INTERACTIVE CONTROL ENGINEERING
Principles and Applications

Volume 2 Practical PID Control

CHAPTER 1

Introduction: PID Control in Practice

Pages 1–23

Edited by John D. Hedengren

How to cite this chapter. "Introduction: PID Control in Practice," in *Practical PID Control*, John D. Hedengren, Ed., vol. 2 of *Interactive Control Engineering: Principles and Applications*. Advanced Process Solutions, LLC, 2026, ch. 1, pp. 1–23.

The page numbers in this file are those of the printed volume.

The volume online: contents, other chapters and citation records
https://apmonitor.com/apps/books/v02-practical-pid/

Copyright © 2026 Advanced Process Solutions, LLC. All rights reserved. Produced by Advanced Process Solutions, LLC.

CHAPTER 1

Introduction: PID Control in Practice



The chapters 2 cruise control 3 blending tank 4 steam heat exchanger 5 cooling tower 6 pH neutralization 7 router queue 8 ramp meter 9 ride-hailing price 10 budget and debt 11 inflation targeting 12 servo axis

Notice that the five units on the left are machines and process plant with valves and probes, while the next five are a network, a road, a market and two public institutions, and the servo at the right closes the row with a machine again. Illustration, not to scale.

IN PLAIN TERMS **One Simple Rule for Many Kinds of Loops**

The main idea. A feedback controller compares what it measures with what it wants and moves a handle to close the gap. The common rule adds three parts: how big the gap is now, how long it has lasted, and how fast it is changing.

The challenge. On a real plant the handle stops at its end, the reading is noisy or late, and the controller may act only once a minute or once a year. The plant can also respond more strongly on one day than on another.

Why it matters. The same rule runs car speed, tanks, internet routers, freeway ramps, prices and interest rates, so setting it well pays almost everywhere.

THREE WAYS TO READ THIS CHAPTER

Explore	7 min	This page, Section 1.2 with its timeline and two History boxes, the Did you know? boxes in Sections 1.1, 1.4.3 and 1.5, Figure 1.11, and Key points in Section 1.8. No equations.
Study	45 min	The whole chapter in order, with each number of the oven example worked, then Exercises 1 to 10.
Practice	10 min	This page, Section 1.4.5 with Table 1.1, Section 1.4.9, Section 1.5 with Figures 1.7 and 1.9, and Section 1.8.

1.1 The Question This Volume Answers

The proportional-integral-derivative (PID) controller is three terms and three numbers, and it is still the algorithm behind most industrial feedback loops [1]. The equation is rarely what makes a loop fail. A loop fails because its actuator reaches a limit, its measurement is noisy or late, its controller runs once a minute, or its process gain at today's operating point is not the gain at which it was tuned. This volume answers one question: how is a PID loop built and tuned so that it works on a real plant, with limits, noise, dead time, sampling and a gain that moves?

Eleven chapters answer it on eleven plants, each with an app that the reader runs in a browser. Chapters 2 to 6 are process and vehicle loops: a cruise control, a blending tank, a steam-heated exchanger, a cooling tower basin and a pH neutralizer. Chapters 7 to 11 carry the same algorithm outside the plant, to a network router, a freeway ramp, a ride-hailing market, a national budget and a central bank, where the controller acts on a calendar and in fixed steps. Chapter 12 closes the volume on a servo axis whose three nested loops run at three rates.

A reader who has worked through the volume can do the following on a loop that is new to the reader.

- Read a vendor's PID equation, name its form and convert its settings to another form.
- Choose the controller action from the sign of the process gain.

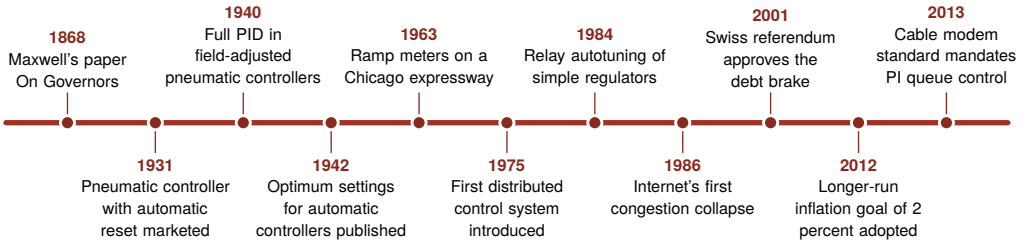


Figure 1.1 Milestones of PID control in practice. The dates are from [6] (1868), [7] (1931, 1940), [8] (1942), [9] (1963), [10] (1975), [11] (1984), [12] (1986), [13] (2001), [14] (2012) and [15] (2013). Notice that the three terms were settled in the process plant before 1942 and that the same feedback reached roads, networks and public policy later. Events are evenly spaced, not to a time scale.

- Fit a first-order or an integrating model with dead time to a step test, and tune from the fit with a named rule.
- Run a relay test and say what its result does and does not measure.
- Filter the derivative, protect the integral against windup and transfer between Manual and Auto without a bump.
- Say what a sample time, a dead time or a moving gain costs in stability margin, and decide when to schedule the controller gain.

This chapter develops each of these on one small example and names the chapter that uses it. Standard texts give fuller derivations [2–4].

> ✦ **DID YOU KNOW? A Hundred-Billion-Dollar Industry**
>
> A standard text of 2021 puts control and automation at a 100 billion dollar industry. Distributed control systems make up about 20 % of the market, and sensors, actuators, software and other components make up the rest [5].

1.2 A Short History

The milestones of Figure 1.1 follow one algorithm from the engine to the economy. In 1868 Maxwell described the governor as the part of a machine that keeps its velocity nearly uniform despite changes in the driving power or the resistance [6], the problem of Chapter 2.

Process plants took the three terms one at a time. A pneumatic controller with automatic reset, the action now called integral, was marketed in 1931, and derivative action, then called pre-act, was found the same year. By 1940 full PID was available in pneumatic controllers adjusted in the field [7]. A paper of 1942 gave practical ways to set the three knobs [7, 8], and pneumatic control then became the standard of the oil, gas and chemical industries [7].

The same feedback then left the plant. In 1963 ramp meters were set up on a Chicago expressway and run by hand by an officer in the field [9]. Chapter 8 closes that loop on detector reports. The first distributed control system was introduced in 1975 [10]. A relay test that tunes a simple regulator by itself was published in 1984 [11], and tuning from a model followed in 1986 [16] and in simple analytic rules in 2003 [17]. In October 1986 the Internet had the first of a series of congestion collapses [12], and in October 2013 a cable modem standard required a variant of a PI queue controller [15], the loop of Chapter 7.

> ⧖ HISTORY **By 1940: Three Knobs on a Pneumatic Controller**
>
> An early pneumatic proportional controller with a wide band was set with a gain low enough to limit hunting but high enough to keep the steady error small, and the setpoint was reset by hand so that the process stood on target despite the offset. A circuit that copied that manual correction, called automatic reset, is the integral action of today. Derivative action, then called pre-act, began as an accident: a capillary put into a feedback line by mistake gave the response a kick. By 1940 all three actions were available in pneumatic controllers adjusted in the field, three knobs turned largely by trial and error [7]. The offset of Equation (1.3) and the kick of Section 1.4.8 are the same two effects.

Public policy keeps a calendar of its own. Swiss voters approved a debt brake in a referendum of 2001, and it first applied to the federal budget of 2003 [13]. In January 2012 the monetary policy committee of the United States stated that inflation of 2 percent is most consistent with its mandate over the longer run [14]. Each rule gives a policy a published number to steer toward, the setpoint that Chapter 10 and Chapter 11 treat as the target of a loop.

> ⧖ HISTORY **1962 to 1975: From Relay Cabinets to Distributed Control**
>
> As instruments multiplied, bundles of pneumatic tubing ran into the control room. Wired transmitters replaced them, first with a signal of 10 to 50 mA and then, after the transistor, with the 4 to 20 mA range of the same ratio. A conference of 1962 discussed putting every action under the direct digital control of one computer, but process plants held back, with a large base of pneumatic and analog equipment and a fear that the one computer would fail. In 1968 the motor industry asked for a programmable replacement for its cabinets of clicking relays, and in 1975 the first distributed control system was introduced, still adjusting independent controllers [10]. Each loop of such a system is a digital PID that executes once per period, the sampled controller of Section 1.4.11.

1.3 Notation and Conventions

The chapters share the symbols of the notation page. The setpoint is SP, the process variable is PV and the manipulated variable is MV. In equations the measurement is y, the controller output is u and the error is $e = \mathrm{SP} - \mathrm{PV}$. The controller settings are the gain K_c, the integral time τ_I and the derivative time τ_D. The process model has a gain K_p, a time constant τ_p and a dead time θ_p. The closed-loop time constant chosen in a tuning rule is τ_c, and the control period is Δt.

A gain carries units. The process gain K_p is in PV units per MV unit, and the controller gain K_c is in MV units per PV unit, so the product $K_c K_p$ is a pure number. The MV is in percent of span unless a chapter says otherwise. Each chapter keeps the time unit of its plant: seconds for the cruise control and the router, milliseconds for the servo axis and minutes for the process units and the ramp meter. The ride-hailing zone runs in hours and the two economies in years. The running example of this chapter uses minutes, and frequencies are in radians per minute.

Three symbols need care across chapters. This chapter writes K_u and P_u for the ultimate gain and period, K' for the gain of an integrating process and $K_i = K_c/\tau_I$ for an integral gain. Chapter 2 uses K_u for the process gain from pedal to speed, and Chapter 5 writes K_I for the integrating gain of the basin. Each chapter defines its own symbols where they first appear.

1.4 PID Theory on One Loop

1.4.1 The Loop and the Running Example

The running example is a small electric oven. The MV is the heater power u in percent and the PV is the oven temperature y in °C. At rest the temperature is $y = 20 + 2u$, so the power range of 0 to 100 % covers 20 to 220 °C. The oven operates at $u = 50\,\%$ and $y = 120\,^{\circ}\mathrm{C}$. Its response to a change in power is first order with dead time, as shown in Equation (1.1).

$$G(s) = \frac{Y(s)}{U(s)} = \frac{K_p\, \mathrm{e}^{-\theta_p s}}{\tau_p s + 1}, \qquad K_p = 2\,^{\circ}\mathrm{C}/\%, \quad \tau_p = 10\,\mathrm{min}, \quad \theta_p = 2\,\mathrm{min} \tag{1.1}$$

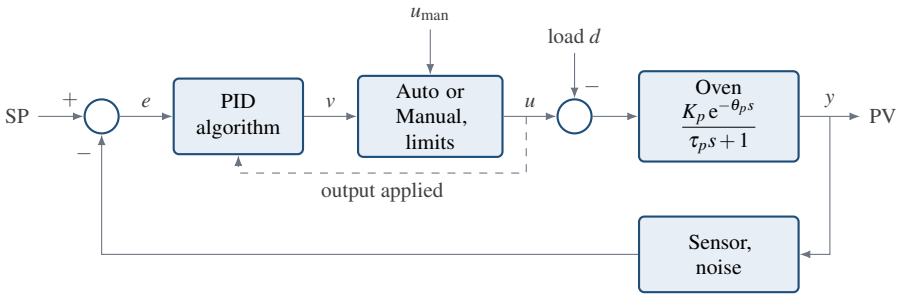


Figure 1.2 The Loop of the Running Example. The Dashed Signal Returns the Applied Output to the Algorithm for Anti-Windup and Tracking.

where Y and U are the transforms of the deviations of y and u from the operating point. Figure 1.2 shows the loop. A load d, such as a door left ajar, acts as a loss of heater power. The limits act on the output v that the algorithm computes, and the algorithm is told the output u that was applied. The controller executes every $\Delta t = 0.1$ min unless stated.

