FEATURE SERIES: PRACTICAL PROCESS CONTROL

3: PID algorithm versions

***Myke King** continues his detailed series on process control, seeking to inspire chemical engineers to exploit untapped opportunities for improvement*

THE proportional-integral-derivative (PID) control algorithm has been the regulatory controller of choice for around 85 years. Initially provided by pneumatic instrumentation, and later by electronic analog, the advent of digital control systems facilitated a range of modifications. Industry has yet to fully exploit many of the valuable features now available. To understand these missed opportunities we need to develop the algorithm from first principles.

PROPORTIONAL

The proportional action is defined by

$$M = K_c E + C$$

M is the controller output; using our example of the fired heater in last month's article, it is the set-point of the fuel flow controller. E is the error – the deviation of the process variable (PV) from the set-point (SP). While not all control system vendors have adopted it, the now recognised definition is $PV - SP$. This is part of the standard published by the ISA (formerly the Instrument Society of America, now the International Society of Automation). Traditionally, text books will use $SP - PV$; and accounts for differences in sign when formulae from different sources are compared.

The algorithm includes the first of our three tuning parameters – the controller gain (K_c). The term C is required because it is unlikely that the error will be zero when the fuel flow is zero. It represents the flow of fuel required when the temperature is at set-point. Unfortunately C is not a constant, varying with feed rate and many other factors.

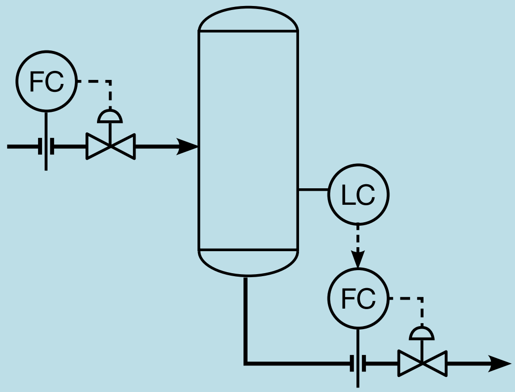
As presented, the algorithm is in the *full position* form. To eliminate C most control systems use the velocity form. We obtain this by differentiation.

$$\frac{dM}{dt} = K_c \frac{dE}{dt}$$

Modern control systems are digital, operating at a fixed *scan interval* (ts). So we make an approximation to give the *incremental* form.

$$\frac{\Delta M}{ts} = K_c \frac{\Delta E}{ts} \quad \text{or} \quad \Delta M = K_c (E_n - E_{n-1})$$

Figure 1: Vessel level controller



The main purpose of proportional control is to respond to changes in set-point. In the first control interval, after a change, ΔE will be equal to ΔSP. The controller will generate a step change in output (known as the *proportional kick*) equal to $K_c \Delta SP$.

However, the PV will not reach the set-point. Instead there will be a sustained error, known as *offset*. To understand why this occurs, consider the level controller illustrated as Figure 1. Assume that the process is at steady state and that the controller error is zero. If the inlet flow is then increased by f then, to achieve steady state, the controller must increase the outlet flow also by f.

$$\Delta M = f = K_c (E - 0) \quad \text{or} \quad E = \frac{f}{K_c}$$

INTEGRAL

No matter how large we make K_c, we cannot reduce E to zero. To resolve this, we add integral action. The principle is to change M at a rate proportional to the error. In other words, M will only stop changing once the error is zero. Extending the P only controller to include integral action gives the PI controller. The amount of integral action is determined by our second tuning parameter – the *integral time* (Ti)

$$\frac{dM}{dt} = K_c\left[\frac{dE}{dt} + \frac{E}{T_i}\right] \quad \text{or} \quad \Delta M = K_c\left[(E_n - E_{n-1}) + \frac{ts}{T_i}E_n\right]$$

