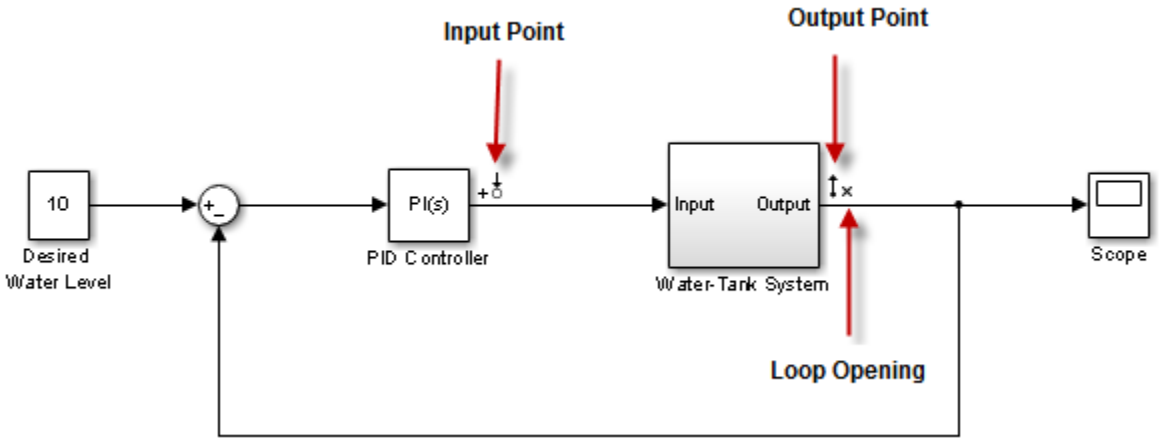
The Water-Tank System block represents the plant in this control system and includes all of the system nonlinearities.

To specify the portion of the model to linearize, first open the **Linearization** tab. To do so, in the Simulink window, in the **Apps** gallery, click **Linearization Manager**.

To specify an analysis point for a signal, click the signal in the model. Then, on the **Linearization** tab, in the **Insert Analysis Points** gallery, select the type of analysis point.

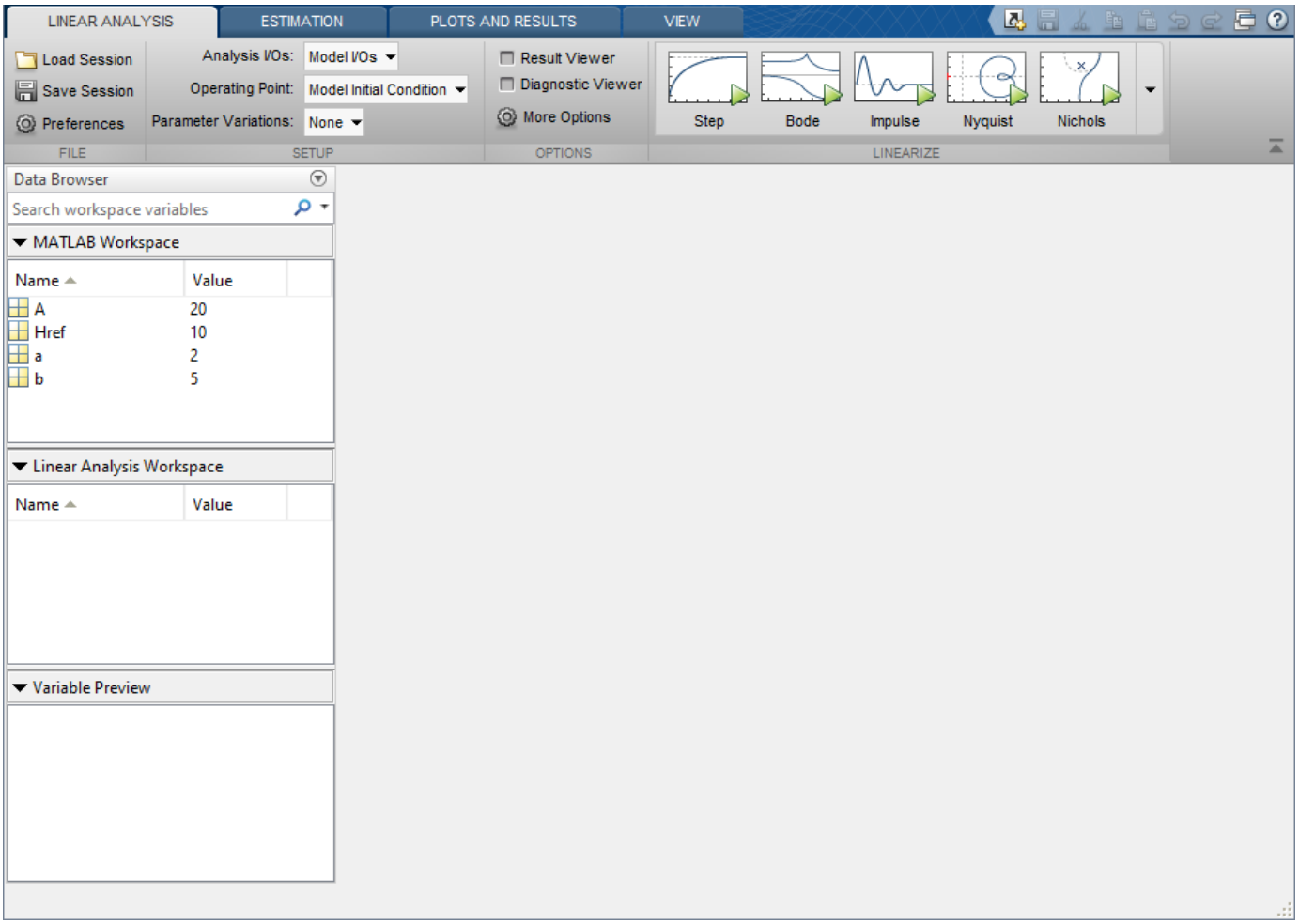
- Configure the output signal of the PID Controller block as an **Input Perturbation**.
- Configure the output signal of the Water-Tank System block as an **Open-loop Output**. An open-loop output point is an output measurement followed by a loop opening, which removes the effects of the feedback signal 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”.



