The Process PID control tuner provides the open and closed loop process system responses for a continuous process model \( G \) with a continuous PID controller \( G_c \). The Process model can be characterized by the numerator and denominator polynomial coefficients of the process transfer function model. Delay time \( (t_d) \) can be specified to the process transfer function. The PID controller tuning parameters are the control gain \( (K_c) \), integral action \( (\tau_i) \) and derivative action \( (\tau_d) \). The PID controller has also the derivative action-filter parameter that attenuates the effect of the derivative action in the closed loop response, and makes realizable the controller transfer function.

The continuous process transfer function is denoted by \( G \):

\[
G = \frac{Y(s)}{U(s)} = \sum_{i=0}^{m} b_i s^i e^{-t_d s} = \frac{b_0 s^m + b_1 s^{m-1} + \cdots b_n}{a_n s^n + a_{n-1} s^{n-1} + \cdots a_0} e^{-t_d s} \quad (Eq. 4.40)
\]

The tuner open loop response represents the response of \( G(s) \) to a step on \( U(s) \). The adjustable step size represents the magnitude of the input step, i.e., \( u(t > 0) \). The next diagram illustrates the open loop block representation:

![Open Loop Block Diagram]

The PID controller structure is given by:

\[
G_c = \frac{U(s)}{E(s)} = K_c \left( 1 + \frac{1}{\tau_i s} + \frac{\tau_d s}{1 + \alpha s} \right) \quad (Eq. 8.14')
\]

Note that this PID structure resembles the Parallel form except for the incorporation of the derivative filter. The closer \( \alpha \) gets to zero, the more the equation above will resemble Eq. 8.14 of the book. Making \( \alpha=0 \) will result in a transfer function with a numerator polynomial order greater than the denominator.

The resulting closed loop is given by the following diagram:
And it is represented by the following transfer function:

$$Gcl = \frac{Y(s)}{SP(s)} = \frac{GcG}{1 + GcG}$$

where the sub-index “cl” stands for “closed loop”. The response for the closed loop transfer function is obtained in the process PID Control Tuner when the case “Closed Loop” is chosen. The closed loop input step is applied to the Set-point, SP(s), instead of the process input, U(s). This structure represents the closed loop servo problem as the controller is used to make the process output follow a set-point change (also known as reference). The comparison symbol ‘⊗’ shown in the block diagram compares the set-point to the process output, Y(s). The difference is the error E(s), which represents the input of the PID controller. This control configuration is also called the feedback structure, as the process output is brought to the input side of the diagram and compared against the set-point.

The PID controller parameters are adjusted to obtain a desirable closed loop response. The three main parameters used are the controller gain (K_c), the Integral (τ_i) and the Derivative time (τ_d). The main objective of the PID controller is to bring the process output as close as possible to the set-point in a reasonable amount of time, that means, it needs to bring the error E(s) closer to zero in a reliable and promptly manner.

A controller tends to be more aggressive when a slight change in the error E(s) triggers a considerable change in its output U(s). Moreover, the larger the K_c magnitude, the more aggressive is the controller. The gain K_c and the process transfer function gain should have the same sign when no other block element besides the controller and the process is placed between the error E(s) and the output signal Y(s); as it is the case of the diagram shown above.

Why the process gain sign can be either positive or negative? It is simple: sometimes the process output increases when the input increases - positive process gain, like when more fuel flow increases the temperature of the combustion chamber. In other situations the process output decreases when its input increases. This is the case of using cooling water to control the temperature: the more cold water flow, the less is the process temperature.
In the case of a positive process gain, the PID controller should try to bring a negative error $E(s) < 0$, defined as

$$E(s) = SP(s) - Y(s)$$

towards zero by increasing $U(s)$. That means, to make $E(s) < 0$ increase, the controller should increase $U(s)$, which will eventually make $Y(s)$ increase. Controller with direct action will provide this type of behavior ($K_c > 0$).

In the case of a negative process gain, the PID controller should try to bring a positive error $E(s) > 0$ towards zero by increasing $U(s)$. That means, to make $E(s) > 0$ decrease, the controller should increase $U(s)$, which will eventually make $Y(s)$ decrease. Controller with reverse action will provide this type of behavior ($K_c < 0$).