Integral action gets its name from the full position version of the algorithm. Integrating gives

$$M = K_c\left[E + \frac{1}{T_i}\int E.dt\right] + C$$

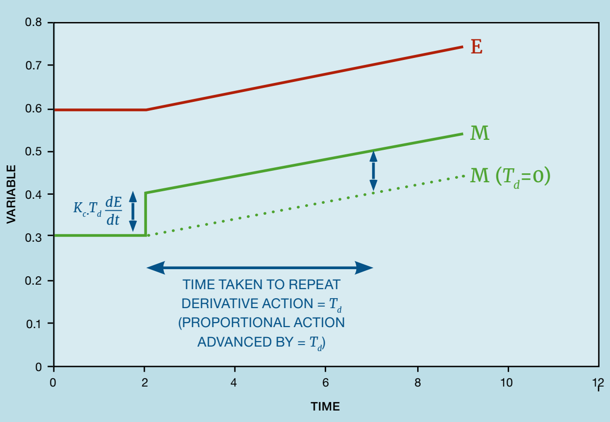
DERIVATIVE

For many applications the PI algorithm will give adequate control. However, when we later come to tune the controller, we'll show that the addition of derivative action permits K_c to be significantly increased and so resolve process disturbances more quickly. Derivative action is based on the rate of change of error. Adding it to the full position PI algorithm

$$M = K_c\left[E + \frac{1}{T_i}\int E.dt + T_d\frac{dE}{dt}\right] + C$$

This adds our third tuning parameter – *derivative time* (T_d). If the error is zero, no action will be taken by the proportional or integral actions. But, if it is changing quickly, an error will surely exist in the future. Derivative action anticipates this; indeed, it was once called *anticipatory control*. Figure 2 illustrates this. At the point where the error begins to move there is a change in the rate-of-change of error. The derivative action responds to this, making a step change in the controller output. This is then followed by the proportional action, keeping M in proportion to E. Without the derivative action the controller would respond as shown by the dashed line – eventually making the same change as the derivative action but delayed by the time T_d.

Figure 2: Advantages of derivative action



Differentiating and converting to digital control

$$\frac{dM}{dt} = K_c\left[\frac{dE}{dt} + \frac{E}{T_i} + T_d\frac{d^2E}{dt^2}\right]$$

$$\frac{\Delta M}{ts} = K_c\left[\frac{\Delta E}{ts} + \frac{E}{T_i} + T_d\frac{\Delta(\Delta E)}{ts^2}\right]$$

$$\Delta M = K_c\left[(E_n - E_{n-1}) + \frac{ts}{T_i}E_n + \frac{T_d}{ts}(E_n - 2E_{n-1} + E_{n-2})\right]$$

This is generally known as the *ideal* version of the algorithm. An alternative version adds derivative action to the PI controller by replacing E with the *projected error* (E') – again anticipating the need to take corrective action.

$$E' = E + T_d\frac{dE}{dt}$$

This results in the algorithm

$$\Delta M = K_c\left[\left(1 + \frac{T_d}{T_i}\right)(E_n - E_{n-1}) + \frac{ts}{T_i}E_n + \frac{T_d}{ts}(E_n - 2E_{n-1} + E_{n-2})\right]$$

If derivative action is included ($T_d > 0$) then changing either T_i or T_d will now also affect the amount of proportional action. For this reason, the algorithm is described as *interactive*.

In principle, both the ideal and interactive algorithms are implemented in the control system as derived. However, the digital approximation causes a problem with the derivative action. Imagine that the process has been steady for some time and the operator causes a controller error by changing the set-point. Ignoring, for the moment, the action taken by the proportional and integral parts of the algorithm, the changes made by the derivative action are shown in Table 1. It causes a *derivative spike* that has a duration of one controller scan interval and a magnitude of $K_c T_d E/ts$. The value of T_d will be around a minute or more, while ts will be few seconds. T_d/ts will therefore be of the order of 60, so even a small change in set-point can cause M to change by more than 100%. Potentially the fuel valve could be fully open

Table 1: Derivative spike

TIME	E_n	E_{n-1}	E_{n-2}	$\Delta M_{derivative}$
$-ts$	0	0	0	0
0	E	0	0	$K_c T_d E/ts$
ts	E	E	0	$-K_c T_d E/ts$
$2ts$	E	E	E	0
.	E	E	E	0
.	E	E	E	0
$\theta - ts$	E	E	E	0

(or fully closed) for one scan interval. This may cause a serious problem in its own right, potentially causing a plant trip. Once the process deadtime has elapsed, the temperature will show a similar spike. This will be detected as an error and the controller will take corrective action causing the spike to repeat.