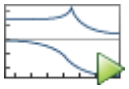
For more information on defining analysis points in a Simulink model, see “Specify Portion of Model to Linearize in Simulink Model”. Alternatively, if you do not want to introduce changes to the Simulink model, you can define analysis points using the **Model Linearizer**. For more information, see “Specify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

To open the **Model Linearizer** for the model, in the Simulink model window, in the **Apps** gallery, click **Model Linearizer**.

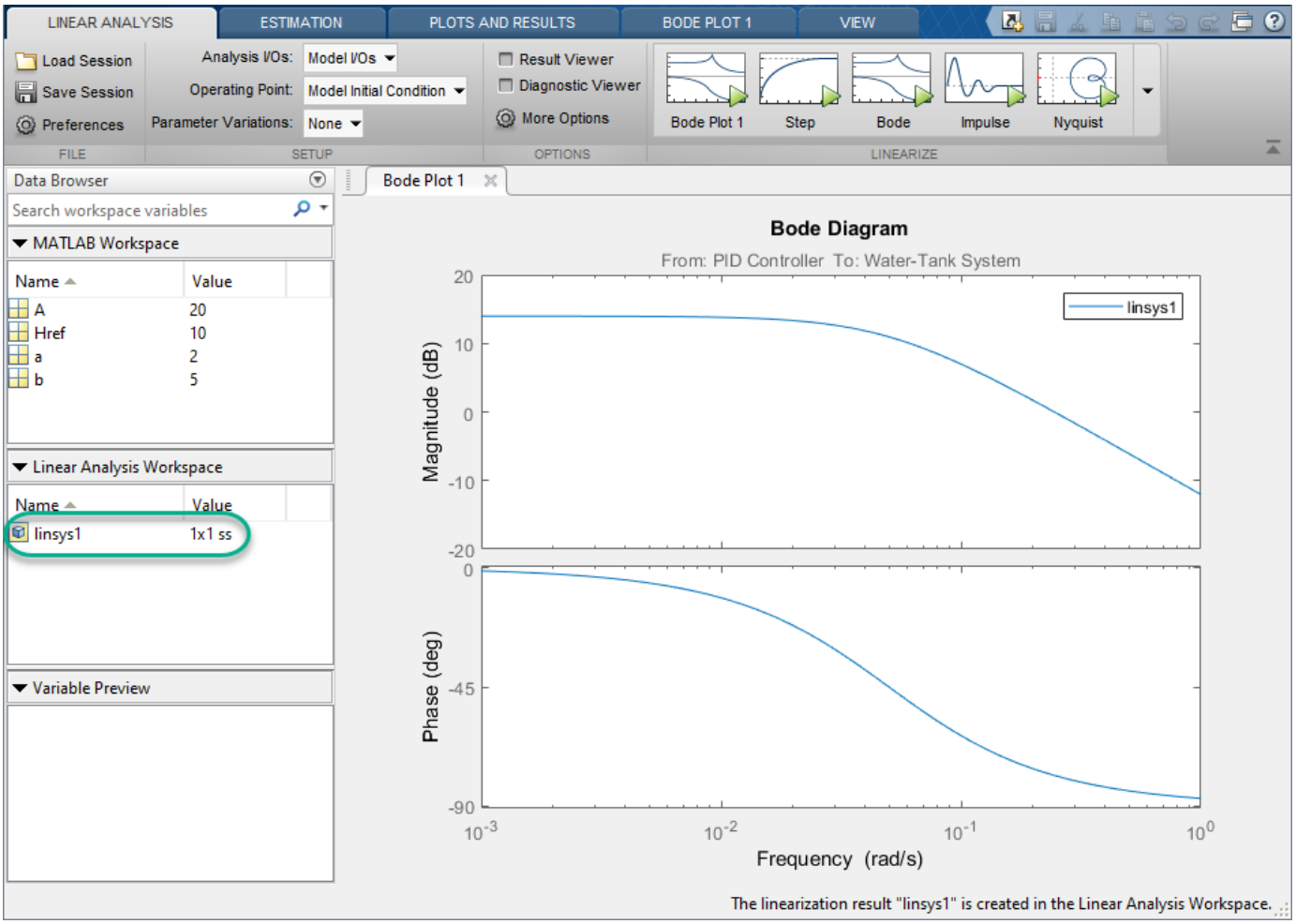


To use the analysis points you defined in the Simulink model as linearization I/Os, on the **Linear Analysis** tab, in the **Analysis I/Os** drop-down list, leave `Model I/Os` selected.

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

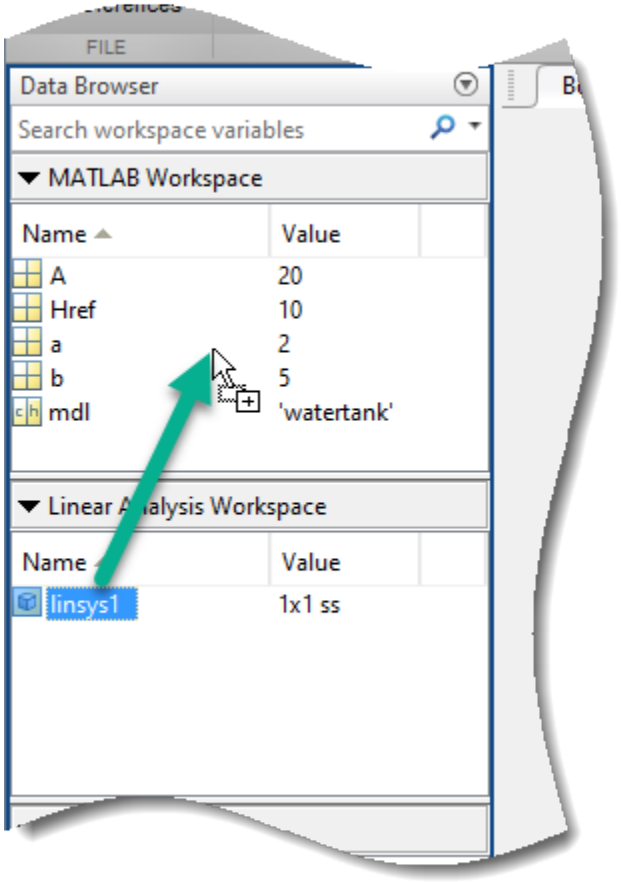
To linearize the system and generate a response plot for analysis, in the **Linearize** section, click a response. For this example, to generate a Bode plot for the resulting linear model, click  **Bode**.