Integral action is important to eliminate the steady state error, also known as offset. In other words, the controller will always tend to eliminate the offset when integral action is active. In the case of integral time, note in Eq. 8.14’ that its reciprocal is used in the controller expression. Therefore, the larger $\tau_i$ the least is the effect of the integral action. $\tau_i=0$ eliminates the integral action term in the controller.

The derivative action is rarely used as the controller output responds to changes in its input derivative (\(\frac{de}{dt}\)). Some problems in the implementation of the derivative action arise when the $Y(s)$ is noisy. Noisy signals tend to change derivative quite often. Because $E(s) = SP(s) - Y(s)$, if the output derivative changes, the error derivative will also change drastically. This effect will propagate the noise to process components like valves and actuators.

After this short background you are ready to use the Process PID Control Tuner to determine the best tuning parameters for your PID controller. Some performance specification of the closed loop response are based on the settling time (time that it is necessary for the output to be within 5% from the set-point based on the output response span), or the response overshoot (response pick measured from the set-point line). Figures 5.9 and 5.10 in the book illustrate the different closed loop performance specifications.

If you are looking to configure a PID controller based on how well it brings the process output back to its original set-point in the presence of a load disturbance, you are considering the regulator instead of the servo problem. The regulator problem is represented in the following diagram:
and its respective closed loop transfer function is given by:

\[ G_{cl} = \frac{Y(s)}{D(s)} = \frac{G}{1 + GcG} \]

where \( D(s) \) represents the load or disturbance. Note that the output \( Y(s) \) is the addition of two signals. Typical regulator problem example is the presence of wild flows in a mixing process. A tank process could be mixing a manipulated flow \( U(s) \) and a wild flow \( D(s) \). The controller main task will consist of adjusting the manipulated flow \( U(s) \) in order to reject the fluctuations in the tank concentration \( Y(s) \) due to changes in the disturbance or perturbation flow \( D(s) \).

Most of the PID tuning rules mentioned above apply for the servo as well as for the regulator problem. Note that he closed loop transfer function denominator is the same regardless of type of problem being considered.

An example VI (Virtual Instrument) has been developed in LabVIEW to simulate open-loop or closed-loop dynamic responses for process control systems under PID control. The VI is accessible through a web browser. The instructions below explain how to launch and navigate the tuner using an example.

Click on the Process_Tuner.html page (which has the LabVIEW VI running in a browser)

You’ll need the LabVIEW 8.6 RunTime engine (a free plug-in from NI). This only has to be done the first time. You don’t need LabVIEW itself installed. If you do have LabVIEW 8.6, the RunTime engine is already installed.
Process PID Control Tuner

Once you have LabVIEW RunTime installed, right-click on the security bar and select “Allow Blocked Content”.

Say Yes to the Security Warning.

The VI comes up already running in a browser, with the closed-loop response of a second order process.
The VI is organized into three areas:

- **Process and Disturbance parameters** – allows changes to the process transfer function, step size, time delay, final simulation time
- **Controller Tuning parameters** – allows changes to PID controller gains
- **Open and Closed-loop step response graphs**

There are two modes of operation: open-loop and closed-loop.

In the open loop case, note the controller tuning parameters are grayed out.
In the closed-loop case, the step can be applied to either the process setpoint or the disturbance:

To change the transfer function, you can enter new values into the numerator or denominator fields. To delete an element (decrease the order), simply right click on the value and select “Delete Element.”
The resulting transfer function would now be:

\[ G(s) = \frac{2}{50s + 15} \]

Note the coefficients are in descending order. Grayed-out values of 0 indicate an unused coefficient. If you enter a value of 0, the order will increase, as shown below.
The LabVIEW Control Design and Simulation module is using a Runge-Kutta 4 Fixed step size ODE solver to perform the simulation.

The solver type cannot currently be changed in the web version of the VI. These and other changes can be made in the source code version running in LabVIEW 8.6 or future versions.