1.4.2 Three Forms of One Algorithm

Vendors write the same three terms in three ways. The ideal form, also called the standard or noninteracting form, is Equation (1.2).

$$u(t) = u_b + K_c \left[e(t) + \frac{1}{\tau_I} \int_0^t e\,dt' + \tau_D \frac{de}{dt} \right] \tag{1.2}$$

where u_b is the output bias. Most apps of this volume take their settings in this form.

Each term has one job. The proportional term acts on the present error, and alone it leaves an offset. After a setpoint step ΔSP the oven settles where the output change $K_c e$ produces the temperature change $K_p K_c e$, which gives Equation (1.3).

$$e_\infty = \frac{\Delta \text{SP}}{1 + K_c K_p} \tag{1.3}$$

With $K_c = 1.25\,\%/^\circ\text{C}$ the offset is $1/3.5 = 29\,\%$ of the step. The integral term keeps moving the output until the error is zero, and the derivative term opposes fast changes. Chapter 3 sets this offset beside one that a limit imposes, and Chapter 10 finds it in an economy's built-in stabilizers. The parallel form of Equation (1.4) gives each term its own gain.

$$u(t) = u_b + K_c\, e(t) + K_i \int_0^t e\,dt' + K_d \frac{de}{dt}, \qquad K_i = \frac{K_c}{\tau_I}, \quad K_d = K_c\, \tau_D \tag{1.4}$$

The ramp meter of Chapter 8 is set by such direct gains, and Chapter 2 writes its closed loop with K_i. The series form, also called interacting, multiplies a proportional-integral (PI) part by a derivative lead, as shown in Equation (1.5).

$$C(s) = K_c' \left(1 + \frac{1}{\tau_I' s} \right) \left(\tau_D' s + 1 \right) \tag{1.5}$$

Multiplying out gives the ideal settings that act on the loop, as shown in Equation (1.6).

$$K_c = K_c' \left(1 + \frac{\tau_D'}{\tau_I'} \right), \qquad \tau_I = \tau_I' + \tau_D', \qquad \tau_D = \frac{\tau_I' \tau_D'}{\tau_I' + \tau_D'} \tag{1.6}$$

The inverse exists only when $\tau_I \geq 4\tau_D$. A tuning used later in this section is $K_c = 3\,\%/^\circ\mathrm{C}$, $\tau_I = 4\,\mathrm{min}$ and $\tau_D = 1\,\mathrm{min}$ in the ideal form. Its parallel gains are $K_i = 0.75\,\%/(^\circ\mathrm{C\,min})$ and $K_d = 3\,\%\,\mathrm{min}/^\circ\mathrm{C}$, and its series settings are $K_c' = 1.5$, $\tau_I' = 2$ and $\tau_D' = 2$. The same three numbers typed into a series algorithm without conversion act as $K_c = 3.75$, $\tau_I = 5$ and $\tau_D = 0.8$, a gain 25 % higher than intended. With $\tau_D = 0$ the ideal and series forms are identical. No chapter of this volume runs a series algorithm. The form is given because settings taken from a chapter may be entered in an industrial controller that does.

A digital controller computes the terms at each execution k. The position form computes the whole output, as shown in Equation (1.7), and the velocity form computes its change, as shown in Equation (1.8).

$$u_k = u_b + K_c \left[e_k + \frac{\Delta t}{\tau_I} \sum_{j \leq k} e_j - \tau_D \frac{y_k - y_{k-1}}{\Delta t} \right] \tag{1.7}$$

$$u_k = u_{k-1} + K_c \left[(e_k - e_{k-1}) + \frac{\Delta t}{\tau_I} e_k - \tau_D \frac{y_k - 2y_{k-1} + y_{k-2}}{\Delta t} \right] \tag{1.8}$$

The two give the same output while the gains are constant and no limit is active. They differ in where the operating point is stored. The position form needs a bias and keeps a separate sum of the error. The velocity form has neither, because the last output is its memory. Every chapter of this volume runs a velocity form, and Section 1.6.2 gives the algorithm as the apps implement it.

1.4.3 Direct and Reverse Action

Feedback is negative only when the controller pushes against the error. The oven has a positive process gain, so a temperature above the setpoint must lower the heater power. The output falls as the PV rises, which is a reverse-acting controller. With the error written as $a(\mathrm{SP} - \mathrm{PV})$, it has $a = +1$. A loop that cools the oven with a damper would have $K_p < 0$ and would need $a = -1$, a direct-acting controller whose output rises with the PV. The test that never fails is the sign of the loop gain: $a K_c K_p$ must be positive. With the wrong action the loop gain is negative, each correction enlarges the error and the output runs to a limit.

Three chapters have a negative process gain. More marking shortens the router queue in Chapter 7, a higher price shortens the pickup time in Chapter 9 and a higher policy rate lowers inflation in Chapter 11. The two names are easy to confuse, so each of those chapters is best read by its error equation and not by the name.

> ✦ DID YOU KNOW? **A PID Behind a Monetary Rule**
>
> PID algorithms have been used in many fields, and one unconventional use is to explain popular monetary policy rules [5]. Chapter 11 runs a velocity-form PID and the Taylor rule in the same slot of one simulated bank, so the two can be compared on the same economy.

1.4.4 A Step Test and Its Fit

The model of Equation (1.1) is found from one experiment. The loop is put in Manual at a steady state, the MV is stepped by Δu and the PV is recorded until it settles. The gain is the PV change divided by Δu. Two times read from the record give the other two constants, as shown in Equation (1.9).

$$K_p = \frac{\Delta y}{\Delta u}, \qquad \tau_p = 1.5\,(t_{63} - t_{28}), \qquad \theta_p = t_{63} - \tau_p \tag{1.9}$$

where t_{28} and t_{63} are the times after the step at which the PV has covered 28.3 and 63.2 % of its change. A first-order response reaches those levels $\tau_p/3$ and τ_p after it starts to move.

Figure 1.3 shows the test on the oven. A step from 50 to 60 % raises the temperature from 120 toward 140 °C, so $K_p = 20/10 = 2\,^\circ\mathrm{C}/\%$. The PV passes 125.66 °C at $t_{28} = 5.33\,\mathrm{min}$ and 132.64 °C

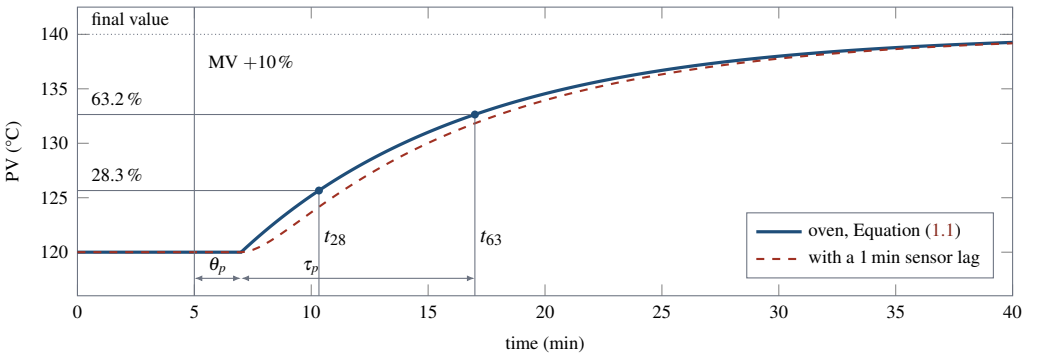


Figure 1.3 Step Test on the Oven With the Two-Point Fit. The MV Steps From 50 to 60 % at 5 min. The Solid Curve Is the Oven and the Dashed Curve Is the Same Test Read Through a 1 min Sensor Lag.

at $t_{63} = 12.00\,\text{min}$, which give $\tau_p = 10.0\,\text{min}$ and $\theta_p = 2.0\,\text{min}$. On a plant the step must stand clear of the noise and still be small enough to keep the process near its operating point. A second step in the opposite direction checks the fit.

A real response is never exactly first order, and the fit then returns effective constants. The dashed curve of Figure 1.3 adds a sensor lag of 1 min. The two-point reading of that record gives $\tau_p = 10.0\,\text{min}$ and $\theta_p = 3.0\,\text{min}$, and a least-squares fit gives 10.3 and 2.9 min. A small lag is read almost entirely as dead time. The half rule, which adds half of a neglected lag to each constant, would predict 10.5 and 2.5 min [17]. The fitted dead time is the number to tune with, because the loop feels the lag as delay. Chapter 4 fits a dead time of 0.94 min to a plant whose transport delay is 0.30 min, and Chapter 5 shows that a valve lag adds its full length to the apparent dead time of an integrating process. Chapter 2 shows the other hazard: on a nonlinear plant the fitted gain depends on the size of the step.

1.4.5 Tuning From the Model

Internal model control (IMC) tuning, also called lambda tuning, starts from the response the designer wants. The PV should follow a setpoint change as a first-order lag of time constant τ_c after the dead time, so the closed-loop transfer function is $T(s) = \mathrm{e}^{-\theta_p s}/(\tau_c s + 1)$. The controller that produces it is $C = T/[G(1-T)]$. Substituting Equation (1.1) and approximating the delay in the denominator by $\mathrm{e}^{-\theta_p s} \approx 1 - \theta_p s$ gives Equation (1.10).

$$C(s) = \frac{\tau_p s + 1}{K_p\,(\tau_c + \theta_p)\,s} = \frac{\tau_p}{K_p\,(\tau_c + \theta_p)}\left(1 + \frac{1}{\tau_p s}\right) \tag{1.10}$$

This is a PI controller [16]. The SIMC rule keeps its gain and shortens the integral time of a slow process so that loads are rejected sooner [17], as shown in Equation (1.11).

$$K_c = \frac{\tau_p}{K_p\,(\tau_c + \theta_p)}, \qquad \tau_I = \min\bigl(\tau_p,\, 4\,(\tau_c + \theta_p)\bigr) \tag{1.11}$$

The single choice is τ_c. A value equal to the dead time is fast with good margins, and a larger value is smoother. On the oven $\tau_c = \theta_p = 2\,\text{min}$ gives $K_c = 10/(2 \times 4) = 1.25\,\%/^\circ\text{C}$ and $\tau_I = \min(10, 16) = 10\,\text{min}$. Chapter 3, Chapter 4 and Chapter 6 tune with this rule.