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 Model Linearizer 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**.



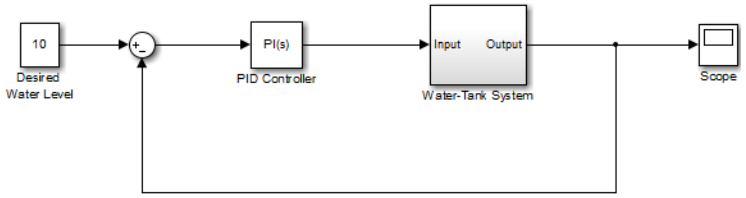
See Also

More About

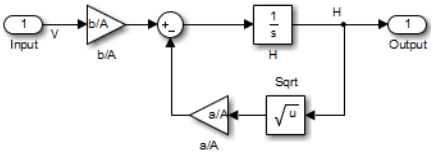
- “Linearize Nonlinear Models”
- “watertank Simulink Model” on page 3-12
- “Open-Loop Response of Control System for Stability Margin Analysis” on page 3-3
- “Steady-State Operating Points (Trimming) From Specifications” on page 2-3

watertank Simulink Model

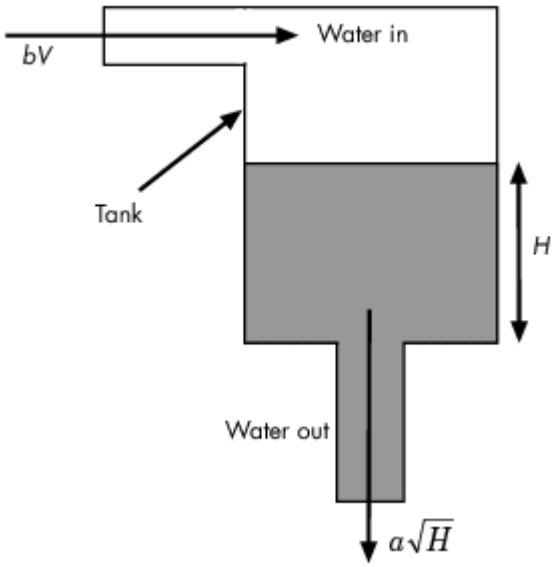
The Simulink model `watertank` includes the nonlinear Water-Tank System plant and a PI controller in a single-loop feedback system.



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

Trim and Linearize Simulink Models

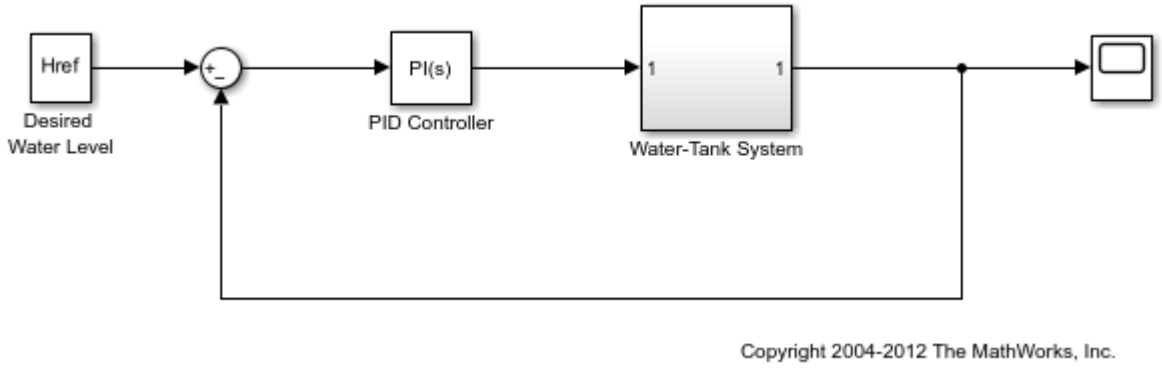
This example shows how to programmatically linearize a “watertank Simulink Model” on page 3-12 feedback control system. In the example, you obtain an open-loop linearized model of the water-tank system at an operating point where the tank level is at a steady state.

For more information on programmatically specifying input and output points for linearizing a model, see “Specify Portion of Model to Linearize” and “Specify Portion of Model to Linearize at Command Line”.

For more information on finding operating points for linearization, see “Compute Steady-State Operating Points from Specifications” and “Compute Operating Points from Specifications at the Command Line”.

Open the model.

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



In this model, there is an expected steady-state operating condition when the water level is at `H = 10`.

Compute Operating Point

To linearize a model, you must first obtain an operating point for the conditions at which you want to linearize the model. One approach is to first simulate the model and extract an operating point when the simulation is near the desired value. You can then use this operating point as a starting point for an optimization-based search (trimming) for a steady-state operating point.

Using the `findop` function, simulate the model and obtain an operating point using the model conditions after `10` seconds.

```
opsim = findop(mdl,10)
```

```
opsim =


 Operating point for the Model watertank.
 (Time-Varying Components Evaluated at time t=10)
```

```
States:
----------
   x
_______

(1.) watertank/PID Controller/Integrator/Continuous/Integrator
1.6949
(2.) watertank/Water-Tank System/H
10.0796

Inputs: None
----------
```

In this operating point, `H` is not at the desired value of `10`. However, you can use this operating point to initialize a search for an operating point where `H = 10`.

