PID Explained for Process Engineers: Part 1 — The Basic Control Equation

Cecil L. Smith, P.E.
Cecil L. Smith, Inc.

The differential equation for PID control contains three possible modes: proportional, integral, and derivative. This article describes the control equation in language that process engineers can readily understand.

Understanding the process is the key to successful control applications in the process industries. Since process engineers do this far better than anyone else, the logical conclusion is that developing and enhancing control configurations should be the responsibility of process engineers. The counter argument is that process engineers do not have a sufficient understanding of the principles of automatic control to undertake such a task.

Many process engineers lack this understanding because the traditional method of teaching automatic control is not inherently clear. Explanations that rely on the mysterious "s" variable in LaPlace transforms do more to obscure the basic principles than to elucidate them. This article focuses on the time domain *(1)*.

The objective of this three-part series of articles is to explain the proportional-integral-derivative (PID) control equation in language that process engineers, most being chemical engineers, can readily understand. Part 1 focuses on the basic PID equation. Next month, Part 2 examines the tuning coefficients, and in March, Part 3 explains the most common controller features and options.

Loop representation

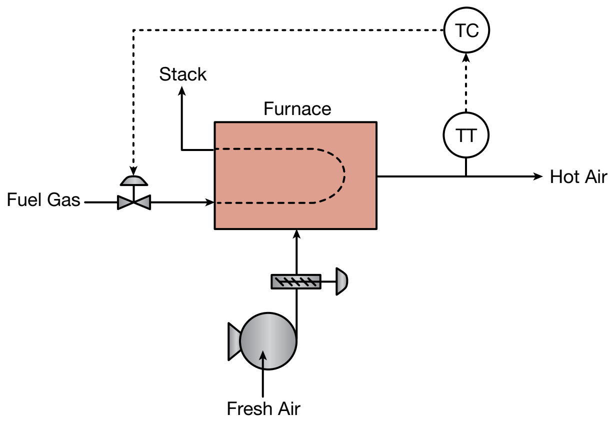
Figure 1 presents a piping and instrumentation (P&I) diagram for a simple temperature control loop, which consists of three components:

- measurement device — the hot air temperature transmitter
- PID controller
- final control element — the fuel control valve.

Additional components are sometimes included in the P&I diagram, but to avoid cluttering the figure, a separate control logic diagram presenting all components can be prepared.

Figure 2 is a block diagram representation of the PID loop in Figure 1. This block diagram contains three function blocks:

- *PV block* — converts the input from the measurement device to a value in engineering units (*i.e.,* °C for the hot air temperature). The result is usually referred to as a process variable (PV).
- *PID block* — provides the PID control calculations. The result of the control calculations is the output to the final



▲ **Figure 1.** In this piping and instrumentation (P&I) diagram for a simple temperature control loop, the hot air temperature is controlled by a valve on the fuel gas supply line.

control element in engineering units of % open.

• *valve block* — converts the % open input to whatever signal is required by the final control element.

In addition to the PV input, the PID block also requires an additional input — the desired value, or setpoint, for the PV input.

Control engineers often use the following terminology for the PV input and the controller output:

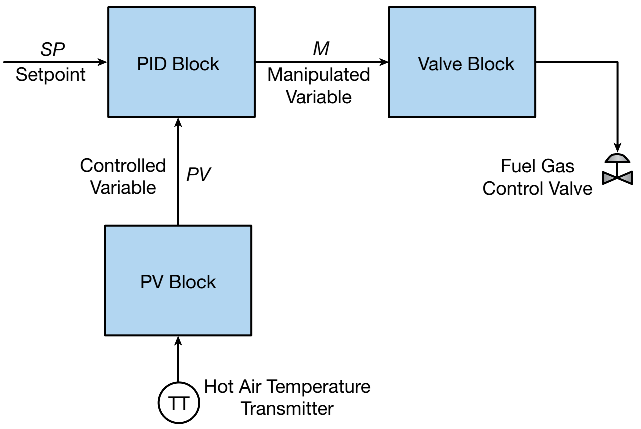
• *controlled variable* — the PV whose value is to be maintained at or near the setpoint

• *manipulated variable* — the variable whose value is at the discretion of the controller. For the loop in Figure 1, the manipulated variable is the fuel gas valve opening. Throughout this article, the controller output will be designated by *M*.

