PID math demystified, part 1

You’ve seen the equations, but have you thought about how those elements work together? Part 1: The basic concepts and proportional control.

Scott Hayes
05/06/2013

Most process control engineers have been exposed to the basic equation in a form that looks something like this:

$$u(t) = K_p e(t) + K_i \int_0^t e(t)\,dt + K_d \frac{de(t)}{dt}$$

More than you want to swallow in one bite? Let’s break this down into the major components:

$$u(t) \quad = \quad K_p e(t) \quad + \quad K_i \int_0^t e(t)\,dt \quad + \quad K_d \frac{de(t)}{dt}$$

Output = proportional + integral + derivative

Output: *u(t)* is the output of the controller at the end of the scan. If the output of the controller is a valve, then the output is the valve position that the controller is requesting after it has seen the inputs. In most controllers, this is actually the change in output from 50%. So if *u(t)* = 0 then the valve output is 50%; if *u(t)* = 1 then the valve output is 51%; and if *u(t)* = -2 then the valve output is 48%. You get the idea. But what’s important is that it’s not a change in output from the previous scan, but a new output.

Proportional

$K_p e(t)$ is the proportional component, the P in PID. If you have a controller configured as proportional only, this is it. So let's look at how this works.

Let's start with my own misconception of how I thought it worked. When I imagine a controller, I picture myself turning a valve while watching a gage. I look at the gage, decide if I need more or less, turn that valve a little more or a little less, and then repeat the process until the gage shows the value I want. That sounds fundamentally logical, but it is not how a proportional only controller works. It's more like if I were to look at the gage, subtract what it reads from what I want it to read, and then take that error over to a chart to look up a new value for the valve.

$e(t)$ usually called error, is simply the difference between the setpoint and the process variable. It is the difference between where you are and where you want to be, right now, at this instant.

K_p gain, is a factor that is multiplied by the error to give you the new output, the new valve position. It's that simple. The error at that instant of the scan is calculated and the new output is calculated.

Let's look at an example of pseudo code to explore how this works:

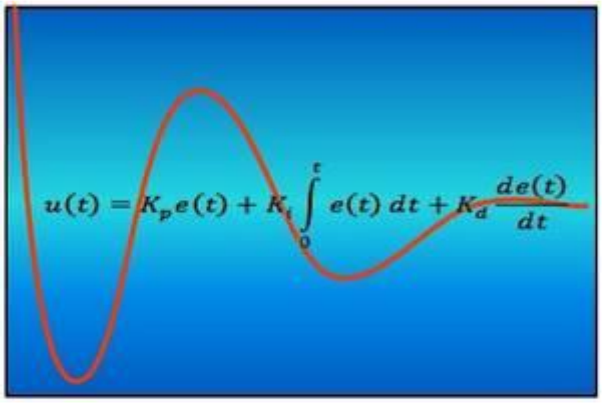
```
Error = Setpoint - ProcessValue;

Output = K * Error;
```

This control algorithm is deceptively simple, yet it gives an immediate response to a setpoint change or a disturbance in the process. And if *K* is set correctly, will quickly move the process toward the setpoint. But, it won't get the process to the setpoint because there has to be some error if the output is anything other than 50%.

Note: On some systems (though not often in modern systems), gain is expressed as proportional band. Proportional band is defined as the amount of change in the controlled variable required to drive the loop output from 0 to 100%. To convert between the two, gain = 100/PB.

Next week: Understanding I and D.


$u(t) = K_p e(t) + K_i \int_0^t e(t)\,dt + K_d \frac{de(t)}{dt}$

This post was written by Scott Hayes. Scott is a senior engineer at MAVERICK Technologies, a leading system integrator providing industrial automation, operational support, and control systems engineering services in the manufacturing and process industries. MAVERICK delivers expertise and consulting in a wide variety of areas including industrial automation controls, distributed control systems,manufacturing execution systems, operational strategy, and business process optimization. The company provides a full range of automation and controls services – ranging from PID controller tuning and HMI programming to serving as a main automation contractor. Additionally MAVERICK offers industrial and technical staffing services, placing on-site automation, instrumentation and controls engineers.

Source:

http://www.controleng.com/search/search-single-display/pid-math-demystified-part-1/df787b7bb0e5119d4c98743e1dc25f74.html