To configure the operating point search, first create an operating point specification object.

```
opspec = operspec(mdl);
```

Initialize the values of the states in the operating point specification with the state values in the operating point `opsim`.

```
opspec = initopspec(opspec,opsim);
```

Trim the model using the operating point specifications.

```
opss = findop(mdl,opspec);


 Operating point search report:
---------------------------------

opreport =


 Operating point search report for the Model watertank.
 (Time-Varying Components Evaluated at time t=10)

Operating point specifications were successfully met.
States:
----------
    Min          x          Max        dxMin        dx         dxMax
___________ ___________ ___________ ___________ ___________ ___________

(1.) watertank/PID Controller/Integrator/Continuous/Integrator
   -Inf       1.2649        Inf          0           0           0
(2.) watertank/Water-Tank System/H
     0          10          Inf          0      -1.0991e-14      0

Inputs: None
----------

Outputs: None
----------
```

In this operating point, `H = 10` as expected. The operating point is at a steady state since the `dx` values for the model states are near zero.

Configure Linear Analysis Points

To linearize the model, you must specify the portion of the model that you want to linearize. Linear analysis points specify the inputs and outputs of a linearized model. To extract the open loop linearized model of the water-tank plant, add an input point at the output of the Controller block and an output point, with a loop opening, at the output of the Water-Tank System block.

Specify the input point.

```
watertank_io(1) = linio('watertank/PID Controller',1,'input');
```

Specify the output point with a loop opening.

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

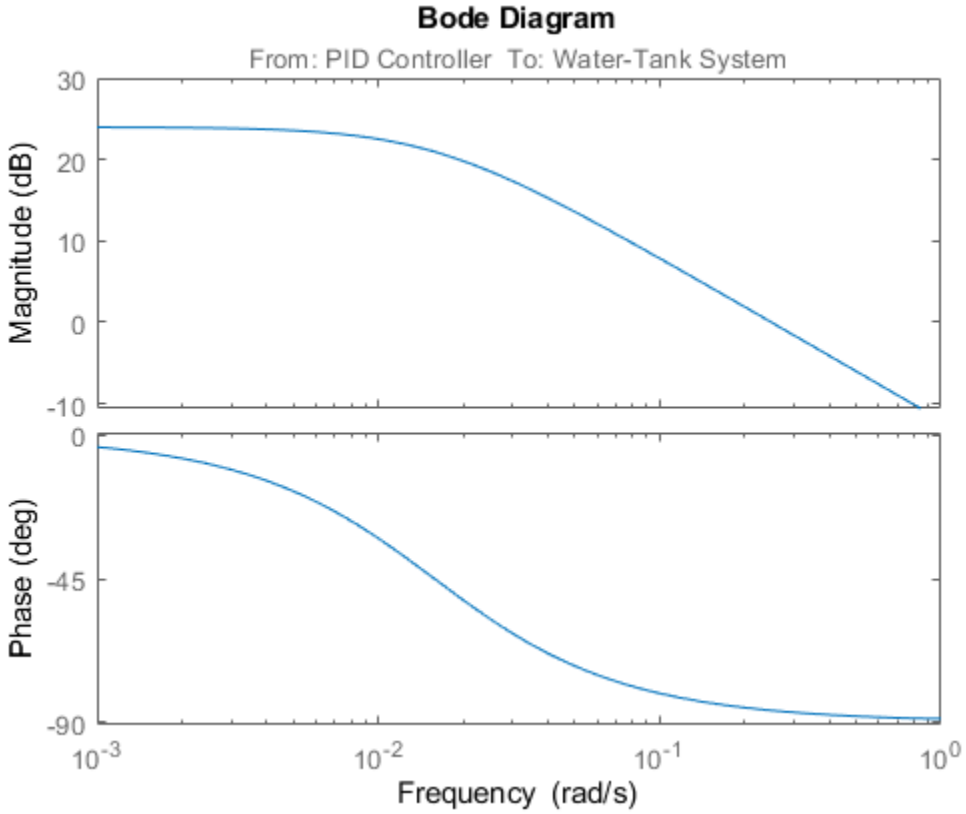
Linearize and Analyze Model

You can now linearize the model using the specified operating point and linear analysis points.

```
sys = linearize(mdl,opss,watertank_io);
```

The resulting model is a state-space object that you can analyze using any of the tools in the Control System Toolbox™ software. For example, view the frequency response of the linear model.

```
bode(sys)
```



Close the Simulink® model.

```
bdclose(mdl)
```

See Also

`findop` | `linearize`

Related Examples

- “Linearize Nonlinear Models”
- “Linearize at Trimmed Operating Point”
- Trimming and Linearization Part 1: What is Linearization?
- Trimming and Linearization Part 2: The Practical Side of Linearization

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.

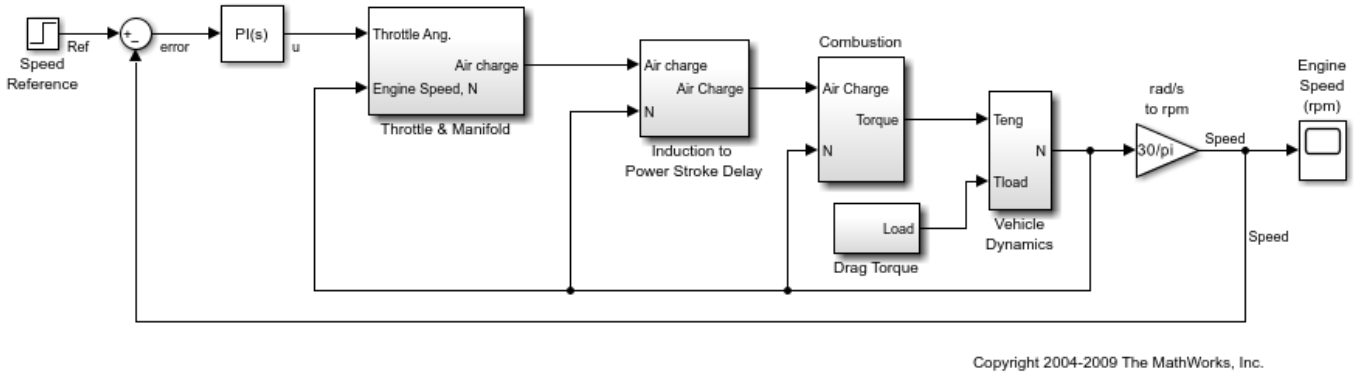
(2) Tune the controller in the **PID Tuner** by manually adjusting design criteria in two design modes. The tuner computes PID parameters that robustly stabilize the system.