Process relationships such as material and energy balances are expressed in terms of flows, not valve or damper openings. For the loop in Figure 1, process-minded people sometimes refer to the fuel flow as the manipulated variable. Technically, such flows are dependent variables that are largely determined by the valve opening but are affected by upstream pressure, downstream pressure, and fluid properties.

Valve block

In Figure 1, the symbol for the fuel gas valve suggests a valve equipped with a pneumatic diaphragm actuator. Such actuators are common in process facilities, primarily because of their failure characteristics. On a loss of either the control signal or power (supply air), the actuator can be configured to drive the valve either fully closed or fully open. This greatly simplifies the hazards analysis — each actuator can be configured to whatever is most appropriate for that loop.



▲ **Figure 2.** In this block diagram representation of the PID loop in Figure 1, the setpoint is entered as input by the operator and the process variable is determined by the hot air temperature transmitter. The PID block provides the PID control calculations.

For 4–20-mA current loop installations, the actuator interprets the output signal from the controller as follows:

• *fail-closed.* The output signal determines the % open value for the final control element. For an output signal of 8 mA (25% of the 4–20-mA span), the final control element will be 25% open. Increasing the output signal increases the opening of the final control element.

• *fail-open*. The output signal determines the % closed value for the final control element. For an output signal of 8 mA (25% of the 4–20-mA span), the final control element will be 25% closed, or 75% open. Increasing the output signal decreases the opening of the final control element.

The configuration parameters for the valve block specify the fail-closed/fail-open behavior of the actuator. Based on that, the valve block performs the necessary conversion from the input value (always % open) to the output signal appropriate for the final control element.

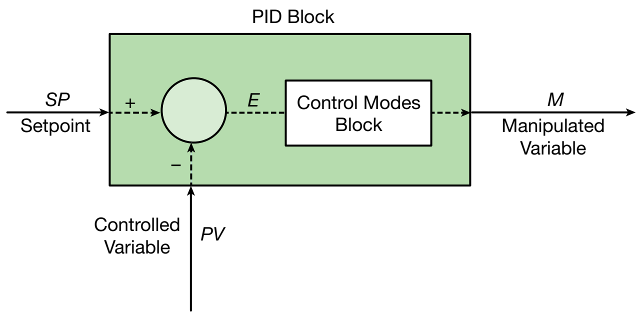
Process operators and control engineers prefer to think of a final control element in terms of % open. The use of valve blocks permits all control configurations to generate every output as % open. The valve block converts the output to % closed if necessary, that is, if the actuator is fail-open. Only the valve block needs to know the failure behavior of the final control element.

Incorporating the functions of the valve block into the PID block was very common in older models of commercial process control systems, and some continue to do so. However, this works well only for simple loops such as the one in Figure 1. When a high/low selector block (for override control) or other computation *(2)* is inserted between the PID block and the final control element, it is best to separate the valve block from the PID block.

Controller action

Figure 3 illustrates the PID block in more detail. It has two components:

• *comparator.* The control error (*E*) is the difference between the setpoint (*SP*) and the process variable (*PV*). Fig-



▲ **Figure 3.** Within the PID function block, a comparator computes the control error, *E*. The differential equation for PID control is implemented in the control modes block, whose output is the manipulated variable, *M*.

ure 3 suggests that the control error is computed as *SP* minus *PV*. But for some loops, the control error must be computed as *PV* minus *SP*. (More on this later.)

• *control modes.* The differential equation for PID control contains three possible modes: proportional, integral (also called reset), and derivative.

When drawing a block diagram for a PID loop, control engineers often label the control modes block in Figure 3 as the controller. Indeed, this block is an important one.

Before examining the individual modes, an explanation of controller action is essential.