The Ziegler–Nichols reaction-curve rule is older and far more aggressive [8]. It was designed for a decay ratio of one quarter, which accepts a large first overshoot. It is Equation (1.12).

$$K_c = \frac{1.2\,\tau_p}{K_p\,\theta_p}, \qquad \tau_I = 2\,\theta_p, \qquad \tau_D = 0.5\,\theta_p \tag{1.12}$$

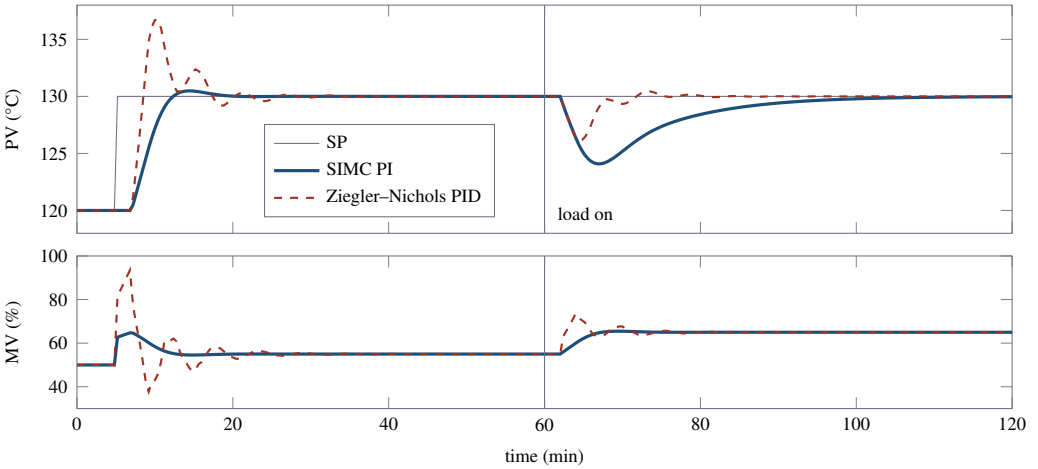


Figure 1.4 Two Tunings of the Oven Loop. The Setpoint Steps From 120 to 130 °C at 5 min and a Load of 10 % of Heater Power Enters at 60 min. Solid: SIMC PI. Dashed: Ziegler–Nichols PID With a Derivative Filter of 0.1 min.

Table 1.1 The Two Tunings of Figure 1.4. Settling Is Into ±0.2 °C. Margins Are Defined in Section 1.4.12.

	SIMC PI	Ziegler–Nichols PID
K_c (%/°C), τ_I, τ_D (min)	1.25, 10, 0	3, 4, 1
Setpoint step: overshoot (%)	4.8	67.7
Setpoint step: settling time (min)	12.5	20.2
Setpoint step: IAE (°C min)	43.7	54.8
Load step: largest deviation (°C)	5.9	3.9
Load step: IAE (°C min)	79.7	16.1
MV travel over the run (%)	37	220
Gain margin, phase margin	3.14, 61°	1.33, 39°

On the oven it gives $K_c = 3\,\%/^\circ\mathrm{C}$, $\tau_I = 4\,\mathrm{min}$ and $\tau_D = 1\,\mathrm{min}$, the tuning converted in Section 1.4.2. Chapter 4 and Chapter 5 load both rules from the reader's own fit.

Figure 1.4 runs the two tunings through a setpoint step of 10 °C and a load of 10 % of heater power, and Table 1.1 gives the measures. The comparison is a trade and not a ranking. The Ziegler–Nichols tuning rejects the load with a fifth of the integrated error, because its integral gain K_c/τ_I is six times larger. It pays with 68 % overshoot on the setpoint step, six times the heater travel and a gain margin of 1.33, so a 33 % rise in process gain makes it unstable. Such thin margins are the known weakness of the rule [18]. The SIMC tuning tolerates a tripled gain.

Both rules divide by the dead time, so both inherit its reading error. A dead time read as 1 min doubles the Ziegler–Nichols gain to 6.0. The SIMC gain rises by one third, to 1.67, when τ_c is held at 2 min, because the dead time is added to τ_c before it divides. It also doubles if τ_c is tied to the misread value.

1.4.6 An Integrating Process

Some processes do not settle after a step. The basin level of Chapter 5 ramps for as long as makeup and losses differ. The model is Equation (1.13), and the step test reads a slope and a delay in place of a gain and a time constant.

$$G(s) = \frac{K' \mathrm{e}^{-\theta_p s}}{s}, \qquad K' = \frac{\text{change in slope of the PV}}{\Delta u} \tag{1.13}$$

An oven with no heat loss is the integrating variant of the running example. It keeps the initial slope of the real oven, $K' = K_p/\tau_p = 0.2\,^{\circ}\mathrm{C}/\mathrm{min}$ per percent, and the dead time of 2 min. The SIMC settings follow from Equation (1.11) as τ_p grows without bound at a fixed K_p/τ_p, as shown in Equation (1.14).

$$K_c = \frac{1}{K'(\tau_c + \theta_p)}, \qquad \tau_I = 4(\tau_c + \theta_p) \tag{1.14}$$

With $\tau_c = \theta_p$ the gain is again 1.25 %/°C and $\tau_I = 16\,\mathrm{min}$. The minimum in Equation (1.11) therefore tunes any process with $\tau_p > 4(\tau_c + \theta_p)$ as an integrator.

Two properties separate this loop from the self-regulating one. Proportional control alone tracks a setpoint with no offset, since the process supplies the integrator. A load d still leaves an offset of d/K_c, which is 8 °C for a 10 % load, so integral action is kept. The integral time must be long, because the controller and the process together put two integrators in the loop. On a setpoint step of 5 °C the overshoot is 29 % at $\tau_I = 16\,\mathrm{min}$, 51 % at 8 min and 90 % at 4 min, and at 2 min the loop is unstable. Shortening the integral time speeds up a self-regulating loop and destabilizes this one. Chapter 5 measures the same effect on its basin.

1.4.7 The Ultimate Gain and the Relay Test

A second way to characterize a loop needs no model. Under proportional control the loop oscillates steadily at one gain, the ultimate gain K_u, with the ultimate period P_u. The frequency $\omega_u = 2\pi/P_u$ is where the process phase reaches −180°, as shown in Equation (1.15).

$$\omega_u \theta_p + \arctan(\omega_u \tau_p) = \pi, \qquad K_u = \frac{1}{|G(j\omega_u)|} = \frac{\sqrt{1 + \omega_u^2 \tau_p^2}}{K_p} \tag{1.15}$$

For the oven $\omega_u = 0.844\,\mathrm{rad}/\mathrm{min}$, $P_u = 7.44\,\mathrm{min}$ and $K_u = 4.25\,\%/^{\circ}\mathrm{C}$. The integrating variant has $K_u = \pi/(2K'\theta_p) = 3.93$ and $P_u = 4\theta_p = 8\,\mathrm{min}$, the formulas Chapter 5 uses.

Raising the gain until a plant oscillates is slow and risky. The relay test finds the same point safely [11]. The controller is replaced by a relay that holds the MV at $u_0 + \delta$ while the PV is below the setpoint and at $u_0 - \delta$ while it is above. The loop settles into a bounded cycle near P_u, and the swing $\pm\alpha$ of the PV gives the gain, as shown in Equation (1.16).

$$K_u \approx \frac{4\,\delta}{\pi\sqrt{\alpha^2 - \varepsilon^2}} \tag{1.16}$$

where ε is the half-width of a switching band added so that noise does not chatter the relay, zero for an ideal relay. Figure 1.5 shows the test on the oven with $\delta = 10\,\%$ and no band. The cycle has a period of 7.6 min, 2 % longer than P_u, and a swing of ±3.75 °C, so Equation (1.16) returns $K_u = 3.39$. That is 20 % below the true value. The equation assumes a sine wave, and this PV is closer to a triangle, whose fundamental is only 0.81 of its peak. Chapter 12 avoids the error by taking the fundamentals of both signals.

Table 1.2 lists the rules that turn K_u and P_u into settings, with their values at the oven's true ultimate point. The Tyreus–Luyben settings [19] land close to the SIMC tuning, with a phase margin of 69°, against 36° for the Ziegler–Nichols PI.

A relay test measures one point of the frequency response and nothing else. Chapter 2 and Chapter 3 find that the switching band and the noise, not the plant, set their cycle, and Chapter 12 finds that its relay measures a coupling resonance and not the motion the loop must control.

1.4.8 Derivative Action and Its Filter

The derivative term adds phase lead, and it multiplies whatever changes fastest in its input. Two precautions are standard. The first takes the derivative of the measurement and not of the error, as Equations (1.7) and (1.8) already do. With the Ziegler–Nichols tuning a setpoint step of 10 °C

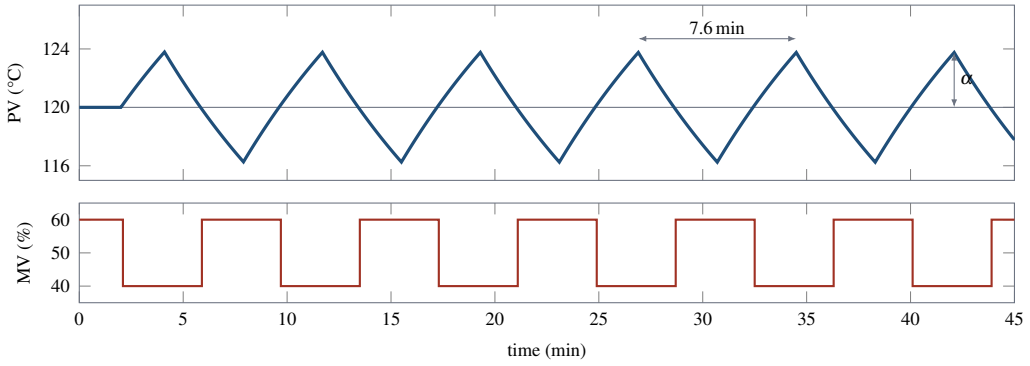


Figure 1.5 Relay Test on the Oven. The Relay Switches the Heater Between 40 and 60 % Each Time the PV Crosses 120 °C.