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

Open 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

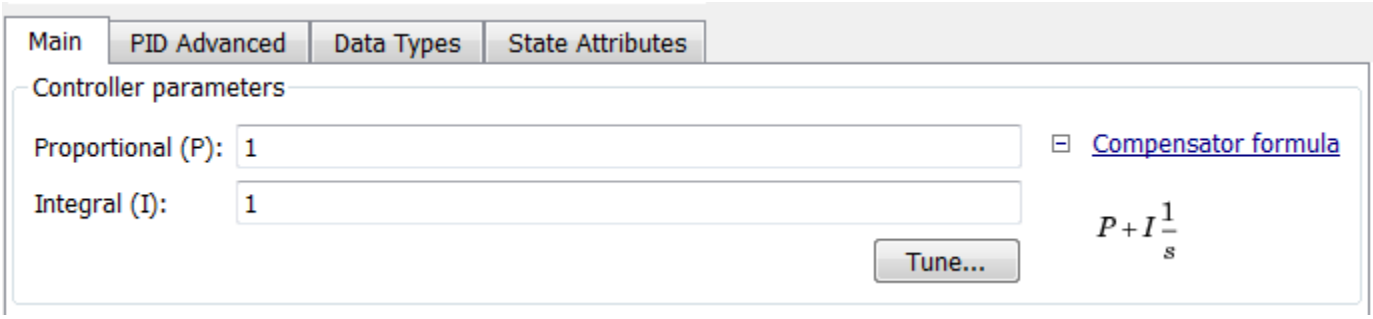
In this example, you design a PI controller in an engine speed control loop. The goal of the design is to track the reference signal from a Simulink step block `scdspeedctrlpidblock/Speed Reference`. The design requirement are:

- 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**.

Open PID Tuner

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

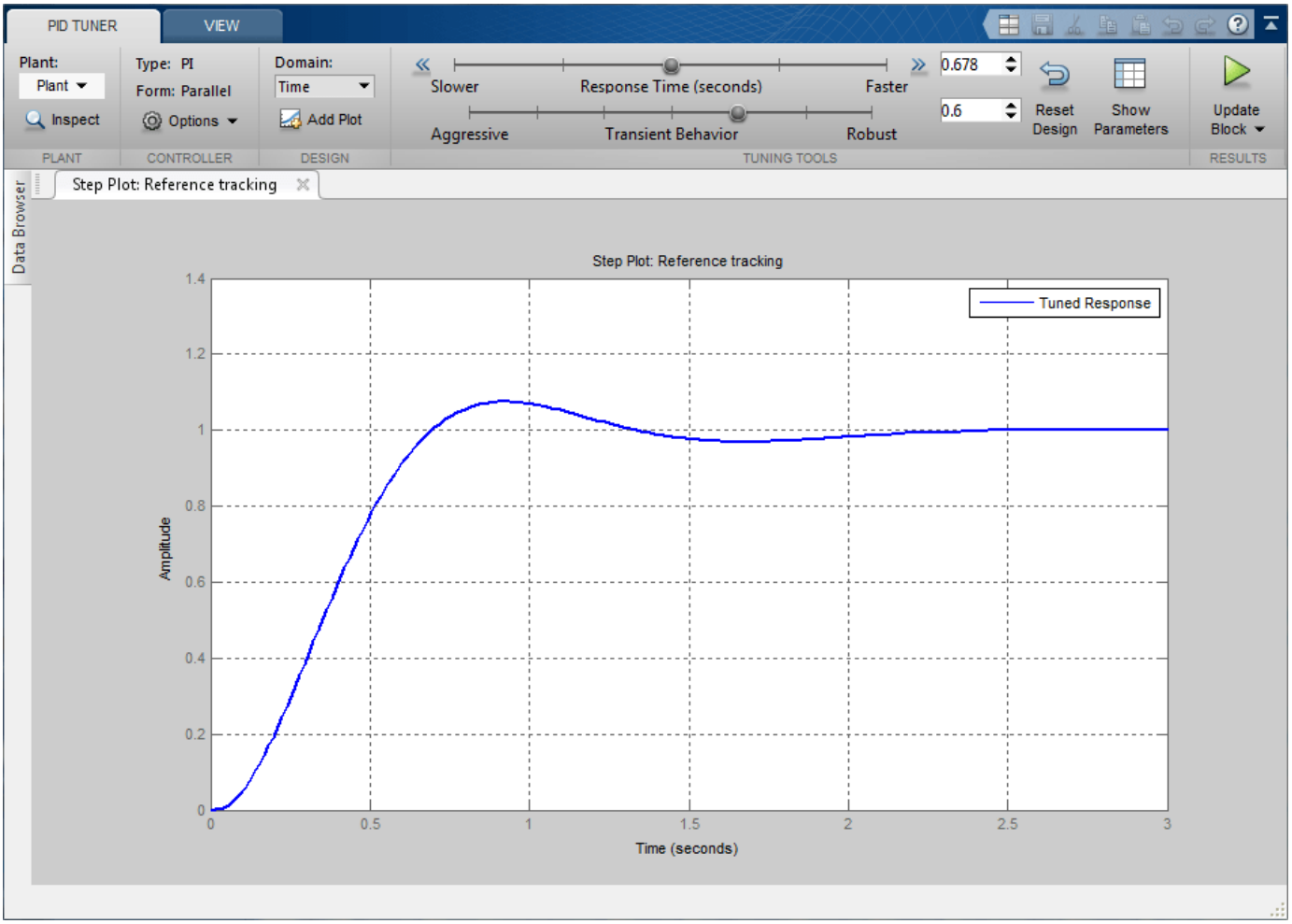


Initial PID Design

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.