Consider the control loop in Figure 1. The hot air temperature is controlled by manipulating the fuel gas valve opening. Suppose the hot air temperature is increasing. How should the controller respond? The controller should decrease the fuel gas valve opening. An increase in the controlled variable (hot air temperature) requires a decrease in the manipulated variable (fuel gas valve opening or fuel flow).

An alternative approach is to control the hot air temperature by manipulating the fresh air damper opening. This configuration is illustrated in Figure 4. Suppose the hot air temperature is increasing. How should the controller respond? The controller should increase the fresh air damper opening. An increase in the controlled variable (hot air temperature) requires an increase in the manipulated variable (fresh air damper opening or fresh air flow).

Controller action is defined as reverse or direct *(3)*:

• *reverse* (also called increase-decrease) — in response to an increase in the process variable, the controller decreases its output

• *direct* (also called forward or increase-increase) — in response to an increase in the process variable, the controller increases its output.

Specifically note that controller action is based on the process variable and not the control error. Why base controller action on the process variable? By defining controller action based on the process variable, the definition can also be applied to regulators (which do not contain a comparator, as the PID block in Figure 3 does).

When a valve block (or its equivalent) is used to isolate the controls from the fail-open/fail-closed nature of the final control element, the output of every PID controller is % open. The direction in which the controller must respond depends solely on the behavior of the process.

The configuration parameters for each PID block include a specification for the controller action. If the controller is reverse-acting (as in Figure 3), the control error, *E*, is computed by:

$$E = SP - PV \tag{1}$$

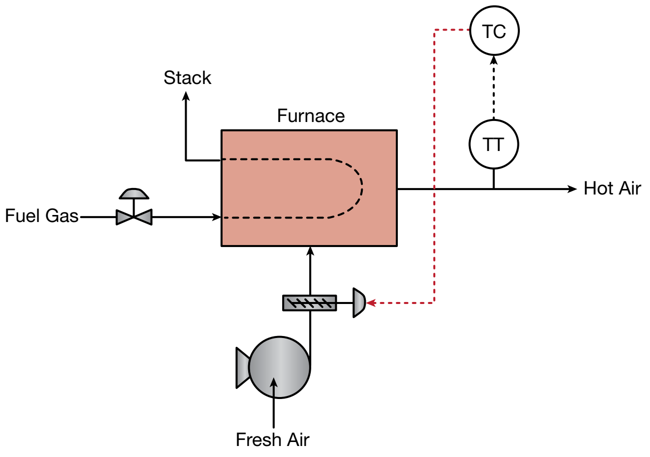
Increasing *PV* decreases *E*, which decreases the output of the controller.

If the controller is direct-acting, the inputs are switched, and the control error is computed by:

$$E = PV - SP \tag{2}$$

Increasing *PV* increases *E*, which increases the output of the controller.

This article uses this approach to incorporate direct or reverse action into the control equation. While most pneumatic and electronic controls used this approach, other possibilities are available with digital controls. From an application perspective, how direct or reverse action is implemented is immaterial.



▲ **Figure 4.** If the hot air temperature is controlled via the fresh air damper, an increase in the hot air temperature must result in an increase in the fresh air damper opening. In this scenario, the user must specify direct controller action.

Proportional-plus-bias

In process applications, almost all PID controllers rely at least to some degree on the proportional mode of control. By proportional, it is understood that the controller output, *M*, is proportional to the control error, *E*. This suggests the following equation for the proportional mode of control:

$$M = K_C E \tag{3}$$

where K_C is the controller gain (a tuning coefficient).

Consider the hot air temperature loop in Figure 1. Suppose the setpoint is 210°C and that the controller has acted so that the hot air temperature is also 210°C. Since $PV = SP$, the control error is zero. Substituting $E = 0$ into Eq. 3 suggests that $M = 0$, which means that the fuel gas valve is 0% open (fully closed). This makes no sense. The fresh air enters at the ambient temperature, so some fuel is required to heat the air to 210°C. The fuel valve cannot be fully closed.

Article continues on next page

Although the mode is referred to as proportional, the equation for the mode is best described as proportional-plus-bias:

$$M = K_C E + M_R \tag{4}$$

where M_R is the bias, specifically the controller output bias (the term bias is used in other contexts within process control). The control error E is computed, then multiplied by the controller gain K_C, and the result added to the controller output bias M_R to obtain the controller output M.

