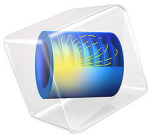

PID Controller

Introduction

A PID (proportional-integral-derivative) controller is the most common form of feedback controller (if you exclude simple on-off controllers). It continuously calculates the error, $e(t) = c_{\text{set}} - c(t)$, between a setpoint variable, c_{set}, and the measured variable output, $c(t)$. The control signal, u, is generated from three terms:

- One that is proportional to the error (P).
- One that is proportional to the integral of the error (I).
- One that is proportional to the derivative of the error (D).

The proportionality constants are chosen so that the controlled system settles quickly. The integral term is needed to eliminate a steady-state error. The derivative terms can be sensitive to noise and is therefore often turned off or modified with a high-frequency filter. Also, the setpoint is typically not part of the derivative term (its derivative is zero except for changes in the setpoint).

In its most general form, which includes a bias constant u_{bias}, the algorithm for the PID controller provided by this add-in reads:

$$u(t) = u_{\text{bias}} + k_{\text{p}}[c_{\text{set}} - c(t)] + k_{\text{i}}\int_0^t [c_{\text{set}} - c(\tau)]d\tau - k_{\text{d}}\frac{d}{dt}c(t)$$

You can use the PID controller to control any model variable or parameter. The requirements for using the add-in is that the model contains at least one component with some physics to control. You define the measured variable output $c(t)$ by adding a probe, typically a domain point probe or boundary point probe, representing a process measurement. Once the add-in has been imported from the COMSOL Multiphysics Add-in Library, it can be added to the model from the **Add-ins** menu on the **Developer** tab.

Add-in Library path: `COMSOL_Multiphysics/pid_controller`

Once you have defined all the controller parameters, click the **Create** button ( ) to add a 0D component that implements the controller using global equations. When the creation of the PID controller is successful, you get a message about the available control variable (such as `comp2.u_in_ctrl`) and the deduced unit for the controller. It is important to make sure that all controller quantities have compatible units; otherwise, a unit consistency error appears.

EQUATION

This section displays the control algorithm and defines the names of the variables and constants that you can specify.

COMPONENT SELECTION

Select the desired component from the **Component** list. When applying the PID controller, a new 0D component will be added to your model with variables and global equations that implement the PID controller, and you will be presented with the name of the control variable. This variable can then be used in your model as the actuator; for example, the velocity at an inlet for a transport model.

MEASURED VARIABLE OUTPUT

Before applying the PID controller, you need to define a probe that measures the controlled variable output. When available, select it from the **Probe** list.

If the selected probe is a Domain Point Probe or a Boundary Point Probe, also select the desired expression from the **Probe expression** list.

CONTROLLER PARAMETERS

Select the **Bias** check box to include a bias that you enter in the $\mathbf{u_{bias}}$ text field. If the **Bias** check box is cleared, the bias is ignored.

Define the controller constants $\mathbf{k_p}$, $\mathbf{k_i}$, and $\mathbf{k_d}$ in the **Proportional gain**, **Integral gain**, and **Derivative gain** text field, respectively. Leaving $\mathbf{k_d}$ at its default zero value gives a PI controller. Defining these controller constants as global parameters makes it easy to change them without recreating the PID controller.

In the **Reference value** text field you specify the setpoint, $\mathbf{c_{set}}$.

Select or clear the **Lower limit** and **Upper limit** check boxes to restrict the lower and upper limits of the control value to the values $\mathbf{u_{min}}$ and $\mathbf{u_{max}}$, respectively.

ANTI-WINDUP AND PREFILTERING

Select the **Enable integral anti-windup** check box and enter a time constant T_t (SI unit: s) in the **Time constant** text field to enable integral anti-windup. Integral windup can occur when the control variable (actuator) reaches a limit. The control error will then continue to be integrated and become very large (that is, it "winds up"). This phenomenon means that an actuator saturation can cause large transients for a controller with an integral action. The anti-windup scheme keeps the integrator at a value so that the controller output is exactly at the limit so that no windup occurs. You can control the amount of anti-windup with the time constant; the smaller the value of T_t, the faster the reset of the integrator, but a too small value may cause some loss of stability when a derivative action is included in the PID controller.

Select the **Enable filtering of derivative** check box and enter a time constant T_f (SI unit: s) in the **Time constant** text field to enable filtering of the derivative term. Such a filter works as a high-frequency filter that avoids that the derivative term becomes large due to high-frequency noise and instead only acts as a derivative for low-frequency components. The larger the time constant, the more filtering is added for the high-frequency noise.

INFORMATION

This section displays information about the name of the control variable and the deduced unit for the controller. It also contains warning messages, if present.