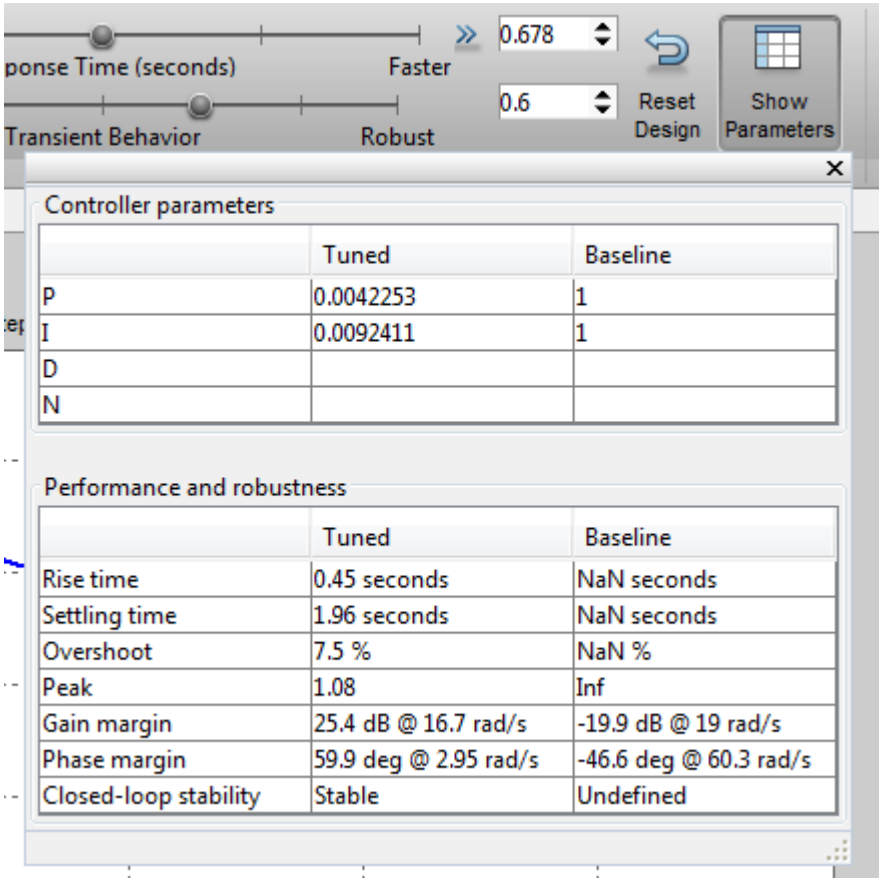
The **PID Tuner** computes an initial PI controller to achieve a reasonable tradeoff between performance and robustness. By default, step reference tracking performance displays in the plot.

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



Display PID Parameters

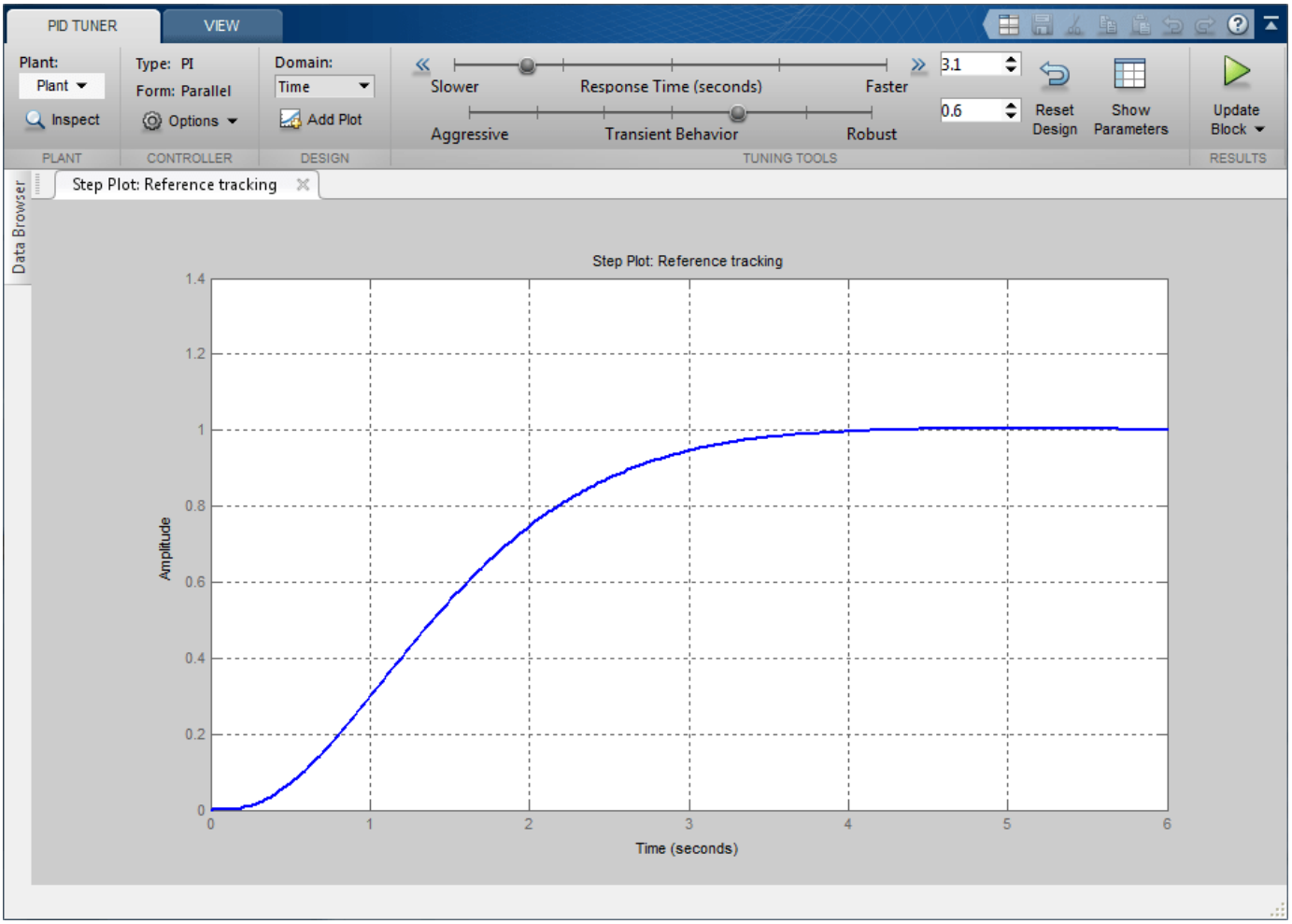
Click **Show parameters** to view controller parameters P and I, and a set of performance and robustness measurements. In this example, the initial PI controller design gives a settling time of 2 seconds, which meets the requirement.