Table 1.2 Tuning Rules Built on the Ultimate Point, and Their Values for the Oven ($K_u = 4.25$, $P_u = 7.44$ min)

Rule	Settings	On the oven	Used in
Ziegler–Nichols PI	$0.45K_u$, $P_u/1.2$	1.91, 6.2 min	Chapter 12
Ziegler–Nichols PID	$0.6K_u$, $P_u/2$, $P_u/8$	2.55, 3.7, 0.93 min	Chapter 5
Tyreus–Luyben PI	$K_u/3.2$, $2.2P_u$	1.33, 16.4 min	Chapter 2, Chapter 3

Table 1.3 Derivative Filter on the Ziegler–Nichols PID With Measurement Noise of 0.2 °C

Filter time constant τ_f (min)	none	0.1	0.25	0.5
Standard deviation of the MV (%)	11.3	5.6	2.8	1.8
Phase margin (°)	41.0	39.3	34.9	23.4

differentiated through the error would ask for $K_c\tau_D\Delta\text{SP}/\Delta t = 300\,\%$ of output for one execution. Differentiating the measurement removes that derivative kick and leaves the proportional kick of 30 %.

The second precaution is a filter, because measurement noise changes at every execution. A first-order filter of time constant τ_f turns the term into Equation (1.17).

$$D(s) = -\frac{K_c\,\tau_D\,s}{\tau_f s + 1}\,Y(s) \tag{1.17}$$

Its gain at high frequency is limited to $K_c\tau_D/\tau_f$. Without the filter the gain per execution is $K_c\tau_D/\Delta t$, which is 30 %/°C here. Table 1.3 shows the trade on the oven with noise of standard deviation 0.2 °C. A filter of $\tau_D/10$ halves the heater movement for 2° of phase margin. Longer filters quiet the heater further and take margin faster, so the filter belongs to the tuning and is not an afterthought [20]. The SIMC PI moves the heater with a standard deviation of a quarter of a percent on the same noise, which is why many loops run without derivative action.

Chapter 2 and Chapter 5 measure derivative action on a noisy speedometer and a noisy level. Chapter 10 finds that a derivative term helps its lagged loop on clean data and hurts it on published data, and Chapter 7 explains why derivative action earns little when dead time dominates.

1.4.9 Windup and Three Remedies

An actuator at a limit breaks the loop. The error persists, the integral keeps growing, and when the PV finally arrives the stored integral holds the output at the limit long after it should have left [21]. A setpoint step from 120 to 200 °C shows it on the oven with the SIMC PI. The new setpoint needs 90 %

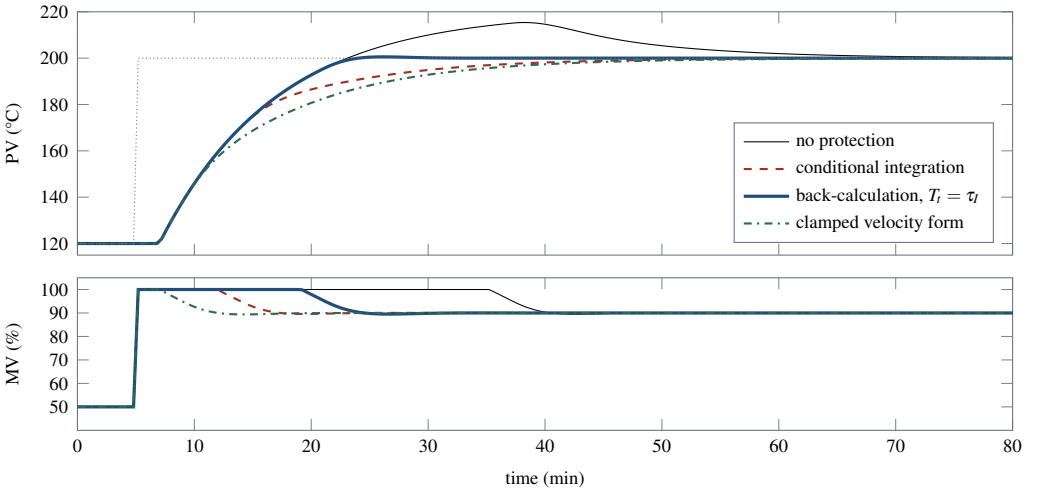


Figure 1.6 Windup on a Setpoint Step From 120 to 200 °C at 5 min, SIMC PI. Thin Solid: No Protection. Dashed: Conditional Integration. Thick Solid: Back-Calculation. Dash-Dotted: Clamped Velocity Form. The Dotted Line Is the Setpoint.

of power, and at the step the proportional term alone raises the computed output to 150 %. Three remedies are in common use [4, 22].

- *Conditional integration* stops the integral while the output is limited and the error would drive it further into the limit.
- *Back-calculation* feeds the difference between the applied and the computed output back to the integral through a tracking time constant T_t, as in Equation (1.26).
- *A clamped velocity form* limits the running output of Equation (1.8) itself, so that nothing is stored beyond the limit.

Figure 1.6 and Table 1.4 compare them with an unprotected integral. Without protection the integral term reaches 80 %, the heater stays at full power for 30 min, and the oven overshoots by 15.4 °C. All three remedies cut the overshoot to less than 1 °C. They differ in when the output leaves the limit. The clamped velocity form leaves after 2 min, as soon as the PV starts to rise, and then creeps to the setpoint. Back-calculation with $T_t = \tau_I$ holds full power for 14 min and settles in less than half the time of the others. A shorter T_t is not better here: at $T_t = 2$ min the settling time grows from 18 to 42 min.

The setting $T_t = \tau_I$ has a simple reading. With the output at a limit u_{lim} and no derivative term, the proportional parts of Equation (1.26) cancel and leave Equation (1.18).

$$\frac{\mathrm{d}(u_b + I)}{\mathrm{d}t} = \frac{u_{\text{lim}} - (u_b + I)}{\tau_I} \tag{1.18}$$

The stored output $u_b + I$ relaxes toward the limit with time constant τ_I, whatever the error, and the computed output exceeds the limit only by the proportional term $K_c e$. On the oven the stored output has reached 88 % when the heater leaves full power, close to the 90 % the new setpoint needs. Chapter 2, Chapter 3 and Chapter 7 derive the same relaxation for their loops.

Clamping the running output has a second cost. Every increment that crosses the limit is discarded and never restored. With integral action off, the clamped form settles this step at 64.3 % of power where the position form gives 78.6 %, because the 50 % discarded at the limit is lost. Noise does the same in small steps. With a setpoint of 225 °C, which is out of reach, and noise of 0.5 °C, the clamped PI averages 99.3 % of power and holds the oven 1.4 °C below what full power reaches.

Table 1.4 The Four Runs of Figure 1.6. Times Are From the Setpoint Step. Settling Is Into ±1 °C.

Integral protection	Leaves the limit (min)	Peak PV (°C)	Settling (min)	IAE (°C min)
None	30.3	215.4	61.8	960
Conditional integration	7.2	200.0	41.2	779
Back-calculation, $T_t = \tau_I$	14.2	200.6	18.0	642
Clamped velocity form	2.1	200.0	44.9	884

Table 1.5 Effect of the Control Period on the Oven Loop. The Tuning in the Last Two Rows Is the SIMC PI With $K_c = 1.25$ and $\tau_I = 10$ min.

Control period Δt (min)	0.1	0.5	1	2
Ultimate gain, exact sampled loop	4.15	3.79	3.39	2.76
Ultimate gain with $\theta_p + \Delta t/2$ in Equation (1.15)	4.16	3.82	3.47	2.95
Phase margin (°)	60.7	57.9	54.6	48.1
Overshoot of a setpoint step (%)	4.8	8.2	12.8	23.4

Back-calculation stays at 100 %. For this reason the apps limit only the output they emit and correct only the integral, as Section 1.6.2 describes.

Every chapter meets a limit. Chapter 3 is built on one, and it computes the overshoot that back-calculation avoids. Chapter 4 and Chapter 6 let the reader switch the protection off, and the second adds a rate limit, which winds an integral up far from either stop. Chapter 11 winds its integral up at the floor of the policy rate.

1.4.10 Bumpless Transfer

A switch from Manual to Auto should not move the actuator. It does when the algorithm starts from a state that does not match the output in force. Suppose the oven is in Manual at 50 % with an error of 4 °C for 20 min. A position form whose integral kept running in Manual has stored $(K_c/\tau_I)\,e\,t = 10\,\%$, and its first automatic output is $50 + 5 + 10 = 65\,\%$, a bump of 15 %. Resetting the integral to zero at the switch still leaves the proportional bump of 5 %. The remedy is tracking: while in Manual the integral is set at every execution so that the computed output equals the manual one. The first automatic output is then 50 %, and the heater moves by the integral increment of 0.05 % per execution. The conditioning technique treats windup and transfer as this one problem [23].

A change of gain is a second source of bumps. Doubling K_c while the error is 4 °C moves the output by 5 % at once in the position form. Equation (1.8) does not move it, because it applies the new gain only to later changes of the error. The apps form each increment from the whole proportional term, so a new gain acts at once, as Chapter 2 and Chapter 3 note. The chapters on markets and economies apply tracking to a different switch: Chapter 9, Chapter 10 and Chapter 11 keep an idle policy tracking the active one.

1.4.11 Sample Time

A digital controller reads the PV once per period and holds its output until the next execution. On average the held output is half a period old, so sampling acts on the loop much like an added dead time of $\Delta t/2$ [17]. Table 1.5 tests that rule on the oven against the exact sampled loop. The ultimate gain falls from 4.25 to 2.76 as the period grows to 2 min, and the SIMC PI loses 13° of phase margin and overshoots by 23 %. The rule predicts each ultimate gain within 7 %.

The remedy is to tune for the dead time the loop has. With $\theta_p + \Delta t/2 = 3$ min in Equation (1.11) the gain at $\Delta t = 2$ min falls to 0.83, the phase margin returns to 60° and the overshoot to 4.9 %. By the same rule a period below a fifth of the dead time costs this tuning less than 3°. A long period is not always a choice. The ramp meter of Chapter 8 acts on one detector report a minute, and the

bank of Chapter 11 meets eight times a year. The budget of Chapter 10 is decided once a year, which halves the gain its loop can use. Chapters 9 to 11 also round the output to a fixed step, which can leave a small sustained cycle such as the one Chapter 11 measures. The drive of Chapter 12 runs three loops at three rates and counts the phase each hold costs.