Initialization and PV tracking

Where do we get a value for the controller output bias M_R? The PID controller has two operational modes:

• *manual* — the controller output, M, is specified by the process operator, control calculations are not performed, and no value is required for M_R

• *automatic* — the controller output, M, is computed based on the PID control equation; these computations require a value for M_R.

When a controller is switched from manual to automatic, the control equation must be properly initialized so that there is no abrupt change in the controller output, M. This is known as a bumpless transition.

The time of the switch is designated as time zero, and M_0 is the controller output just prior to the switch. For the transition to be bumpless, the first value computed in automatic mode must be M_0. To achieve this, the proportional-plus-bias equation is rearranged so that the initial value, $M_{R,0}$, can be computed:

$$M_{R,0} = M_0 - K_C E_0 \tag{5}$$

where E_0 is the difference between the process variable and setpoint at the time the controller is switched to auto.

PV tracking is an option that affects the initialization of the controller (Table 1). With PV tracking enabled, the initial error (E_0) is zero, which simplifies Eq. 5. The simpler computation, with PV tracking enabled, led to its use within pneumatic and electronic analog controls. For digital controls, the additional calculations for PV tracking disabled are trivial, so most allow PV tracking to be enabled or disabled on an individual loop basis.

When a controller is switched to auto, the operator is responsible for providing a proper value for the setpoint. How this is done depends on whether PV tracking is enabled or disabled:

• *PV tracking enabled.* The operator cannot change the setpoint while the controller is in manual. The operator must first switch the controller to auto, and then enter the value for the setpoint.

• *PV tracking disabled.* The value of the setpoint is used to compute the initial control error, E_0, which is then used to calculate $M_{R,0}$. The operator must first enter the value for the setpoint, and then switch the controller to auto.

Some advanced control configurations require PV tracking to be enabled or disabled. But for most loops, the decision to enable or disable is largely personal preference. The best advice is to decide which to use, and make all loops the same except where advanced control (or another specific reason) dictates otherwise.

Offset or droop

In controllers with only proportional action, the value of the controller output bias, M_R, remains at its initial value, $M_{R,0}$. The consequence of a fixed value for M_R is an annoying phenomenon referred to as offset, but the term droop, commonly used in process facilities, is a more accurate description.

Let's look at how droop arises in the loop in Figure 1. As Table 2 indicates, the loop is initially at an equilibrium state (lined out in control jargon) at its setpoint of 210°C. The controller output (M) is 82.5% open, and since $E = 0$, the controller output bias must also be 82.5% open.

Figure 5 shows the behavior of the hot air temperature

Table 1. PV tracking affects the initialization of a controller. When PV tracking is enabled, the initial control error, E_0, is 0, because the setpoint is changed to the value of the process variable.

	PV Tracking Enabled	PV Tracking Disabled
Actions in Manual	Setpoint is changed to the value of the process variable	Setpoint is not changed
Value of Control Error on Switch from Manual to Auto	$E_0 = 0$	Direct-acting: $E_0 = PV_0 - SP_0$ Reverse-acting: $E_0 = SP_0 - PV_0$
Initial Value for Controller Output Bias	$M_{R,0} = M_0$	$M_{R,0} = M_0 - K_C E_0$
Operator May Change Setpoint while in Manual	No	Yes
Switching from Manual to Auto	1. Switch to auto 2. Enter value for setpoint	1. Enter value for setpoint 2. Switch to auto