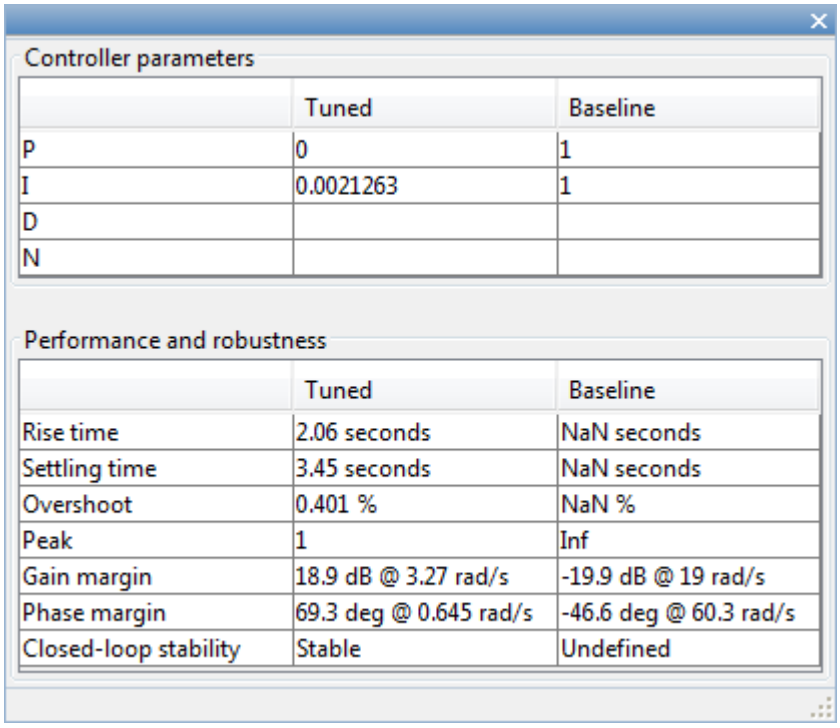


Adjust PID Design in PID Tuner

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
Type: PI
Form: Parallel
Options
Domain:
Time
Add Plot
Slower
Response Time (seconds)
Faster
Aggressive
Transient Behavior
Robust
3.1
0.6
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
0.2
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		