1.4.12 Margins With Dead Time

Two numbers say how far a loop is from instability [5]. Let $L(j\omega) = C(j\omega)\,G(j\omega)$ be the frequency response around the loop. The gain margin is the factor by which the loop gain can rise before the loop oscillates. The phase margin is the phase lag that can be added at the crossover frequency ω_c. Both are defined in Equation (1.19).

$$\mathrm{GM} = \frac{1}{|L(j\omega_{180})|}, \qquad \mathrm{PM} = 180^\circ + \angle L(j\omega_c), \qquad |L(j\omega_c)| = 1 \tag{1.19}$$

where ω_{180} is the frequency at which the phase of L is -180°. Dead time is the reason margins run out. A delay θ_p leaves the magnitude alone and subtracts $\omega\theta_p$ radians of phase, without limit as the frequency rises.

The SIMC PI with $\tau_I = \tau_p$ makes the loop simple enough to solve by hand. The integral zero cancels the process lag and $L(s) = \mathrm{e}^{-\theta_p s}/[(\tau_c + \theta_p)s]$, which gives Equation (1.20).

$$\omega_c = \frac{1}{\tau_c + \theta_p}, \qquad \mathrm{PM} = \frac{\pi}{2} - \frac{\theta_p}{\tau_c + \theta_p}, \qquad \mathrm{GM} = \frac{\pi}{2}\,\frac{\tau_c + \theta_p}{\theta_p} \tag{1.20}$$

with the phase margin in radians. For $\tau_c = \theta_p$ the phase margin is 61.4° and the gain margin is $\pi = 3.14$, whatever τ_p and K_p are. The loop also tolerates a dead time up to 3.14 times the fitted one. Solving the phase margin for the crossover gives the bound of Equation (1.21).

$$\omega_c\,\theta_p \le \frac{\pi}{2} - \mathrm{PM} \tag{1.21}$$

A phase margin of 60° limits the crossover to $0.52/\theta_p$, 45° to $0.79/\theta_p$ and 30° to $1.05/\theta_p$. This is the rule of thumb that a loop cannot be made faster than about $1/\theta_p$ [24]. It is exact for a PI controller whose integral time cancels the lag. Derivative action buys a little more: the Ziegler–Nichols PID of Table 1.1 crosses at $1.26/\theta_p$ with a phase margin of 39°. No tuning removes the delay itself. Volume 5 treats the predictors that work around it.

Chapter 7 is this subsection on a router, where the dead time is the round-trip time and a longer path takes the phase margin from 68° to zero with no setting touched. Chapter 12 reads both margins from a measured frequency response, and Chapter 4 has a dead time that shortens as throughput rises.

1.4.13 Scheduling a Gain

Every tuning rule above sets K_c in inverse proportion to K_p. When the process gain moves with the operating point, a fixed controller gain makes the loop gain $K_c K_p$ move with it. The gain margin says how much rise is tolerated: a factor of 3.14 for the SIMC PI with $\tau_c = \theta_p$. Suppose the oven's gain at another operating point is two or three times the fitted 2 °C/%. With K_c fixed at 1.25 the phase margin falls from 61° to 33 and 4°, and the overshoot of a setpoint step rises from 4.8 to 52 and 103 %. At half the fitted gain the loop is stable and slow, reaching 63 % of a step in 8.1 min against 4.6 min.

Gain scheduling removes the variation by changing the controller gain with a measured variable x that tells where the process is [25], as shown in Equation (1.22).

$$K_c(x) = K_c^0\,\frac{K_p^0}{K_p(x)} \tag{1.22}$$

where K_c^0 is the gain tuned at the point where the process gain is K_p^0. On the oven the scheduled gains are 0.625 and 0.417 at two and three times the fitted gain, and the step response returns to 4.8 %

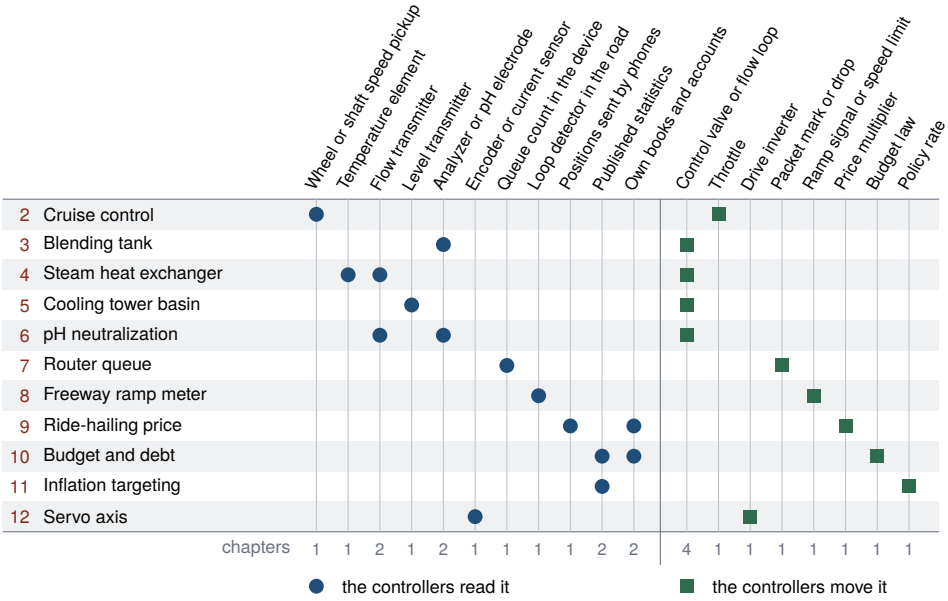


Figure 1.7 What the controllers of each chapter read and move: a round mark where the chapter's controllers read the source and a square mark where they move the element, as its variables table and its technology section name them. A value computed from other readings, such as the pickup time of Chapter 9, is marked under the source it is computed from. Notice that four chapters read no field instrument at all.

overshoot in each case. The schedule needs a gain curve and a measurement to enter it with, and it is only as good as both.

Chapter 6 is the extreme case. Its process gain changes by a factor of 1.96×10^4 across one valve stroke, far beyond any gain margin, and its schedule follows the slope of an assumed titration curve. Chapter 2 is the mild case. Its gain falls by a factor of three with speed, but its time constant falls with it, and Equation (1.11) depends on the ratio τ_p/K_p, which stays fixed. The proportional gain of Chapter 3 is constant for the same reason. In Chapter 9 the gain from price to pickup time grows twentyfold as the market nears a fold.

1.5 Instruments and Final Elements in This Volume

Figure 1.7 lists what the controllers of each chapter read and what they move. Seven chapters read a field instrument, from a wheel-speed pickup to a detector loop cut into the road. The other four read a count inside a router, positions sent by phones, published statistics or their own books, and they move a packet decision, a price, a budget or a policy rate. A control valve or a flow loop is the final element of four chapters, more than any other.

A control valve with excess stem friction moves in jerks and does not answer small reversals of its signal, an effect called stiction. In valves without a positioner to correct the stem, the resulting hysteresis can exceed 10 % [26]. A positioner is a closed-loop controller in its own right [26], the inner loop of Figure 1.8. With integral action a sticking valve makes the loop hunt above and below the setpoint, and the only cure is to reduce the friction [26]. The dead band of Chapter 4 stands for it.

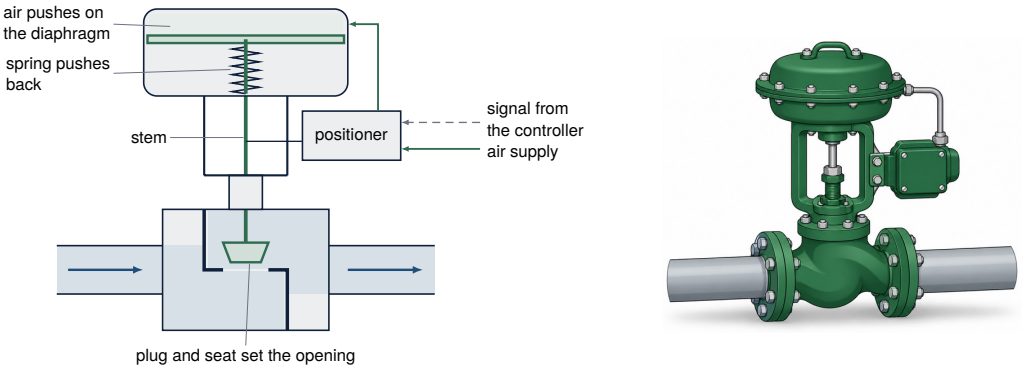


Figure 1.8 Principle of a globe control valve with its positioner. Notice that the positioner compares the controller's signal with the stem position, so the PID above it sees a valve that goes where it is sent until friction or the supply runs out (Chapter 4, Chapter 5 and Chapter 6).

> **DID YOU KNOW? A Zero That Is Not Zero**
>
> The 3 to 15 psi pneumatic and the 4 to 20 mA electric signal standards both begin above zero. This live zero tells a true reading of 0 % from a failed signal, such as a leaking tube or a severed cable [26].

A pH electrode is a glass bulb whose voltage grows by 59.17 mV at 25 °C for each tenfold change in hydrogen ion activity, one pH unit. The voltage also depends on temperature, so many probes carry a resistance thermometer [26]. That reading is the process variable of Chapter 6, whose process gain spans four decades.

Four chapters have no transmitter. Output is compiled from surveys and records: the first estimate of a quarter appears about a month after it ends and is revised several times, and a monthly price index is released in the middle of the following month [27]. A phone's satellite position is typically good to 4.9 m under open sky and worse near buildings [28]. A policy rate is moved by administered rates, among them the rate on reserve balances, set to support the committee's target range [29]. Late data, rounding and a calendar are the instruments of these loops, and Section 1.4.11 prices them.

PID control appears in dedicated devices and in plants with thousands of controllers, alone, inside distributed systems and in embedded systems, and higher controllers give setpoints to the PID layer [5]. Model predictive control sits one level above the PID loops that move the valves [30]. Figure 1.9 places the volume's loops on their platforms. Industrial loops run with cycle times from milliseconds to minutes [31]. The apps go further, from the 0.1 ms current loop of Chapter 12 to the budget of Chapter 10, decided once a year.

1.6 The Simulator Framework

Every chapter of this book has an app that runs in a web browser, collected with the others at https://apmonitor.com/apps/. About 240 apps exist across the series. All but a few are built on one shared framework, so the engine, the PID algorithm, the trends and the data export behave the same way in every chapter. The exceptions are standalone studios, such as the cascade studio and the two Temperature Control Lab apps, which follow the same ideas in their own code. This section describes the framework once and states where the standalone apps differ. Table 1.6 lists what a reader meets.