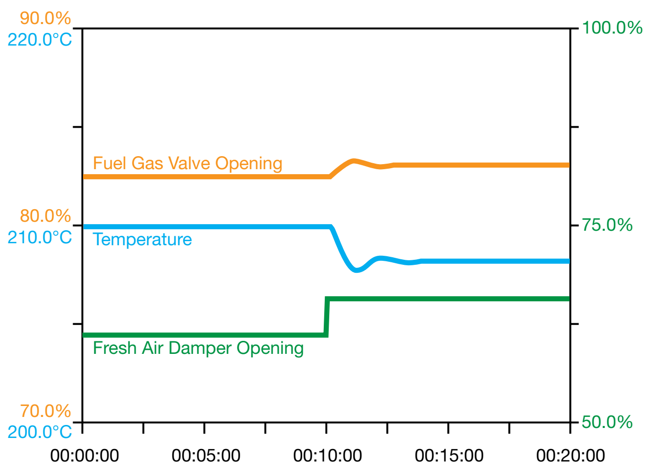
Table 2. After a 5% increase in the fresh air damper opening, a loop with proportional-only control exhibits droop. With a fixed value for the controller output bias, M_R, the process variable will not line out at its setpoint.

	Initial	Final
Setpoint	210.0°C	210.0°C
Fresh Air Damper Opening	60.8%	65.8%
Hot Air Temperature, *PV*	210.0°C	207.8°C
Control Error, $E = SP - PV$	0.0°C	2.2°C
Controller Output Bias, M_R	82.5% open	82.5% open
PV Span	300.0°C	300.0°C
Controller Gain, K_C	1.0 %/% = 0.333%/°C	1.0 %/% = 0.333%/°C
Proportional Term, K_CE	0.00% open	0.73% open
Controller Output, *M*	82.5% open	83.2% open

controller for a 5% increase in the fresh air damper opening (green line). An increase in the fresh air flowrate causes the hot air temperature (blue line) to droop down (from 210°C to 207.8°C, an error of 2.2°C); a decrease causes the hot air temperature to ride up.

Following the increase in the fresh air flowrate, the hot air temperature controller opens the fuel gas valve from 82.5% to 83.2%, an increase of 0.7%. As Table 2 shows, this increase is entirely due to the K_CE term in the proportional-plus-bias equation (Eq. 4).

At the higher fresh air flow, a controller output of 83.5% is required to attain a hot air temperature of 210°C. But should a temperature of 210°C be attained, the control error would be zero and the controller output would be 82.5% (the value of the controller output bias, M_R).



▲ **Figure 5.** A proportional-only controller exhibits droop after a 5% increase in the fresh air damper opening. Initially, the controller output (fuel gas valve opening) is 82.5% open. When the fresh air damper opening increases (green line), the temperature drops from its setpoint at 210°C to 207.8°C (blue line), and the fuel gas valve opening increases to 83.2% (orange line). For the controller to return the temperature to its setpoint, the controller output bias, M_R, must be adjusted.

A value of 210°C for the hot air temperature can be obtained by adjusting the controller output bias. Most commercial process controls do not provide features to readily display the value of the controller output bias nor accept changes to its value. This was not always the case. The earliest controllers were proportional-only, but provided an adjustment (often labeled manual reset) for the controller output bias, M_R, that permitted the process operator to remove any droop or offset.

The addition of integral action to the controller has largely eliminated the need for manual reset. Process operators are no longer familiar with manual reset, nor do most commercial control systems provide such a feature. The integral mode automatically adjusts M_R so that the controlled variable lines out at its setpoint, thus eliminating droop. The term reset has stuck, though, and the integral mode is commonly referred to as the reset mode.

Integral mode

To eliminate the droop exhibited by the hot air temperature response (Figure 5), the controller output bias, M_R, must be increased. This can be achieved by adjusting M_R at a rate proportional to the control error, *E*:

$$\frac{dM_R}{dt} = K_I E \qquad (6)$$

where K_I is the integral gain (a tuning coefficient). For positive values of *E*, the controller output bias M_R increases; for negative values, M_R decreases.

Integrating gives the more familiar equation for the integral mode:

$$M_R = \int K_I E \, dt \qquad (7)$$

To achieve a smooth transition from manual to automatic, the initial value for M_R in both equations must be the value computed for $M_{R,0}$. In practice, the value computed for $M_{R,0}$ becomes the initial condition for the integrator.

If the integral mode is present in the control equation, the controller can only line out with the process variable equal to the setpoint. If the process variable is not equal to the setpoint, the following statements will be true:

• control error, E, is nonzero

• controller output bias, M_R, will be changing, as will the controller output, M

• the loop is not lined out.

