Variations in the PID Controller Algorithm

R. Russell Rhinehart

Develop Your Potential Series for CONTROL Magazine
Vol. 34, No. 7, July, 2021, pp. 39-41.

It is important to understand the variations on the PID algorithm when tuning and when choosing a version that is consistent within your use context. Unfortunately, there are many names for the several key versions of the PID algorithm, and names such as series or parallel are each applied to two distinctly different PID formulations. Accordingly, it has been best to identify the algorithm formulation by its calculus or Laplace representation, not the many names, to make clear distinctions.

To help unify the nomenclature, ISA has a Standards Committee 5.9 that is suggesting naming conventions for the PID algorithms. This article follows what is currently on the agenda to be proposed, but also reveals some of the commonly applied alternate names that you may be using.

The Main Three PID Algorithms

Standard: A Laplace representation of the standard algorithm is $\hat{u} = K_c\left(1 + \frac{1}{\tau_i s} + \tau_d s\right)\hat{e}$. This has been called parallel because the block diagram has the P, I, and D operations in parallel. But, also it is called series, because in this mathematical form the controller gain multiplies each of the P, I, and D terms in series. It is called ideal, because 1) it is the form that arises from controller synthesis, and 2) is most convenient for mathematical analysis. It has been called the ISA standard, even though ISA has not declared any algorithm as 'The Standard'. It is called non-interacting, or non-interactive, perhaps because it is lacking that "feature" of the interacting controller. The ISA 5.9 committee is moving toward recommending this form be called 'standard', but not declaring this as the best or right way to formulate the P, I, and D functions.

There are many versions of how this standard equation can be represented. If the terms inside the parentheses are combined, then it will appear as $\hat{u} = K_c\left(\frac{1+\tau_i s+\tau_i\tau_d s^2}{\tau_i s}\right)\hat{e}$. If proportional band is used to represent the gain, $PB = 100/K_c$, and reset rate is used instead of integral time, $\tau_r = 1/\tau_i$, the standard equation appears as $\hat{u} = \frac{100}{PB}\left(1 + \frac{\tau_r}{s} + \tau_d s\right)\hat{e}$. Although it appears different in these alternate mathematical representations, there is no new functionality. I showed PB and reset rate together, they can be independently applied.

Parallel: If the controller gain and actuating error are multiplied onto each term in the parenthesis, and coefficients combined, the standard form becomes $\hat{u} = K_P\hat{e} + K_I\frac{\hat{e}}{s} + K_D s\hat{e}$. Again, there is no alternate functionality, just the same P, I, and D. This version has been given multiple names. Independent gains, because of the three gains; and parallel gains, or just

parallel because of the structure. The ISA 5.9 committee is moving toward recommending the label 'parallel'.

The translation between tuning coefficients between 'standard' and 'parallel' is straight forward. For example, $K_D = K_c\tau_d$. So, if you have tuning rules for one, the translation to the other is simple. However, I think that it is best practice for an operating unit to choose one algorithm version to prevent confusion, and to standardize operational and tuning procedures.

Series: I was raised calling this the rate-before-reset version: $\hat{u} = K_c\left(1 + \frac{1}{\tau_i s}\right)(1 + \tau_d s)\hat{e}$. This too can be mathematically factored in many ways. Here the $(1 + \tau_d s)\hat{e}$ term is the leaded actuating error, a projection of what the actuating error is anticipated to become, $\hat{e}_{anticipated}$. Then the P and I operations are on the anticipated error, $\hat{u} = K_c\left(1 + \frac{1}{\tau_i s}\right)\hat{e}_{anticipated}$. In a block diagram this calculates the rate before the integral (often termed the reset function), hence the rate-before-reset name. But also, the block diagram suggests the name series, because the rate term (as a lead) happens prior to the integral. It is alternately known as the physically realizable controller because this is how D was added to PI functionality in the mechanical-pneumatic device era. If you multiply the two binomials and collect common terms, then the Laplace formulation becomes $\hat{u} = \left(K_c\frac{\tau_i+\tau_d}{\tau_i}\right)\left(1 + \frac{1}{(\tau_i+\tau_d)s} + \frac{\tau_i\tau_d}{\tau_i+\tau_d}s\right)\hat{e}$. This representation reveals the interaction of terms. If you adjust the derivative action, it also adjusts the P and I term coefficients. So, it is often termed the interacting or interactive version. Interacting as a controller description seems to convey a desirable "feature", but I think the interaction is really a confounding attribute. Note again, there is only the P, I and D functionalities. ISA 5.9 is moving toward recommending the label 'series'. Again, if you have tuning rules for either the 'standard' or the 'parallel' forms you can translate values to the 'series' form, but it is not so simple to convert the series coefficient values to the standard (you need the quadratic formula).

Note: In spite of there being several PID versions, all have, and only have, the same three functionalities of P, I, and D. One controller version is not functionally better than another. But, there are many ways to present the controller terms. So, for tuning, the user needs to know exactly how the vendor has structured the algorithm.

Note: The ISA 5.9 committee is moving toward recommending this nomenclature, but ISA has not yet declared any terminology as standard.