1.6.1 The Simulation Engine

Each app carries a plant model written as ordinary differential equations $\frac{\mathrm{d}x}{\mathrm{d}t} = f(x,u,d,p)$ in the states x, the inputs u, the disturbances d and the parameters p. The engine integrates the model with the classical fourth-order Runge–Kutta method on a fixed step h set for the fastest dynamics of the

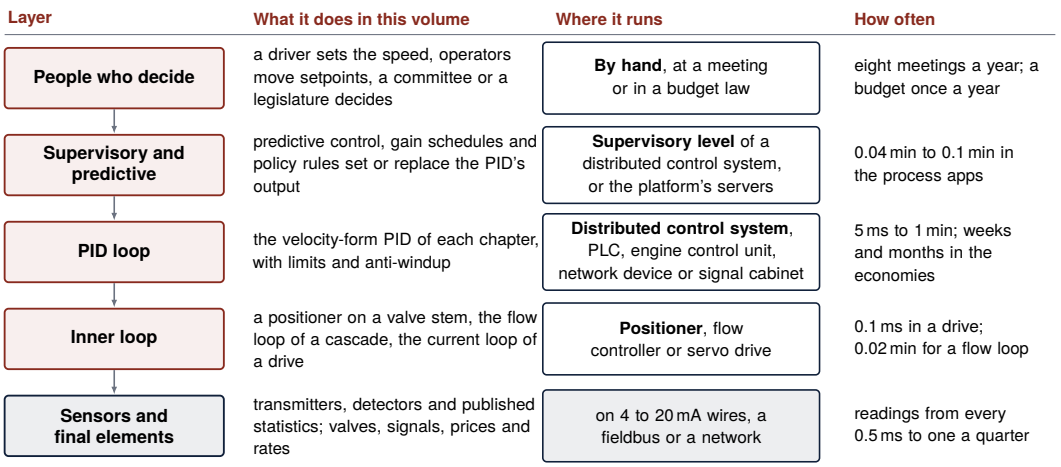


Figure 1.9 The control layers of this volume, the platform that hosts each and how often each acts. Platforms are from [5], [30] and [31], and the app periods from the chapters' technology sections. Notice that the economies sit in the same layers as the plants, with a meeting or a budget law in place of a computer.

Table 1.6 What Every App Provides. Standalone Apps Are Described in Their Own Chapters.

Element	What it does
Run, Pause, Reset	Start, hold and restart the run. Reset keeps the reader's settings.
Speed	Simulated time per second of real time. It changes the pace and never the result.
Manual and Auto	Hand each actuator to the reader or to the controller, without a bump.
Scenario list	Numbered cases, each with a briefing, a completion rule and a reload button.
Control panel	Setpoints, manual outputs, tuning, options and disturbances, at a Beginner or an Advanced level.
Trends	SP and PV above the MV for each loop, with limits, event marks and a cursor.
Scorecard	IAE, overshoot, settling time, MV travel and time at a limit for each loop.
Learn card	Predict, Try It, two hints and Explain, opened by the reader in that order.
Show Model	Equations, live variable values, editable parameters and assumptions.
Export CSV	Every logged sample of the run as a text file with a header row.
Instructions	A separate page with background, an operating guide and the model's limitations.

app. Some apps supply their own step, such as explicit Euler for a discontinuous right-hand side or an exact update for a model driven by events.

Three clocks run at separate rates. The plant advances every step h. The controller executes every control period Δt, equal to h unless the app declares a slower controller, and holds its outputs between executions. The log takes one sample at a fixed interval of simulated time for the trends, the scorecard and the exported file. The order of work in one step is shown in Figure 1.10. The controller also executes once when a scenario is loaded, so every run starts from an initialized controller.

Drawing the screen is kept apart from these clocks. A speed of 5 means five units of the app's time base per second of real time. Speed, frame rate and pauses change how fast a run is shown and never what it computes, so a phone and a workstation produce the same trajectory and the same file.

Noise comes from a deterministic pseudorandom generator. Its seed is declared by the app, or by a scenario that needs its own noise record, and it is restored whenever a scenario is loaded or Reset is pressed. The framework gives the reader no seed setting, so *default seed* in this book means the seed the app ships with. A run repeats bit for bit when the same scenario is loaded and the same settings are made before Run. A change made by hand during a run lands on a different step each time, which

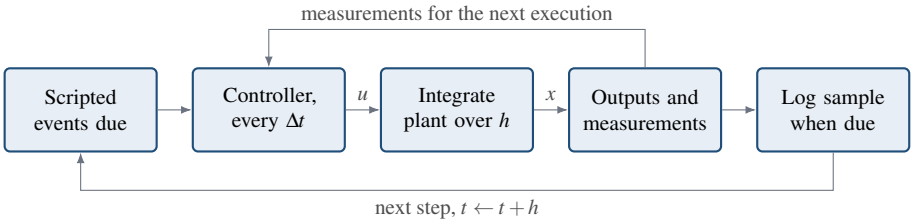


Figure 1.10 Work of the Engine in One Integration Step

is one reason numeric answers carry a tolerance. The standalone simulators draw noise from the browser's own generator, so a noisy run there repeats in its statistics and not sample for sample.

Run and Pause start and hold the clock. Reset restarts the run at $t = 0$, paused, and keeps the reader's settings: every setpoint, tuning value, option, model parameter, disturbance slider and mode the reader changed is put back before the plant and the controller are initialized. Scripted events replay, and a one-shot disturbance button pressed during the run is cleared. The reload button beside the scenario list restores the scenario's own settings instead.

1.6.2 The Velocity-Form PID Algorithm

One PID implementation serves the PID loops of every framework app. It is the velocity (incremental) form: each execution adds the change in the proportional and derivative terms to a running sum, and the integral is a separate term, as shown in Equations (1.23) to (1.26) for execution k and control period Δt.

$$e_k = a\,(\mathrm{SP}_k - \mathrm{PV}_k), \qquad P_k = K_c\,e_k, \qquad D_k = -a\,K_c\,\tau_D\,\frac{\mathrm{PV}_k^f - \mathrm{PV}_{k-1}^f}{\Delta t} \tag{1.23}$$

$$S_k = S_{k-1} + (P_k - P_{k-1}) + (D_k - D_{k-1}) \tag{1.24}$$

$$v_k = u_b + S_k + I_{k-1}, \qquad u_k = \mathrm{sat}(v_k) \tag{1.25}$$

$$I_k = I_{k-1} + \left[\frac{K_c}{\tau_I}\,e_k + \frac{u_k - v_k}{T_t}\right]\Delta t \tag{1.26}$$

where $a = \pm 1$ is the controller action, PV^f is the measurement after an optional first-order filter, u_b is the output bias, S is the running sum of proportional and derivative increments, v is the output before limiting, u is the output sent to the actuator, I is the integral term and T_t is the tracking time constant, equal to τ_I unless a loop sets another value. The function sat clamps v_k to the actuator range and then applies a rate limit if the loop declares one.

The limit acts on the emitted output u_k only. The sum S_k is never clamped, so it equals $P_k + D_k$ and the output before limiting is the position form $u_b + K_c e_k + D_k + I_{k-1}$. Anti-windup touches the integral alone. While the output is limited, the last term of Equation (1.26) drives I toward the value that reproduces the output applied, which is back-calculation [21]. With no integral action, Equation (1.26) is skipped. Clamping the running sum would instead discard each increment that crosses a limit and keep each one that returns, so symmetric noise would push the output off the limit even in a proportional-only loop.

The derivative acts on the measurement, so a setpoint step gives a proportional kick and no derivative kick. A change of K_c acts on the whole error at the next execution, as in the position form. The action $a = +1$ suits a process whose PV rises with the MV, a reverse-acting controller in the ISA convention, and $a = -1$ gives the direct-acting one.

In Manual the controller keeps executing and sets $I_k = u_{\mathrm{man}} - u_b - S_k$, so the output it would send equals the manual value u_{man}. The first automatic output differs from it by one period's change in P and D. In Auto the engine copies each controller output to the manual station, so a switch to Manual

is also bumpless. The standalone studios with PID loops use velocity-form code of their own, with the limit on the output and anti-windup on the integral alone, in which a gain change applies only to later increments.

1.6.3 The Predictive Control and Estimator Options

More than 100 apps offer model predictive control (MPC) as an option beside the structure they open with, which stays the default. Volume 9 develops the theory. In most of them the controller plans with a small linear model in discrete time as shown in Equation (1.27).

$$x_{k+1} = A\,x_k + B\,u_k + E\,w_k + c, \qquad y_k = C\,x_k \tag{1.27}$$

where w is a measured or scheduled disturbance and c is a constant left by linearization. The model comes from a step test at the design point, from the app's balances linearized there or again at every decision, or from the model the reader fits. It never reads the true states or parameters of the run, so a parameter the reader changes in Show Model is model mismatch, as on a real plant.

At each decision the controller minimizes the objective of Equation (1.28) over a horizon of N steps.

$$J = \sum_{k=1}^{N} (y_k - r_k)^{\top} Q\,(y_k - r_k) + \sum_{j} \Delta u_j^{\top} R\,\Delta u_j + \sum_{i} \rho_i\, s_i^2 \tag{1.28}$$

where r is the setpoint, Q weights the tracking error, R is the move suppression on each input change Δu_j and ρ_i is the price of a slack $s_i \geq 0$ that lets an output band be broken when it cannot be held. Actuator limits are hard constraints. The quadratic program is solved in the browser by a dual active-set method [32]. The first move is applied, and the plan is made again at the next decision.

The plan starts from an estimate of the model state. About half of these apps use a steady-state Kalman filter on the model with an added disturbance state, which absorbs what the model did not predict and removes steady offset without an integral term [33]. Others correct the model output with a filtered bias to the same end. About a dozen apps also offer moving horizon estimation (MHE), which solves the estimation problem over a window of recent samples and can hold bounds that a Kalman filter cannot, such as a concentration that is never negative [34].

With the MPC selected, the tuning fields of the loops it replaces leave the Control panel. In most of these apps the Advanced level shows up to three settings, each a multiple of the shipped value: the prediction horizon, the move suppression and the band price. The trend draws the accepted plan ahead of the present time, and a list of constraint prices shows which limits bind. A plan that is not optimal, not finite or outside its limits is rejected, and a fallback, in most apps the shipped controller, holds the actuator until a plan succeeds. The Model Fitting and Tuning Studio builds the same kind of controller from the reader's own data in its tab, MPC from the fit.

1.6.4 Scenarios, Progress and Data

An app teaches through scenarios, the numbered cases at the top of the Control panel, nine to thirteen in most apps. Loading one sets the model parameters, the controller settings, the disturbances, the mode, the initial state and any scripted events. The information button beside the list opens a briefing in four parts: what the case is, why it matters, what to do and one detail for the practicing engineer. The Learn card holds the lesson back, with Predict and Try It before the run, hints on request and Explain afterward.