To resolve this problem, we replace the error (*E*) in the derivative action, with the process value (*PV*). If *SP* is constant, changes in *PV* are the same as changes in *E*. The response of controller to changes in *PV* (known as *load changes*) will be unaffected but now only the proportional and integral actions will respond to changes in *SP*.

$$\Delta M = K_c\left[E_n - E_{n-1} + \frac{ts}{T_i}E_n + \frac{T_d}{ts}(PV_n - 2PV_{n-1} + PV_{n-2})\right]$$

It has become to be known as the PI-D algorithm (actions before the hyphen are based on *E*, those after on *PV*). While it resolves the problem caused by step changes in the *SP* , it is still vulnerable to such changes in the *PV*. For example, discontinuous on-stream analysers, such as chromatographs employ *sample-and-hold* – transmitting the last measurement until a new value is obtained. The steps in the resulting staircase output signal will cause derivative spikes. Another source of steps can be the measurement analog-to-digital conversion. For example, if its resolution is 0.1% of instrument range, then any change in measurement will be a multiple of 0.1%. Even if as low as 0.1%, the resulting derivative spike could readily by as large as 10%. Such issues preclude the use of derivative action.

I-PD ALGORITHM

Another important modification also bases the proportional action on *PV* to give the I-PD algorithm.

$$\Delta M = K_c\left[PV_n - PV_{n-1} + \frac{ts}{T_i}E_n + \frac{T_d}{ts}(PV_n - 2PV_{n-1} + PV_{n-2})\right]$$

Rather than include a library of control algorithms, many control systems adopt a single algorithm that includes additional parameters. These are then set by the engineer to select the required algorithm. An example of this is the *two degrees of freedom algorithm*, defined as

$$\Delta M = K_c\left[x_n - x_{n-1} + \frac{ts}{T_i}E_n + \frac{T_d}{ts}(y_n - 2y_{n-1} + y_{n-2})\right]$$

$$x = PV - \alpha SP \qquad y = PV - \beta SP$$

So, for example, setting both α and β to zero will give the I-PD algorithm. Setting them both to 1 gives the PID algorithm. The system may also support intermediate values of α and β that will give a blend of algorithms.