Other PID Modifications

There are many other embodiments for each of these three PID versions. These include:

- Velocity mode, which calculates incremental changes to the MV, not the MV value. In calculus notation for the standard controller: $\Delta u = \Delta t\, K_c \left(\frac{de}{dt} + \frac{1}{\tau_i} e + \tau_d \frac{d^2 e}{dt^2}\right)$, then $u = u_{prior} + \Delta u$. This eliminates the integral which can wind-up in any situation in which the controller is not in charge of the process. But velocity mode does let the controller output wind-up to 100% or 0%.
- An internal override to limit the controller integral value when the output exceeds the limits of 100% or 0% so that the integral does not wind-up.
- Reset feedback replaces the integral with a filter on the controller output, or an equivalent PV. This is a clever way to prevent integral wind-up with an additional benefit that it does not let the output go to a limit. This keeps the output at a take-over value.
- The standard or series forms could also use either PB or reset rate instead of gain or integral time.
- A vendor might have a noise filter on the derivative action. In the standard version this would appear as $\hat{u} = K_c \left(1 + \frac{1}{\tau_i s} + \frac{\tau_d s}{(1+\tau_f s)}\right) \hat{e}$, which of course can be presented in many mathematically equivalent ways. Often τ_f is a fraction of τ_d.
- The units on time are commonly seconds or minutes. Don't presume what the vendor or installer might have chosen. Check.
- The legacy units on controller gain is % (of full MV range) per % (of full CV range), often considered dimensionless. In the digital era, many vendors are providing an ability to use engineering units for the PV, which permits engineering units on K_c. Many find this to be convenient in cascade and ratio control strategies. Some vendors provide a unit conversion factor $\hat{u} = K_{units} K_c \left(1 + \frac{1}{\tau_i s} + \tau_d s\right) \hat{e}$, that keeps K_c as %MV/%CV. Be sure to check with your user manual.
- A vendor can choose any of many numerical methods to digitally calculate the integral (rectangle rule, trapezoid rule, Simpson's rule, etc.) or the derivative (two-point or three-point finite difference, etc.). The vendor's numerical method should be inconsequential to the user; but, for most to be stable or have fidelity to the ideal, the controller interval (scan time) should be at least a tenth of the smallest time-constant: $\Delta t < \min(\tau_i, \tau_d)$.
- There are several conventions for the initialization of terms while in the MAN mode such as setting the controller bias or integral value to the current MV value for bumpless transfer to AUTO. Similarly, initializing the setpoint to the current CV value.
- Setpoint softening, commonly a ramp or first-order evolution from the current CV value to the new SP value, might be an option to prevent a bump when there is a SP change.
- Options to calculate the derivative on the PV (D-on-x) not on the actuating error (D-on-e) to prevent one-sample output spikes on a set point change, or to calculate proportional on PV (P-on-x) not e (P-on-e) also prevent an output bump on a set point change.
- Gain scheduling: Here controller gain and time-constants are scheduled (changed) along with some key variable (perhaps the CV) that would indicate how the process gain, time-constants or delay changes with operating conditions. The controller coefficient values, could be placed in a look-up table or calculated from the key variable for smooth transitions.

A chapter in the Instrument Engineer's Handbook provides additional information on PID versions: Rhinehart, R. R., H. L. Wade and F. G. Shinskey "Control Modes - PID Variations," Instrument Engineers' Handbook, Vol II, Process Control and Analysis, 4th Edition, B. Liptak, Editor, Section 2.3, pp. 124-129, Taylor and Francis, CRC Press, Boca Raton, FL, 2005.

When tuning, you need to be aware of what options are selected to be sure that the modification is compatible with the method. For example, if part of tuning is to make step changes in the SP, you must remove set point softening features and choose P-on-e.

Other Control Algorithms

There are many alternate control algorithms to the basic PID controller. PID evolved in the mechanical-pneumatic pre-electronic days. Mechanical devices were replaced by analog electronic devices, then by digital devices. The digital devices permit many more sorts of calculations than just P, I and D, and first-order filters. Some, control techniques permitted by digital computers include:

- Model-Based Control (MBC), alternately called Horizon Predictive Control (HPC), Dynamic Matrix Control (DMC), or Advanced Process Control (APC). This traditionally uses linear vector impulse models of the process, and optimization to find the best sequence of future control actions to move the CV toward the set point, while avoiding constraints. It can handle MIMO processes and include delays and feedforward action. Typically, the models are linear (fixed and uniform gains) and stationary (fixed time constants and delays). This has gotten traction in the process industries, and many vendors offer their own version and range of options.
- Modern Control is alternately called State-Space, or ABCD Matrix model-based control. This is also linear, stationary, and MIMO. This has gotten traction in aeronautical and electronic applications.
- Generic Model Control (GMC). When the model is a steady state SISO model, this reduces to PI control with output characterization (the inverse of the SS model). The benefit is when processes are nonlinear, effectively have no or little delay, and time-constants do not change. I believe that this can be easily implemented by in-plant engineers in standard devices, and that use of first-principles models has other benefits for those managing the processes.
- Predictive Functional Control (PFC). This uses a model of the process to forecast what the process might do over the future in response to an MV change. The model could be a traditional linear model or a first-principles nonlinear model, which includes disturbance and nonideal transient effects. In any case, the MV value is selected to get the modeled value to match the SP value at some future time, the coincidence point. There is only one tuning value per CV, the future time of the coincidence point, which is placed beyond the delay or inverse action. These attributes provide many user benefits – tuning simplicity, robustness to ill-behaved dynamics and/or nonlinearity.

- Process-Model Based Control (PMBC). This was developed as a one-step-ahead model-based controller to use the process engineer's first-principles model of the process. It also includes a method to incrementally adjust model coefficients values to keep the model true as the process behavior changes. The single tuning coefficient per CV is the desired initial rate of the process returning to the setpoint. The adapting model can be valuable in both process monitoring and constraint adjustment in supervisory algorithms.

Understand how your controllers were setup by others, and which versions and modifications are best within your application context.

- - - - - - -

Russ Rhinehart started his career in the process industry. After 13 years and rising to engineering supervision, he transferred to a 31-year academic career. Now "retired", he enjoys coaching professionals through books, articles, short courses, and postings on his web site www.r3eda.com.