Each scenario has a completion rule that the app judges from the run on the simulated clock: a run length, the scripted events passed, a setting made by the reader, a state held for a stated time or a target met. Choosing a case earns nothing. A completed case is marked in the list, and a badge row shows the count and a level, Explorer after one case, Practitioner at half and Completed at all. Progress is kept in the reader's browser with no account. A certificate of progress or of completion can be issued from it, as a record of self-paced work and not an assessed credential.

One case, usually the last, is the graded challenge. It states a target, such as an error band to hold through a sequence of upsets. Changing the model parameters or the disturbances voids the attempt in most apps, and a record made with Reveal suggested tuning says so.

Show Model opens the equations, live values of the variables, the editable parameters and the assumptions. The trends plot SP and PV above the MV for each loop, with actuator limits and marks at setpoint moves, disturbances, mode changes and scripted events. The scorecard reports for each loop the integral of the absolute error (IAE), overshoot, settling time, MV travel and share of time at a limit, with the app's own measures, and it does not combine them into one score. IAE and travel accumulate from Reset, so two runs are compared at the same clock reading.

Export CSV saves the log as a text file with a header row of column names that carry their units and one row per logged sample. In most apps the time column is in seconds, minutes or hours according to the run length. The log holds the most recent 60 000 samples unless the app sets another size. Apps with the MPC option begin the file with comment lines that start with #, which a script must skip.

An app exercise in this book names the scenario by its number and name as the app shows them, every setting that differs from the scenario's defaults, the run length in simulated time, the seed (the default seed unless stated), what to read and where (a trend, a scorecard tile or a column of the exported file) and a tolerance on a numeric answer. Settings are made before Run unless the exercise gives a time.

1.7 Working With AI Agents While Studying

A reader of this book is likely to have an AI assistant within reach, and the apps make its work easy to test. This section gives the editor's view of how to use one while working through a chapter. In short, the assistant may do the algebra and the scripting, and the reader keeps the checking.

What to delegate. Tasks with a clear input and a result that can be tested are good candidates. An assistant can derive the linearization of a chapter's balance at an operating point and the transfer function that follows from it. It can write a short script that reads the exported CSV file and fits a first-order-plus-dead-time model to a step test. It can propose a first tuning from that fit with a named rule, convert units, explain a term in a model or outline a report. The assistant knows nothing about the app until it is told. Give it the equations and parameter values from Show Model, the column names of the exported file and the scenario that produced the data, as described in Section 1.6.4. Ask it to state its assumptions and the units of every result.

What to check, and how. An assistant returns a fluent answer whether or not the answer is right, so each result needs a test that it could fail. Four tests cover most of the errors a chapter will produce.

- *Units.* Every term of an equation carries the same units, and a process gain is in PV units per MV unit. A gain computed with the valve as a fraction is 100 times the gain of a loop whose valve is in percent.
- *Signs.* Decide from the physics which way the PV moves when the MV rises. Compare that with the sign of the fitted gain and with the controller action of Section 1.6.2.
- *Steady state.* Set the derivatives to zero. The model must return the operating point that Show Model reports, and a linearization must give zero deviation there.
- *A run on the app.* Load the named scenario at the default seed, enter the proposed numbers and run. The trend and the scorecard decide. A tuning that reads well on paper and holds the valve at a limit for half the run has failed.

A fitted model gets one more test: compare it with data it was not fitted to, such as a second step in the opposite direction. When a check fails, report the failure to the assistant with the evidence and ask for the cause, then check the correction the same way.

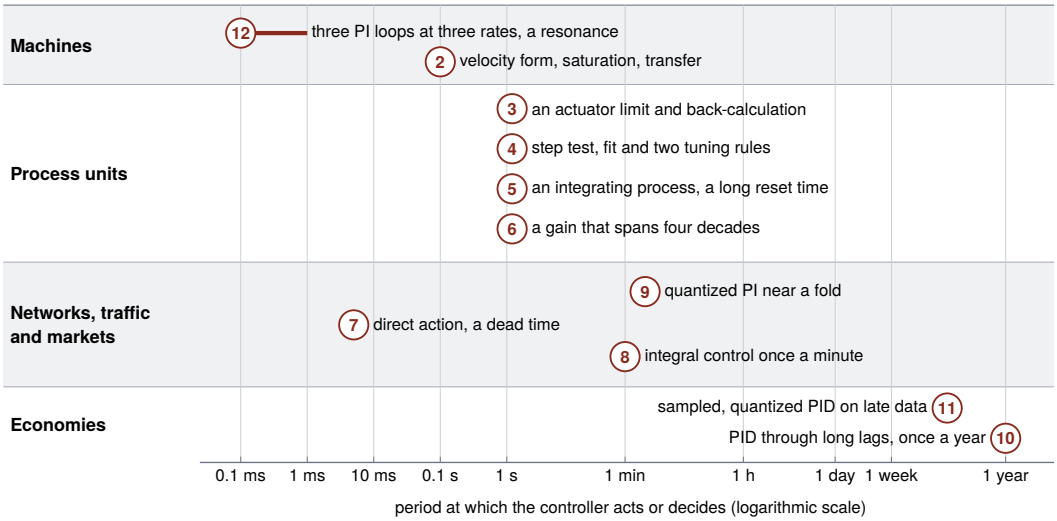


Figure 1.11 Method map of the volume. Each chapter is placed by the period at which its controller acts or decides and by the kind of plant, with its PID lesson from Table 1.7. Periods are those the chapters state for their apps. Notice that one algorithm spans eleven decades of period.

How the exercises train the habit. One exercise in each chapter is marked *Review the agent*. It presents a tuning, a model or a claim as an AI assistant might propose it, with a flaw, and asks the reader to find the flaw in about five minutes. The flaws are ordinary ones: a unit, a sign, a steady state that does not close, a result measured on the wrong scenario or at a different run length, or a limit that was ignored. Each is found with one of the tests above. The purpose is not distrust. An engineer who can say why an answer is right can hand an assistant much larger tasks and still stand behind the result. That larger picture, in which agents do most of the routine work and the engineer supervises and checks, is the subject of Chapter 13.

1.8 Chapter Roadmap

Table 1.7 lists the chapters with the lesson each adds to this one. The level says how much of Section 1.4 a chapter assumes. The time is an estimate for reading the chapter once and running each case of its app once, at about eight minutes a case. Figure 1.11 places the same chapters by the period at which their controllers act, from a tenth of a millisecond to a year.

Chapters 2 to 5 form a first course and are best read in order: the algorithm, a limit, a tuned loop with dead time and an integrating loop. Chapter 6 builds on the tuning of Chapter 4. Chapter 7 needs Section 1.4.12 and stands alone otherwise. Chapters 8 to 11 can be read in any order after Chapters 2 and 3, and Sections 1.4.9 and 1.4.11 prepare them. Chapter 12 uses nearly every subsection of this chapter and is best read last. A reader who has taken a first control course can take a shorter path through Chapters 4, 5, 6, 7 and 12.

Table 1.7 Roadmap of the Volume

Ch.	Plant	The PID lesson	Level	Time
2	Cruise control under quadratic drag	The velocity form, transfer and saturation. Gain and time constant move together with speed.	introductory	2.5 h
3	Stirred blending tank	An actuator limit and back-calculation. An offset that feedback cannot remove.	introductory	2.5 h
4	Steam-heated exchanger with hidden parameters	Step test, fit and two tuning rules on a loop with dead time. A gain that falls with load.	intermediate	4 h
5	Cooling tower basin makeup	An integrating process: slope and delay from a step, long reset time.	intermediate	2.5 h
6	pH neutralization tank	A gain that spans four decades. Scheduling against linearizing the measurement.	advanced	2.5 h
7	Router queue under TCP	Direct action and a dead time that sets the phase margin.	advanced	2.5 h
8	Freeway on-ramp meter	Integral control once a minute, just below a fold in the plant's map.	intermediate	2.5 h
9	Ride-hailing price	PI on a calendar with a quantized, rate-limited output. A gain that grows near a fold.	intermediate	2.5 h
10	National budget and public debt	PID through long lags and an annual decision. The gain the calendar allows.	advanced	2.5 h
11	Central bank policy rate	Sampled, quantized PID on late data. Windup at a lower bound.	advanced	2.5 h
12	Machine-tool servo axis	Three PI loops at three rates. A resonance that caps the gain, and what a relay test measures.	advanced	2.5 h

KEY POINTS

- A PID loop seldom fails because of its equation. It fails at an actuator limit, on noisy or late data, on a slow calendar or at a process gain that has moved since tuning.
- Both model-based tuning rules divide by the dead time, so an error in reading the dead time carries straight into the gain.
- An actuator at a limit breaks the loop, so the apps limit only the output they emit and correct only the integral.
- When the process gain moves with the operating point, a gain schedule holds the loop gain, and it is only as good as its gain curve and its measurement.

Where else this applies. Cascade in Volume 3, overrides in Volume 4, dead time in Volume 5, predictive control in Volume 9.

References

[1] K. J. Åström and T. Hägglund, "The future of PID control," *Control Engineering Practice*, vol. 9, no. 11, pp. 1163–1175, 2001.

[2] ——, *Advanced PID Control*. Research Triangle Park, NC: ISA – The Instrumentation, Systems, and Automation Society, 2006.

[3] D. E. Seborg, T. F. Edgar, D. A. Mellichamp, and F. J. Doyle, III, *Process Dynamics and Control*, 4th ed. Hoboken, NJ: John Wiley & Sons, 2016.

[4] A. Visioli, *Practical PID Control*, ser. Advances in Industrial Control. London: Springer, 2006.

[5] K. J. Åström and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*, 2nd ed. Princeton, NJ: Princeton University Press, 2021.

[6] J. C. Maxwell, "On governors," *Proceedings of the Royal Society of London*, vol. 16, pp. 270–283, 1868.

[7] M. Pitt, "Taking a look back at control: Part 1," *The Chemical Engineer*, no. 988, pp. 51–54, Oct. 2023, thechemicalengineer.com/features/taking-a-look-back-at-control-part-1.

[8] J. G. Ziegler and N. B. Nichols, "Optimum settings for automatic controllers," *Transactions of the ASME*, vol. 64, no. 8, pp. 759–765, 1942.

[9] L. Jacobson, J. Stribiak, L. Nelson, and D. Sallman, “Ramp management and control handbook,” Federal Highway Administration, U.S. Department of Transportation, Washington, DC, Tech. Rep. FHWA-HOP-06-001, Jan. 2006.

[10] M. Pitt, “Taking a look back at control: Part 2,” *The Chemical Engineer*, no. 989, Nov. 2023, thechemicalengineer.com/features/taking-a-look-back-at-control-part-2.