Addition of the integral mode to the control equation does not assure that a loop will line out, but if it does line out, the process variable will be equal to the setpoint. A controller with integral action will not experience the droop depicted in Figure 5.

Proportional-plus-integral (PI)

An advantage of the integral mode is that the controller can line out only at its setpoint. But if the integral mode is used alone, the controller responds very slowly. Proportional responds more rapidly, but when used alone, the loop exhibits droop. Using the two modes together creates a loop that responds more rapidly and can line out only at its setpoint.

The equations for the proportional and integral modes can be written as follows:

$$M = K_C E + M_R \tag{4}$$

$$M_R = \int K_I E \, dt \tag{7}$$

These equations are usually combined into a single equation:

$$M = K_C E + \int K_I E \, dt \tag{8}$$

The integral mode can also be expressed using the integral time or reset time, T_I, instead of the integral gain, K_I:

$$M = K_C E + \int \frac{K_C}{T_I} E \, dt \tag{9}$$

Since the controller gain, K_C, appears in both modes, the equation is often written as:

$$M = K_C \left(E + \frac{1}{T_I} \int E \, dt \right) \tag{10}$$

An alternative form for the PI control equation is the reset feedback form, which will be covered in Part 3 of this series.

Derivative mode

An older term for the derivative mode was "pre-act," which suggests that the controller is anticipating the process and acting in advance. Indeed, this is what derivative control attempts to do.

For the PV input to the controller, two items of information can be computed:

• current value of PV

• rate of change of PV, that is, dPV/dt.

Assuming that PV continues to change at the current rate for some time into the future allows a simple predictor to be formulated.

At time T_D in the future, a projected value, $\hat{I}$, can be computed for PV by:

$$\hat{I} = PV + T_D \frac{dPV}{dt} \tag{11}$$

This is illustrated in Figure 6. PV is below the setpoint but moving toward it. The projected value $\hat{I}$ is closer to the setpoint than the current value of PV. Instead of basing the control calculations on the control error, E, computed from PV, the control calculations could be based on the projected control error, $\hat{E}$, computed from $\hat{I}$.

The derivative time, T_D, is a tuning coefficient. Its value suggests how far into the future the PV can be projected using its current rate of change. This depends on the nature of the process. One of the most common applications of the derivative mode is for temperature processes, which are slow to respond.

A sequence to perform the control calculations for a proportional-plus-derivative (PD) controller is as follows:

1. Compute a value for $\hat{I}$.

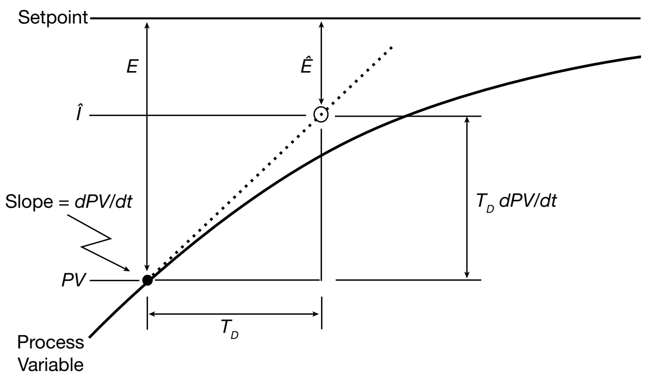
2. Compute the projected control error, $\hat{E}$, using either Eq. 12 or Eq. 13.

For a direct-acting controller:

$$\hat{E} = \hat{I} - SP \tag{12}$$

For a reverse-acting controller:

$$\hat{E} = SP - \hat{I} \tag{13}$$



▲ **Figure 6.** Based on the rate of change of PV, a projected value, $\hat{I}$, for the process variable can be computed. The control calculations are then based on the projected error, $\hat{E}$, computed from $\hat{I}$.