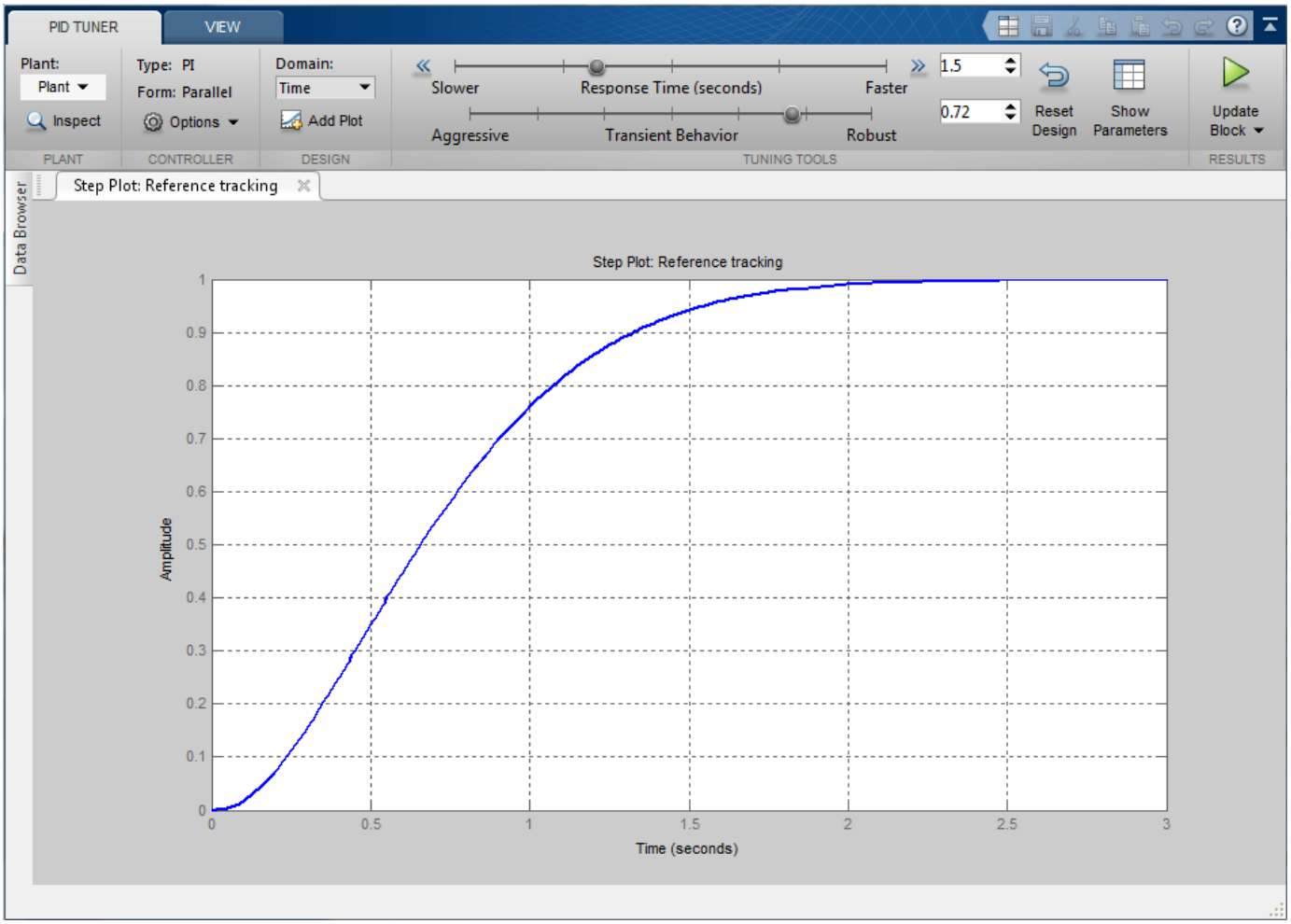
Performance and robustness

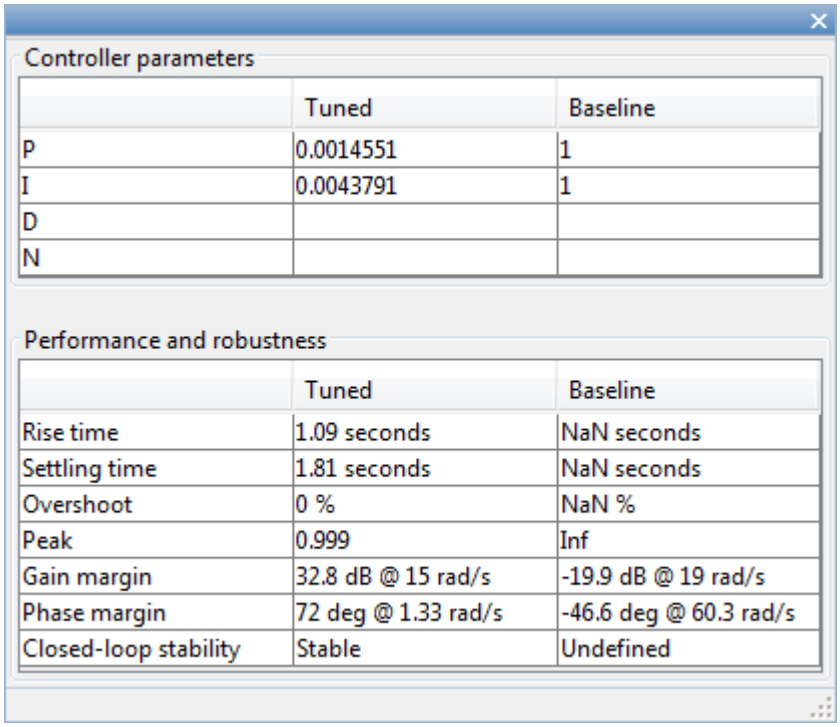
	Tuned	Baseline
Rise time	2.06 seconds	NaN seconds
Settling time	3.45 seconds	NaN seconds
Overshoot	0.401 %	NaN %
Peak	1	Inf
Gain margin	18.9 dB @ 3.27 rad/s	-19.9 dB @ 19 rad/s
Phase margin	69.3 deg @ 0.645 rad/s	-46.6 deg @ 60.3 rad/s
Closed-loop stability	Stable	Undefined

Complete 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
0.5
1
1.5
2
2.5
3



Controller parameters

	Tuned	Baseline
P	0.0014551	1
I	0.0043791	1
D		
N		

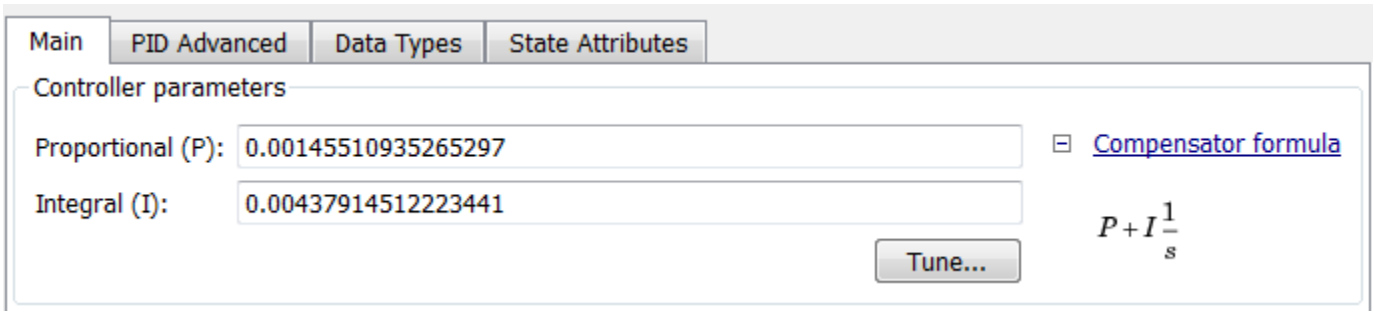
Performance and robustness

	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

Write Tuned Parameters to PID Controller Block

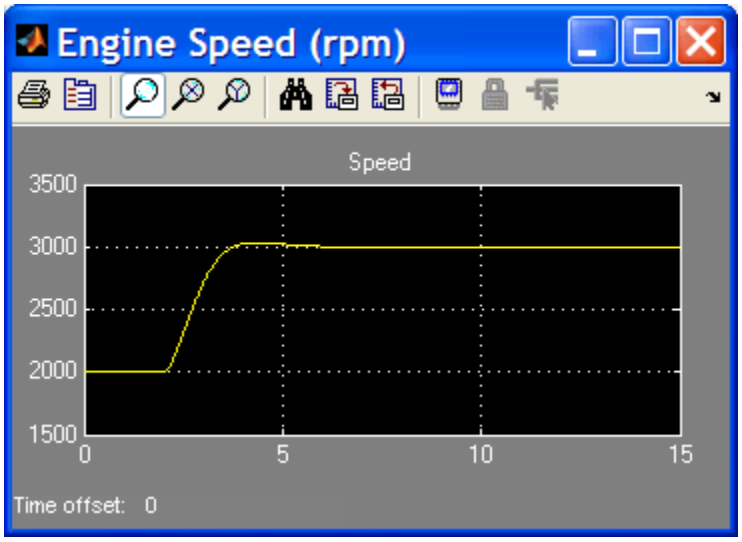
After you are happy with the controller performance on the linear plant model, you can test the design on the nonlinear model. To do this, click **Update Block** in the **PID Tuner**. This action writes the parameters back to the PID Controller block in the Simulink model.

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”