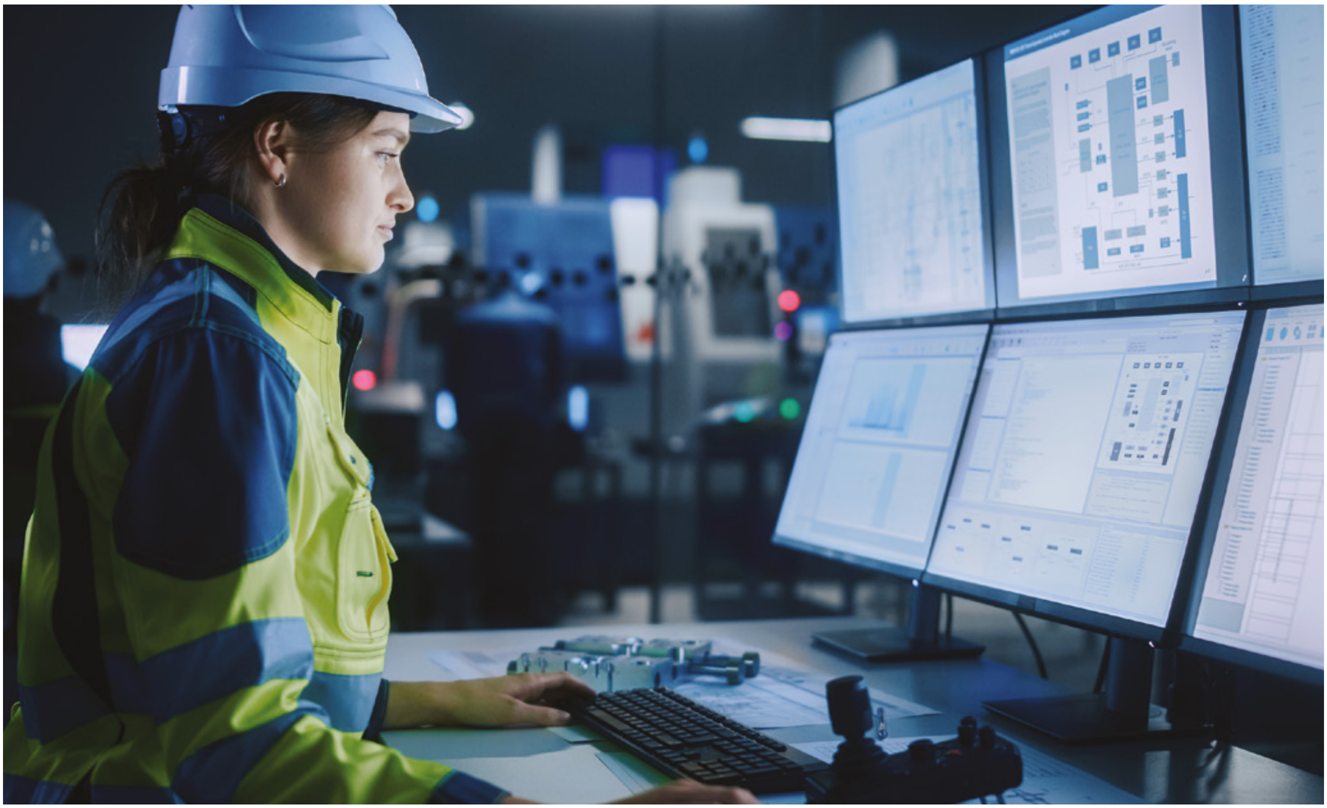


Table 2: DCS Configuration

CONTROL SYSTEM	IDEAL	INTERACTIVE	PID	PI-D	I-PD
ABB 800xA	not optional	not available	not available	BETA=1	BETA=0
Emerson Delta V	STANDARD	SERIES	STRUCTURE=Two Degrees of Freedom; BETA=1; GAMMA=1	STRUCTURE=Two Degrees of Freedom; BETA=1; GAMMA=0	STRUCTURE=Two Degrees of Freedom; BETA=0; GAMMA=0
Foxboro I/A	MODOPT=6	MODOPT=5	not available	SPPLAG=1	SPPLAG=0
Honeywell Experion	IDEAL	INTERACTIVE	EQNA	EQNB	EQNC
Yokogawa Centum	not optional	not available	Basic Type (PID)	PV Derivative Type PID Control (PI-D)	Proportional PV Derivative Type PID Control (I-PD)

Table 2 shows, for the leading DCS vendors, what algorithms are available and how they are selected.

ENGINEERING UNITS

Tuning parameters are expressed in different units in different control systems. For example, ABB, Emerson and Honeywell use controller gain (K_c) as we have defined it – as a dimensionless value. However, Foxboro and Yokogawa instead use *proportional band* (*PB*). This is defined as the percentage change in error (*E*) required to move the controller output (*M*) 100%. Conversion from one parameter to the other is simple.

$$PB = \frac{100}{K_c}\%$$

ABB, Foxboro and Honeywell express the integral time (T_i) and derivative time (T_d) in minutes. Emerson and Yokogawa use seconds.

We saw in the previous article that the process gain (K_p) can be either positive or negative. Controller gain (K_c) is, however, always configured as a positive number. We must also define, as part of its configuration, the controller action as either *direct* or *reverse*. A direct acting controller will increase the controller output (*M*) if the measurement (*PV*) increases. In the case of our fired heater example, we therefore need to specify a reverse-acting controller. In general, perhaps rather confusingly, if the process gain is positive we need a reverse-acting controller – selecting direct action if the process gain is negative.

An important feature of any control algorithm is that, when switched from manual to automatic, it does not cause a process disturbance. This is known as *bumpless transfer*

PV TRACKING

An important feature of any control algorithm is that, when switched from manual to automatic, it does not cause a process disturbance. This is known as *bumpless transfer*. In the case of the PID algorithm, this is achieved by *PV tracking*. When the controller is in manual, the set-point tracks the process variable so keeping the error zero. When switched to automatic, tracking is stopped and the set-point is made available for the process operator to change.

The choice of the incremental algorithm makes this *initialisation* straightforward. It also permits tuning constants to be changed, with the controller in automatic, without bumping the process. ■

NEXT ISSUE

Which version of the algorithm should be used (and how it should be tuned) will be covered in the next article.

Myke King CEng FIChemE is director of Whitehouse Consulting, an independent advisor covering all aspects of process control. The topics featured in this series are covered in greater detail in his book Process Control – A Practical Approach, published by Wiley in 2016.