3. Use $\hat{E}$ instead of E in the proportional-plus-bias equation:

$$M = K_C \hat{E} + M_R \tag{14}$$

To obtain the customary expression for a PD controller, the above equations are combined:

$$\begin{aligned} M &= K_C \hat{E} + M_R \\ &= K_C (SP - \hat{I}) + M_R \\ &= K_C \left(SP - PV - T_D \frac{dPV}{dt} \right) + M_R \\ &= K_C \left(E - T_D \frac{dPV}{dt} \right) + M_R \end{aligned} \tag{15}$$

Equation 15 applies to a reverse-acting controller. For a direct-acting controller, the sign on the derivative term would be positive.

Proportional-integral-derivative

Starting with the PI control equation, a similar approach gives the equation for a PID controller:

1. Compute the projected control error, $\hat{E}$.

2. In the proportional term of the PI control equation, replace the control error, E, with the projected control error, $\hat{E}$.

For a reverse-acting controller:

$$\hat{E} = SP - \hat{I} = E - T_D \frac{dPV}{dt} \tag{16}$$

$$M = K_C \left(\hat{E} + \frac{1}{T_I} \int E\, dt \right) \tag{17}$$

Combining Eqs. 16 and 17 gives a single equation for PID control:

$$\begin{aligned} M &= K_C \left(E - T_D \frac{dPV}{dt} + \frac{1}{T_I} \int E\, dt \right) \\ &= K_C \left(E + \frac{1}{T_I} \int E\, dt - T_D \frac{dPV}{dt} \right) \end{aligned} \tag{18}$$

This equation applies to a reverse-acting controller. For a direct-acting controller, the sign on the derivative term would be positive.

Derivative based on *E* or *PV*

The approach described in the previous section is based on the rate of change of the process variable and yields a PID equation with its derivative in terms of *PV*. In practice, basing derivative on *PV* is generally preferred, and some commercial control systems provide only this option. The PID equation presented in most textbooks contains a derivative term based on the control error, *E*, and some commercial control systems also provide this option. Whether derivative is based on the process variable or on the control error has a minor effect on the performance of a loop.

To project the control error, *E*, one derivative time into the future and obtain the projected control error, $\hat{E}$, use the equation:

$$\hat{E} = E + T_D \frac{dE}{dt} \tag{19}$$

Replacing the control error in the proportional term of the PI control equation (Eq. 10) with the projected control error calculated by Eq. 19 gives a PID equation with derivative based on *E*:

$$M = K_C \left(E + \frac{1}{T_I} \int E\, dt + T_D \frac{dE}{dt} \right) \tag{20}$$

An issue arises when derivative is based on the control error. Say the operator makes an abrupt change in the setpoint, such as raising a temperature setpoint from 210°C to 215°C. An abrupt setpoint change produces an abrupt change in the control error, *E*. Mathematically, the derivative of an abrupt change is the impulse function. In process control systems, the result is a rate of change with a large magnitude but a short duration — that is, a spike. Part 3 of this series examines the consequences of a spike as part of the discussion on computing rates of change.

PID computations

Only the continuous forms of the control equations have been presented so far. Digital controls must use difference equations that are numerical approximations to the continuous equations.

With today's computers, the time between executions of the PID equation, known as the sampling time, Δt, is one second or less. Most processes are slow, and such sampling times do not introduce significant errors into the calculations. Technically, one should write difference equations, but with such good approximations, the continuous equation is common. Some people find the difference equations more informative, so the computational procedure outlined here uses those.

An iteration consists of the computations performed on each sampling instant. Each iteration is indicated by a subscript, with n being the current iteration, n–1 being the previous iteration, and 0 being the first iteration following a switch from manual to auto. On iteration n, M_n is the controller output, PV_n is the process variable, and $M_{R,n}$ is the controller output bias. On the first iteration, M_0 is the controller output, PV_0 is the process variable, and $M_{R,0}$ is the controller output bias computed for bumpless tansfer.