[11] K. J. Åström and T. Hägglund, “Automatic tuning of simple regulators with specifications on phase and amplitude margins,” *Automatica*, vol. 20, no. 5, pp. 645–651, 1984.

[12] V. Jacobson, “Congestion avoidance and control,” in *Symposium Proceedings on Communications Architectures and Protocols (SIGCOMM ’88)*. ACM, 1988, pp. 314–329.

[13] T. Beljean and A. Geier, “The Swiss debt brake — has it been a success?” *Swiss Journal of Economics and Statistics*, vol. 149, no. 2, pp. 115–135, 2013.

[14] “Statement on longer-run goals and monetary policy strategy,” Board of Governors of the Federal Reserve System, Federal Open Market Committee. Adopted effective January 24, 2012; as reaffirmed effective January 27, 2026, 2026, accessed 2026-11-20. [Online]. Available: https://www.federalreserve.gov/monetarypolicy/files/FOMC_LongerRunGoals.pdf

[15] R. Pan, P. Natarajan, F. Baker, and G. White, “Proportional integral controller enhanced (PIE): A lightweight control scheme to address the bufferbloat problem,” RFC Editor, RFC 8033, 2017.

[16] D. E. Rivera, M. Morari, and S. Skogestad, “Internal model control: PID controller design,” *Industrial & Engineering Chemistry Process Design and Development*, vol. 25, no. 1, pp. 252–265, 1986.

[17] S. Skogestad, “Simple analytic rules for model reduction and PID controller tuning,” *Journal of Process Control*, vol. 13, no. 4, pp. 291–309, 2003.

[18] K. J. Åström and T. Hägglund, “Revisiting the Ziegler–Nichols step response method for PID control,” *Journal of Process Control*, vol. 14, no. 6, pp. 635–650, 2004.

[19] B. D. Tyreus and W. L. Luyben, “Tuning PI controllers for integrator/dead time processes,” *Industrial & Engineering Chemistry Research*, vol. 31, no. 11, pp. 2625–2628, 1992.

[20] T. Hägglund, “A unified discussion on signal filtering in PID control,” *Control Engineering Practice*, vol. 21, no. 8, pp. 994–1006, 2013.

[21] K. J. Åström and L. Rundqwist, “Integrator windup and how to avoid it,” in *Proc. 1989 American Control Conference*, 1989, pp. 1693–1698.

[22] M. V. Kothare, P. J. Campo, M. Morari, and C. N. Nett, “A unified framework for the study of anti-windup designs,” *Automatica*, vol. 30, no. 12, pp. 1869–1883, 1994.

[23] R. Hanus, M. Kinnaert, and J.-L. Henrotte, “Conditioning technique, a general anti-windup and bumpless transfer method,” *Automatica*, vol. 23, no. 6, pp. 729–739, 1987.

[24] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control: Analysis and Design*, 2nd ed. Chichester, UK: John Wiley & Sons, 2005.

[25] W. J. Rugh and J. S. Shamma, “Research on gain scheduling,” *Automatica*, vol. 36, no. 10, pp. 1401–1425, 2000.

[26] T. R. Kuphaldt, *Lessons in Industrial Instrumentation*, version 2.33 ed. Open textbook, Creative Commons Attribution 4.0 license, ibiblio.org/kuphaldt/socratic/sinst, 2024.

[27] B. Bok, D. Caratelli, D. Giannone, A. Sbordone, and A. Tambalotti, “Macroeconomic nowcasting and forecasting with big data,” Federal Reserve Bank of New York, Staff Report 830, 2017. [Online]. Available: https://www.newyorkfed.org/medialibrary/media/research/staff_reports/sr830.pdf

[28] GPS.gov, National Coordination Office for Space-Based Positioning, Navigation, and Timing, “GPS accuracy,” 2026, accessed 2026-11-20. [Online]. Available: https://www.gps.gov/gps-accuracy

[29] “Interest on reserve balances,” Board of Governors of the Federal Reserve System, 2026, accessed 2026-11-20. [Online]. Available: https://www.federalreserve.gov/monetarypolicy/reserve-balances.htm

[30] S. J. Qin and T. A. Badgwell, “A survey of industrial model predictive control technology,” *Control Engineering Practice*, vol. 11, no. 7, pp. 733–764, 2003.

[31] K. Stouffer, M. Pease, C. Tang, T. Zimmerman, V. Pillitteri, S. Lightman, A. Hahn, S. Saravia, A. Sherule, and M. Thompson, “Guide to operational technology (OT) security,” National Institute of Standards and Technology, Gaithersburg, MD, Special Publication 800-82r3, 2023.

[32] D. Goldfarb and A. Idnani, “A numerically stable dual method for solving strictly convex quadratic programs,” *Mathematical Programming*, vol. 27, no. 1, pp. 1–33, 1983.

[33] G. Pannocchia and J. B. Rawlings, "Disturbance models for offset-free model-predictive control," *AIChE Journal*, vol. 49, no. 2, pp. 426–437, 2003.

[34] C. V. Rao, J. B. Rawlings, and D. Q. Mayne, "Constrained state estimation for nonlinear discrete-time systems: Stability and moving horizon approximations," *IEEE Transactions on Automatic Control*, vol. 48, no. 2, pp. 246–258, 2003.

1.9 Exercises

1.1 Concept check · 2 min

A reactor is cooled by a jacket valve. Opening the valve lowers the reactor temperature, which is the PV. The controller forms its error as $e = a\,(\mathrm{SP} - \mathrm{PV})$ with a positive gain K_c. Which setting and name are right?

(a) $a = +1$, reverse-acting, so that the output falls as the temperature rises

(b) $a = +1$, direct-acting, because the process gain is negative

(c) $a = -1$, direct-acting, so that the output rises as the temperature rises

(d) $a = -1$, reverse-acting, because the error has changed its sign

1.2 Theory and calculation · 5 min

A series (interacting) PID controller holds the settings $K_c' = 2\,\%/^\circ\mathrm{C}$, $\tau_I' = 5\,\mathrm{min}$ and $\tau_D' = 1\,\mathrm{min}$. Find the ideal-form settings K_c, τ_I and τ_D that give the same controller, and the parallel gains K_i and K_d.

1.3 Theory and calculation · 6 min

A dryer is in Manual at a steady outlet temperature of 60 °C. The heater output is stepped up by 8 % and the temperature settles at 72 °C. It passes 63.4 °C at 4.0 min after the step and 67.6 °C at 9.0 min. Fit a first-order-plus-dead-time model with the two-point method.

1.4 Theory and calculation · 8 min

Tune a controller for the model $K_p = 1.5\,^\circ\mathrm{C}/\%$, $\tau_p = 7.5\,\mathrm{min}$ and $\theta_p = 1.5\,\mathrm{min}$.

(a) Find the SIMC PI settings for $\tau_c = \theta_p$. *Checkpoint:* $K_c = 1.67\,\%/^\circ\mathrm{C}$, $\tau_I = 7.5\,\mathrm{min}$.

(b) Find the Ziegler–Nichols reaction-curve PID settings. *Checkpoint:* $K_c = 4.0\,\%/^\circ\mathrm{C}$, $\tau_I = 3.0\,\mathrm{min}$, $\tau_D = 0.75\,\mathrm{min}$.

(c) Give the phase margin and the gain margin of the loop of part (a). *Checkpoint:* 61.4° and 3.14.

1.5 Concept check · 3 min

A tank level is an integrating process and is held by a PI controller. After the integral time was shortened to speed the loop up, the level rolls slowly about its setpoint and the valve never rests. Which change is the best first correction?

(a) Shorten the integral time further, so the integral catches the error sooner

(b) Lengthen the integral time toward $4\,(\tau_c + \theta_p)$ and keep the gain

(c) Add derivative action with no filter, so the valve leads the level

(d) Change the controller from reverse to direct action

1.6 Theory and calculation · 6 min

A relay test with no switching band moves a heater by $\delta = \pm 5\,\%$. The PV settles into a cycle of $\pm 1.6\,^\circ\mathrm{C}$ with a period of 6.0 min.

(a) Estimate the ultimate gain. *Checkpoint:* $K_u = 3.98\,\%/^\circ\mathrm{C}$.

(b) Give the Tyreus–Luyben PI settings from that estimate. *Checkpoint:* $K_c = 1.24\,\%/^\circ\mathrm{C}$, $\tau_I = 13.2\,\mathrm{min}$.

(c) The recorded PV looks like a triangle wave. Is the true ultimate gain above or below the estimate, and by how much at most? *Checkpoint:* Above, by up to a factor of 1.23, about 4.9.

1.7 Theory and calculation · 6 min

A PI controller with $K_c = 1.25\,\%/°\mathrm{C}$ and $\tau_I = 10\,\mathrm{min}$ has no windup protection. Its output is held at the 100 % limit for 12 min while the error averages 20 °C.

(a) By how much does the integral term grow during that time? *Checkpoint:* 30 %.

(b) The PV then passes the setpoint and stays 5 °C above it. How long does the integral take to give back what it stored? *Checkpoint:* 48 min.

1.8 Concept check · 3 min

A PI controller with $K_c = 2\,\%/°\mathrm{C}$ and reverse action is in Manual with its output at 40 %. The PV is 6 °C below the setpoint. In Manual the algorithm tracks: at each execution it sets its integral so that the output it computes equals the manual output. The loop is switched to Auto. Which value is closest to the first automatic output?

(a) 40 %, the manual output

(b) 52 %, the manual output plus the proportional term

(c) 28 %, the manual output less the proportional term

(d) 100 %, because the integral grew while the loop was in Manual

1.9 Theory and calculation · 7 min

The oven of this chapter has $K_p = 2°\mathrm{C}/\%$, $\tau_p = 10\,\mathrm{min}$ and $\theta_p = 2\,\mathrm{min}$. It runs the SIMC PI with $\tau_c = \theta_p$, so $K_c = 1.25\,\%/°\mathrm{C}$ and $\tau_I = 10\,\mathrm{min}$.

(a) The controller is moved to a system that executes once a minute. Use the half-period rule to estimate the phase margin with the settings unchanged. *Checkpoint:* About 54°.

(b) Retune with the effective dead time and τ_c equal to it. *Checkpoint:* $K_c = 1.0\,\%/°\mathrm{C}$, $\tau_I = 10\,\mathrm{min}$.

1.10 Concept check · 3 min

The oven loop runs the SIMC PI with $\tau_c = \theta_p$. At a new operating point the process gain is twice the fitted value, and the time constant and the dead time are unchanged. What follows?

(a) The loop is unstable, because the loop gain has doubled

(b) Nothing changes, because integral action absorbs a change of process gain

(c) The loop becomes slower, because the closed-loop time constant doubles

(d) The loop is stable with a thin phase margin, and half the controller gain restores it