The following computational procedure mirrors how

PID control actually functions:

1. Start with values for PV_n and PV_{n-1}.
2. Compute the value of PV'_n, the rate of change of PV on iteration n:

$$\frac{dPV}{dt} \cong \frac{PV_n - PV_{n-1}}{\Delta t} = PV'_n \quad (21)$$

3. Compute the projected value, $\hat{I}_n$, for PV on iteration n:

$$\hat{I}_n = PV_n + T_D PV'_n \quad (22)$$

4. Compute the control error, E_n, and the projected control error, $\hat{E}_n$.

For a direct-acting controller:

$$E_n = PV_n - SP_n \quad (23)$$

$$\hat{E}_n = \hat{I}_n - SP_n \quad (24)$$

For a reverse-acting controller:

$$E_n = SP_n - PV_n \quad (25)$$

$$\hat{E}_n = SP_n - \hat{I}_n \quad (26)$$

5. Apply integral action to obtain a new value for the controller output bias, $M_{R,n}$:

$$M_{R,n} = M_{R,n-1} + \frac{K_C E_n \Delta t}{T_I} \quad (27)$$

The initial condition is the value $M_{R,0}$ computed for bumpless transfer.

6. Apply the proportional-plus-bias equation based on $\hat{E}$ to obtain the controller output, M_n:

$$M_n = K_C \hat{E}_n + M_{R,n} \quad (28)$$

These relationships provide the basic PID control equation with derivative based on PV.

Closing thoughts

This article has presented the basic PID control equation. However, stopping here would be premature.

This article uses K_C, T_I, and T_D as the tuning coefficients (also called tuning parameters). All control systems provide a tuning coefficient for each mode, but not all use K_C, T_I, and T_D. The next article in this three-part series discusses alternative tuning coefficients for each mode. In practice, the most undervalued is the coefficient for the proportional mode. This is unfortunate; the speed of response of a loop is primarily a function of the proportional mode tuning coefficient. Part 2 examines such issues.

The flexibility of digital technology permits additional features and options to be incorporated into the PID block. The third article explains the most common of these, including windup and windup protection. CEP

Nomenclature

E	= control error, difference between the process variable and setpoint
E_0	= control error on iteration 0
E_n	= control error on iteration n
$\hat{E}$	= projected control error
$\hat{E}_n$	= projected control error on iteration n
$\hat{I}$	= projected value of PV
$\hat{I}_n$	= projected value of the PV on iteration n
K_C	= controller gain
K_I	= integral gain
M	= controller output, manipulated variable
M_n	= controller output on iteration n
M_0	= controller output on iteration 0
M_R	= controller output bias
$M_{R,0}$	= controller output bias on iteration 0
$M_{R,n}$	= controller output bias on iteration n
n	= current iteration (iteration 0 occurs when the controller is switched to auto)
PV	= process variable, *i.e.,* controlled variable
PV_0	= process variable on iteration 0
PV_n	= process variable on iteration n
PV'_n	= rate of change of the PV on iteration n
SP	= setpoint
SP_0	= setpoint on iteration 0
SP_n	= setpoint on iteration n
t	= time
T_D	= derivative time
T_I	= integral time or reset time

Greek Letters

Δt	= sampling time (time between iterations)

Literature Cited

1. **Smith, C. L.,** "Practical Process Control: Tuning and Troubleshooting," John Wiley and Sons, Hoboken, NJ (2009).
2. **Smith, C. L.,** "Advanced Process Control: Beyond Single Loop Control," John Wiley and Sons, Hoboken, NJ (2010).
3. **Instrument Society of America,** "Process Instrumentation Terminology," ISA-51.1-1979 (R1993).

CECIL L. SMITH, PhD, P.E., is president of Cecil L. Smith, Inc. (2306 Avalon Place, Houston, TX 77019; Phone: (225) 268-4391; Email: cecilsmith@cox.net). He has gained expertise in virtually every control technology used in industrial production facilities during his 50 years of experience in process control. He has presented a wide range of continuing education courses on various aspects of process control, with a primary focus on developing and enhancing control configurations. He is featured in the International Society of Automation's (ISA's) Leaders of the Pack (2003) and *Control's* Process Automation Hall of Fame (2009). He has authored five books on process control. Smith received his BS, MS, and PhD, all in chemical engineering, from Louisiana State Univ. (LSU). He is a registered P.E. in California.